Actas del

I Congreso Español de Recuperación de Información

Proceedings of the

1st **Spanish Conference on Information Retrieval**

David E. Losada, Pablo Castells, Juan M. Fernández-Luna (Eds.)

Universidad Autónoma de Madrid, Madrid (España), 15 y 16 de junio de 2010

Actas del I Congreso Español de Recuperación de Información
(**Proceedings of the** 1^{st} **Spanish Conference on Information Retrieval**)
http://ir.ii.uam.es/ceri2010/

ISBN: 978-84-693-2200-0 (edición en Internet)

Depósito Legal: GR 2818-2010

Editores:
David E. Losada (david.losada@usc.es), Universidad de Santiago de Compostela.
Pablo Castells (pablo.castells@uam.es), Universidad Autónoma de Madrid.
Juan M. Fernández-Luna (jmfluna@decsai.ugr.es), Universidad de Granada.

Editado en Granada, a 15 de junio de 2010

Patrocinadores del Congreso

Ministerio de Ciencia e Innovación

Universidad Autónoma de Madrid

Consorcio MAVIR

Red gallega de Procesamiento del Lenguaje y Recuperación de Información

Bienvenida del Presidente de Organización

Los organizadores de CERI 2010 damos la bienvenida a todos los participantes y autores que contribuyen a esta primera edición del congreso.

La Recuperación de Información es un área sólidamente establecida a nivel internacional, con identidad bien definida y tradición de varias décadas. El crecimiento de la actividad en el área en nuestro país en los últimos años, tanto a nivel de investigación como de actividad comercial –como así lo refleja la presencia de investigadores españoles en los principales foros científicos, y la orientación de una actividad empresarial importante hacia este campo– hace que éste se haya percibido como un momento propicio para movilizar a una comunidad que se viene fraguando de hecho por suma de iniciativas de grupos y personas independientes. Es en este contexto en el que se gesta la primera edición del Congreso Español de Recuperación de Información.

El evento arranca con vocación de continuidad, con la intención de iniciar una serie estable de congresos periódicos que sirvan de punto de encuentro para investigadores, profesionales y docentes con interés en el área de la Recuperación de Información, en el que se obtenga una visión de la actividad que se realiza en (pero no restringida a) España, sirviendo al tiempo de foro para el debate y fomento de iniciativas orientadas a la promoción del área en nuestro país. CERI busca ser un congreso de referencia en este campo, que aglutine a los mejores investigadores del área en España. Es objetivo del congreso consolidar y promover una comunidad de especialistas en Recuperación de Información, propiciar intercambios y colaboraciones entre éstos, y mejorar la posición de nuestro país en el panorama internacional. CERI quiere ser también un foro abierto a investigadores de otras comunidades, que desde sus respectivas disciplinas, genuinamente confluyentes a la Recuperación de Información, contribuyen de modo natural y/o tienen interés en el área, bien sea como dominio de aplicación, o como campo donde desarrollar métodos propios basados en herramientas de diferentes disciplinas.

El I Congreso Español de Recuperación de Información, CERI 2010, se celebra en la Escuela Politécnica Superior de la Universidad Autónoma de Madrid el 15 y 16 de junio de 2010. Esta primera edición representa una primera toma de contacto donde se podrá valorar el potencial de ésta y otras iniciativas, y perspectivas futuras. En esta línea, se plantea en particular la constitución de una Sociedad de ámbito nacional en Recuperación de Información, para el fomento de esta disciplina en España. El programa del congreso incluye la celebración de una asamblea constituyente en la que se planteará la creación de la sociedad buscando el apoyo más amplio posible de la comunidad.

En nombre de la organización de CERI 2010 quiero agradecer a los conferenciantes invitados, Ricardo Baeza-Yates y Fabio Crestani, su ilustre contribución a esta primera edición del congreso. Nuestra gratitud también a la Universidad Autónoma de Madrid y la Escuela Politécnica Superior, que han facilitado y apoyado la celebración del congreso en esta sede, al Ministerio de Ciencia e Innovación y las redes MAVIR y PL&IR, que patrocinan y colaboran con CERI. Corresponde un importante agradecimiento también al Presidente y miembros del Comité de Programa por su excelente y rigurosa labor, así como a los miembros de los Comités de Organización y Publicidad, cuyo trabajo y dedicación han hecho posible la celebración del evento.

Pablo Castells
Presidente del Comité Organizador de CERI 2010

Bienvenida del Presidente del Comité de Programa

Estas actas contienen los artículos científicos del I Congreso Español en Recuperación de Información (CERI 2010). En la última década, el campo de Recuperación de Información (RI) ha crecido a buen ritmo y la presencia de investigadores españoles en foros internacionales de prestigio como ACM SIGIR o BCS-IRSG ECIR ha sido constante. Sin embargo, la inexistencia de un foro nacional sobre RI impedía de algún modo la interacción regular entre ellos. CERI nace con el objetivo de aglutinar la investigación nacional en RI en España pero, además, fomentando la participación de investigadores extranjeros. Por ello, la llamada a trabajos de esta primera conferencia nacional permitía contribuciones escritas en castellano o en inglés. De este modo, CERI está abierto también a la comunidad internacional y, al mismo tiempo, autores hispanohablantes pueden publicar sus artículos en inglés para que puedan tener mayor difusión externa. CERI tiene la vocación de ser un foro competitivo que pueda convertirse a corto plazo en una referencia sólida y regular para la investigación en RI, con presencia de trabajos que pasan un riguroso proceso de evaluación científica y que puedan tener impacto externo.

La llamada a contribuciones para CERI 2010 dio lugar a un total de 43 envíos de artículos científicos. Como era de esperar, la mayoría procedían de instituciones nacionales (36 artículos) y el resto provenían del Reino Unido, Estados Unidos, Brasil, Irán e India. Muchas contribuciones eran originarias de equipos de universidades pero otras venían de empresas, bibliotecas y centros tecnológicos nacionales e internacionales (8 artículos de este tipo). Las temáticas de las contribuciones eran diversas y con presencia de artículos tanto del ámbito de la Informática como del ámbito de la Biblioteconomía y Documentación.

Todas las contribuciones fueron revisadas por al menos 3 miembros del Comité de Programa. El Comité de Programa de CERI 2010 estuvo compuesto por 44 científicos nacionales y extranjeros procedentes tanto del ámbito académico como del ámbito empresarial. Además de los investigadores de instituciones nacionales más activos en Recuperación de Información, el Comité de Programa contó con la presencia de expertos internacionales procedentes de Yahoo! Research, Google, Microsoft Bing, The Open University, y Universidad de North Texas. El proceso de revisión fue doblemente ciego y en algunos casos, cuando surgían discrepancias entre los revisores, se generó discusión on line a través del sistema web para tratar de aclarar la posición que se debía tomar con respecto al artículo. A veces también se recurrió a una cuarta revisión para tratar de llegar a una decisión lo más informada posible. Finalmente, de los 43 envíos, 20 se aceptaron como articulos completos (12 páginas en las actas y presentación oral), 7 fueron aceptados como artículos cortos (8 páginas en las actas y presentación poster) [1] y 16 artículos han sido rechazados. Aunque inicialmente no disponíamos de la categoría de artículos cortos se llegó a la conclusión de que había una serie de contribuciones que no estaban suficientemente maduras para ser presentadas como artículos completos pero que sin embargo incorporaban algunos aspectos interesantes y novedosos por lo que merecían aparecer de algún modo en CERI 2010. Por ello, se decidió crear la categoría de "artículos cortos", a los que se dedican 8 páginas en las actas y que serán presentados en la conferencia mediante un poster en una sesión dedicada especialmente a ellos.

Como Presidente del Comité de Programa me gustaría agradecer el esfuerzo realizado por los miembros del Comité, que ha permitido definir el programa científico de este primer foro nacional, y a todos los autores por su interés en participar en CERI 2010.

David E. Losada
Presidente del Comité de Programa de CERI 2010

[1] En estas actas no figuran todos los artículos cortos aceptados puesto que algunos autores declinaron el ofrecimiento de presentar su contribución como un artículos corto.

Comité de Organización

Presidente: Pablo Castells, Universidad Autónoma de Madrid.
Alejandro Bellogín, Universidad Autónoma de Madrid.
Iván Cantador, Universidad Autónoma de Madrid.
José Manuel Conde, Universidad Autónoma de Madrid.
Fernando Díez, Universidad Autónoma de Madrid.
Juan Manuel Fernández Luna, Universidad de Granada.
Sergio López, Universidad Autónoma de Madrid.
Mará Medina, Universidad Autónoma de Madrid.
Doroteo Torre, Universidad Autónoma de Madrid.
David Vallet, Universidad Autónoma de Madrid.

Comité de Publicidad

Fidel Cacheda, Universidade da Coruña.
Juan Antonio Martínez Comeche, Universidad Complutense de Madrid.

Comité de Programa

Presidente: David E. Losada, Universidade de Santiago de Compostela.
Enrique Alfonseca, Google Zurich.
Miguel A. Alonso, Universidade da Coruña.
Omar Alonso, Microsoft Bing.
José Luis Alonso Berrocal, Universidad de Salamanca.
Xavier Amatriain, Telefónica I+D.
Ricardo Baeza-Yates, Yahoo! Research, Barcelona.
Alvaro Barreiro, Universidade da Coruña.
Rafael Berlanga, Universitat Jaume I.
Roi Blanco, Yahoo! Research, Barcelona.
Manuel de Buenaga, Universidad Europea de Madrid.
Fidel Cacheda, Universidade da Coruña.
Luis M. de Campos, Universidad de Granada.
Iván Cantador, Universidad Autónoma de Madrid.
Pablo Castells, Universidad Autónoma de Madrid.
Irene Díaz, Universidad de Oviedo.
Miriam Fernández Sánchez, The Open University.
Juan Manuel Fernández Luna, Universidad de Granada.
Pablo de la Fuente, Universidad de Valladolid.
María Fuentes, Universitat Politècnica de Catalunya.
Carlos G. Figuerola, Universidad de Salamanca.
Pablo Gamallo, Universidade de Santiago de Compostela.
Julio Gonzalo, Universidad Nacional de Educación a Distancia.
José María Gómez Hidalgo, Optenet.
Enrique Herrera, Universidad de Granada.
Juan Huete, Universidad de Granada.
Fernando Llopis, Universidad de Alicante.
Juan Llorens, Universidad Carlos III de Madrid.
Ainhoa Llorente, The Open University.
María José Martín Bautista, Universidad de Granada.
Juan Antonio Martínez Comeche, Universidad Complutense de Madrid.
Andrés Montoyo, Universidad de Alicante.
José Ángel Olivas, Universidad de Castilla - La Mancha.
Josep M. Pujol, Telefónica I+D.
José Ranilla, Universidad de Oviedo.
Nieves Rodríguez Brisaboa, Universidade da Coruña.
Paolo Rosso, Universidad Politécnica de Valencia.
Miguel Ruiz, University of North Texas.
David Vallet, Universidad Autónoma de Madrid.
Jesús Vegas, Universidad de Valladolid.
Felisa Verdejo, Universidad Nacional de Educación a Distancia.
José Luis Vicedo, Universidad de Alicante.

Manuel Vilares, Universidade de Vigo.
Ángel Zazo, Universidad de Salamanca.

Revisores adicionales:
Xavier Anguera Miro, Telefónica I+D.
Ronald T. Fernández, Universidade de Santiago de Compostela.
Milagros Fernández Gavilanes, Universidade de Vigo.
Javier Parapar, Universidade da Coruña.
Alfonso E. Romero, Universidad de Granada.
Ismael Sanz, Universitat Jaume I.
Jesús Serrano-Guerrero, Universidad de Castilla - La Mancha.
David Tomás, Universidad de Alicante.
Jesús Vilares, Universidade da Coruña.

Programa Científico

Ponencias Invitadas

Búsqueda Social

Sistemas de Recomendación

Multimedia y Recuperación de Información Geográfica

Distribución de puntuaciones. Compresión

Respuesta Automática a Preguntas. Aprendizaje

Recuperación de Información Estructurada. Recuperación de Información Multilingüe

Artículos Cortos

Mining the Web 2.0 to Improve Search

Ricardo Baeza-Yates

Yahoo! Research Barcelona
rbaeza@acm.org

There are several semantic sources that can be found in the Web that are either explicit, e.g. Wikipedia, or implicit, e.g. derived from Web usage data. Most of them are related to user generated content (UGC) or what is called today the Web 2.0. In this talk we show several applications of mining the wisdom of crowds behind UGC to improve search. We will show current applications on image search, find relations in the Wikipedia and exploiting query logs. Our final goal is to produce a virtuous data feedback circuit to leverage the Web itself.

From Linking Text to Linking Crimes: it is still Information Retrieval, but not as you know it

Fabio Crestani

Faculty of Informatics, University of Lugano, Switzerland
fabio.crestani@unisi.ch

Information retrieval techniques have been used for long time to identify links between textual items, like for example for the automatic construction of hypertexts and electronic books. While interest in this research area has been steadily declining in recent years, some of the techniques developed in that context are proving very valuable in a number of new application areas. In this talk I will present some examples of current work at the University of Lugano. Among other things, I will show how automatically linking newswires from different sources can be used to determine the source timeliness and authoritativeness. Also, I will show how linking police criminal reports can be used to prioritise criminal suspects in a police investigation.

Blog Posts and Comments Extraction and Impact on Retrieval Effectiveness

Javier Parapar, Jorge López-Castro, and Álvaro Barreiro

IRLab, Computer Science Department
University of A Coruña, Spain
{javierparapar,irlab,barreiro}@udc.es

Abstract. This paper is focused on the extraction of certain parts of a blog: the post and the comments, presenting a technique based on the blog structure and its elements attributes, exploiting similarities and conventions among different blog providers or Content Management Systems (CMS). The impact of the extraction process over retrieval tasks is also explored. Separate evaluation is performed for both goals: extraction is evaluated through human inspection of the results of the extraction technique over a sampling of blogs, while retrieval performance is automatically evaluated through standard TREC methodology and the resources provided by the Blog Track. The results show important and significant improvements over a baseline which does not incorporate the extraction approach.

1 Introduction and Motivation

Blog Search has become an important area in Information Retrieval as blogs became a significant portion of the Internet. In a blog, besides objective information, users may express their opinion, intention or many other data which may be valuable to exploit.

The Text Retrieval Conference (TREC) [1] is one of the main reference conferences promoting now Information Retrieval tasks. The TREC is composed of a set of tracks, to cover different areas of IR. In response to the need of exploring blog search behaviour, arises a new TREC track: the Blog Track. The TREC Blog Track includes an opinion finding task and an ad hoc retrieval task. The opinion finding task aims to gather opinion information expressed through a set of blogs on a given topic set. On the other hand, the ad hoc retrieval task consists of measuring retrieval performance on a blog set for a given query set.

In order to perform the Blog Track's tasks, different blog collections are provided to be used as a sample of the blog population present on the Internet. Currently, Blogs06 and Blogs08 collections are available for their use in the research of Information Retrieval Blog Search. These collections store the full web page of a blog post, plus some metadata about the page itself or the recollection process. In a blog post page, the most significant information will be on the post body itself and on the comments sent by the readers. In order to work with this

information, it is be necessary to extract it from the whole page body. This task must be performed in the same way, no matter whether the blog page comes directly from the Internet or from a previously gathered collection: the HTML source code for the page must be parsed and processed in order to extract the text from the post and the comments.

The main problem about extracting information from blog pages is caused by the lack of any standard way for representing the blog post and the comments. This makes necessary to perform a detection process in order to locate the desired information and, once located, extract it. The extraction process becomes difficult because of the heterogeneity present among blogs. This heterogeneity may be mainly of two kinds: format heterogeneity and content heterogeneity. Different blogs differ in the way they represent post or comments, and the HTML markup used to do it, because any page layout can be achieved through many different markup constructions. Besides, the content itself varies a lot not only through different blogs, but also through different posts in the same blog: a post may be composed by a long text or a single sentence or word, may be just a picture, a video or any combination of them. Although a human observer can trivially spot which region of the page belongs to the post and which belongs to the comments, it gets difficult to define a pattern which can be used in an automated process implemented with a computer.

Our approach for the extraction of posts and comments makes use of the similarities on posts and comments layout across blogs from the same service provider and identifies post and comment regions based on the structure of the page and certain attributes present on its nodes.

The rest of this paper is organized as presented next. In section 2 background in blog extraction and blog search will be introduced. Section 3 presents our purposed approach. Sections 4 and 5 present the evaluation methodology used to judge the extraction accuracy and its impact on retrieval, respectively, along with the the results obtained in each evaluation. Finally in section 6 conclusions and future work are presented.

2 Background

Post extraction and comment extraction from a blog page are two variants of the same general process: extracting text regions from a full HTML page. This task, easy for human eye, is quite hard to perform in an automatic way. This is due to the fact that a human observer will rely on the page's layout to spot post or comments region, but this layout is not fully defined in the HTML source code, because it is modified by external resources like the presence of images an the application of external style layout definitions. Two main complementary approaches are suitable for this purpose:

– Choosing text based on its constituent features. The approach here is trying to pick up text matching a set of conditions defined over its content. For example, a blog post usually is the bigger piece of text of a page or is composed by a collection of consecutive paragraphs, while the comments are

often represented as a list of items, or surrounded by metadata like author or posting date.
- Extract text based on its position on the global structure. This approach takes advantage of the HTML code hierarchical structure and tries to locate text based on its place on the node tree representing the web page: the DOM tree.

Both approaches are affected by heterogeneity, because every CMS may publish posts or comments following a radically different format from any other. In many of them, the author is even able of changing this format as desired.

In [2] the authors make an exhaustive analysis of blog comments for a broad sample of blogs and need to deal with the automatic detection/extraction problem. The authors choose a combined approach: in first place, they try to locate the region containing comments based on their position, usually located between the post and some sort of footer. Next, they look for a list of dates in that region and built the comments from the text present around those dates.

In [3] a few guiding lines are suggested for post and comment extraction. Here, the markup differences between different CMS are pointed. A per CMS classification for a sample of 551,763 blogs is provided as well, remarking CMS use proportion and suggesting to give priority to heuristic rule construction for most representative CMS, i.e. Blogger or WordPress, with a 51.29% and an 18.37% share of the total number of blog posts analysed. Table 1 shows the blog distribution given by the authors.

Table 1. CMS distribution for the blog sample used in [3]

CMS	Number of Posts	Percentage
Blogger	282,982	51.29
WordPress	101,355	18.37
LiveJournal	99,100	17.96
MovableType	9,267	1.68
Technorati	3,562	0.65
UserLand	1,869	0.34
FeedCreator	626	0.11
Unknown	53,002	9.60
Total	551,763	100.00

From this table, we can see that three CMS generate 87.62% of the total blog posts taken into account, while the rest have a marginal presence. This supports the approach based on independent analysis of each CMS.

3 DOM and Attribute-Based Blog Parts Extraction

The approach for extracting blog parts explored in this paper relies on DOM structure of the page and the value of attributes present on nodes to identify desired blog parts.

3.1 DOM and Problem Domain

Every web page can be represented as a tree, known as the DOM (Document Object Model). Each content fragment in the page is represented by a node or a set of nodes. Each node may have several attributes which describe properties for the node. With our approach, we assume an extraction rule can be defined based on the position of the nodes within the DOM and the values of their attributes so, once applied over the DOM, it prunes the rest of nodes, leaving only those nodes belonging to the desired content, this is, the post or the comments.

CSS selectors are suitable for defining our extraction rules. CSS (Cascading Style Sheets) is a style sheet language for defining presentation of documents written in a markup language, such as HTML or XML. A CSS selector is a declarative expression which specifies over which DOM nodes is applied a given style, based on their position on the DOM itself and the value of their attributes. So, CSS can be used for the definition of extraction rules.

With CSS selectors as a powerful tool for extraction rule definitions, criteria for guiding de extraction process should be defined. To deal with differences between different blog generators we propose a CMS-based detection process. Each CMS uses a common markup layout or a limited set of variants to represent the blog content, allowing definition of extraction rules for every blog belonging to any given CMS. Besides, a small number of CMS are used as providers for a high percent of the Internet blog population, as seen in section 2.

Focusing in popular CMS, as suggested in [3], shows two advantages: firstly, every improvement made to one of these CMS boosts the number of documents correctly extracted; in second place, minority CMS often mimic popular ones structure, so covering popular CMS may result as well in a better coverage of marginal CMS.

3.2 Detection of CMS and Extraction Rules Definition

In order to take advantage of the knowledge about the CMS distribution of blogs, it becomes necessary to detect the generating CMS for a given blog post. To achieve this, two approaches are proposed: generator tag based detection and URL based detection.

Among the HTML standard tags, there is a tag used to declare attributes defined by key-value pairs. This is the `meta` tag. The general syntax of this tag is `<meta name="someName" content="string defining content" />`. There is a convention for specifying the generator of a page through a `meta` tag, where the `name` key takes the value `generator` and the name of the generator CMS is stored in `content`. The negative side of this approach is that the specification of CMS is not mandatory, so there is no guarantee about this tag being present in a given page. Nevertheless, whenever this information is present, it provides the more accurate information about the CMS used.

Whenever the generator is not specified in the markup as explained before, URL-based detection can be performed. The approach here exploits the presence of certain patterns in blog URLs. For example, Blogger hosted blogs usually

follow the host name pattern `*.blogspot.com`, or MovableType based blog posts are filed under `/movabletype/*` or `/MTarchives/*` paths. All this information is in fact present in the page URL, so parsing the URL may point to the used CMS. This approach is handicapped for the ability of the user to configure these options: i.e. if a blog is hosted in Blogger, a custom domain name may be used whenever the user has the domain registered, which may lead to miss the CMS detection.

These two approaches can be combined, so tag-based detection is attempted in first place, since it provides more accurate results. If no CMS can be determined this way, URL-based detection is performed then.

At this point, extraction rules for known CMS need to be defined, along with some sort of default extraction rule for the case in which the CMS can not be detected. Intuitively, the blog post will be in a container HTML element with some defined style or attributes, while de comment area will be a list of container elements with same style or layout among them. For a given template, the DOM tree path for any of these elements can be defined, but too many elements may match the path defined. So, markup attributes used in the container will be used in conjunction with the tree path to achieve a more accurate detection. Often `class` and `id` attributes hold the proper information for this purpose, with typical values like `post`, `entry` or `content` for the post and `comment` or `cmt` for each comment. Many variations of these values may appear, such as hyphenations (i.e `post-*` or `comment-*`) or translations of these terms to other languages.

With this information, a filter set is created for posts, and another one for comments, where each filter consists of a set of matching CMS patterns and a extraction rule set applicable for a matching page.

3.3 General Extraction Algorithm

The general extraction algorithm is analogous in both cases: post extraction and comment extraction. It consists of two sequential steps: CMS detection and node extraction

In the first step, CMS detection is attempted. Current page is matched against the CMS pattern set, sorted in descendent order by priority criteria. The priority criteria used are:

- Generator matches have higher priority than URL matches, because the generator tag is an explicit CMS or author declaration, while the URL may be more loose pointing to the CMS.
- More restrictive patterns have higher priority. For example, a match of the type `*.blogger.com` will be considered before `*movabletype*`, because a blog which URL is `http://movabletype.blogspot.com` would be hosted in, and generated by, Blogger, even although it matches the MovableType pattern as well.

For a given page, each pattern is checked in priority order, until a match is found. If no match is found, CMS is considered undetected. The extraction strat-

egy for this default case will be sequentially trying every extraction rule defined, ordered by priority, until a node or set of nodes match one of the extraction rules[1].

Once the CMS is detected, the extraction rules defined for that CMS are applied to extract the information. This step varies, depending on whether comment or post extraction is being performed.

In post extraction, this step is divided in three phases:

1. The selected filter is applied, yielding a list of candidate nodes from the DOM for the post.
2. If the candidate node list is not empty, the first of the candidate nodes is selected, discarding the rest. This is done because in case of multiple node matches, usually the post is located on the first node. If no matching candidate nodes are found, the default extraction rule will be tried, because the common case is the blog post being present within the page.
3. The plain text content of the selected node is the detected post.

```
filter bloggerPostFilter
  matches:
    host=*.blogspot.com
    | generator=Blogger*
  extracts:
    div.blogPost
    | div.post>div.post-body
    | div.posts
    | div.blogposts>div
    | div#blog.blog
    | div.text AND NOT div.blogComments
    | div.bpost
    | div.entries>div
    | span.main AND NOT (span.main>span.main OR font>span.main)
end filter
```

Fig. 1. Filter for extracting posts from blogs generated by Blogger. The *matches* section describes the cases in which the filter is applicable while the *extracts* section contains the extraction rule set for the DOM nodes. The `|` operator preceding a rule means it will be tried after its precedent in case of fail. Matching patterns use `*` as wilcard. Extraction rules follow CSS notation where `.` stands for `class`, `#` for `id` and `>` denotes parent-child relationship. `AND`, `OR` and `NOT` operators are used to combine CSS selectors for an extraction rule.

Figure 1 shows an example post extraction filter. This filter is suitable for URLs with hostname ending with `.blogspot.com` or pages whose generator tag holds a value starting with `Blogger`. Whenever the filter is applicable, it will try to extract text from nodes defined like `<div class= "blogPost">` (first extraction rule), then, if no candidates are found, from nodes like `<div class="post-body">` and children of a node `<div class="post">` (second extraction rule) and so on.

[1] To facilitate the reproducibility the rules will be available at http://www.irlab.org

For the comment extraction, the phases are:

1. The selected filter is applied, yielding a list of candidate nodes for the comments.
2. If the candidate node list is not empty, all nodes obtained are selected, since it is expected that each node represents a comment. If no matching nodes are found, no default action is performed, since among the blogosphere is quite usual that a blog post has no comments at all.
3. For each selected node, a comment will be built with the plain text obtained from that node.

Since some CMS use a similar markup for blog post and blog comments, comments could appear included in the post text extracted or paragraphs part of the post could be detected as comments. To avoid this, extraction rules are modified before applying them to the page: nodes matching comment extraction rules are explicitly excluded from the post, while post nodes are excluded from comment results. This allows us to minimize the cases in which comments appear to be part of the post or where the comments are wrongly built from post paragraphs.

4 Extraction Evaluation

The detection and extraction process needs to be evaluated to measure its accuracy and impact over retrieval effectiveness. For the purposes of the TREC Blog Track, collections of blogs were created to provide a realistic sample of the Internet's blog population. The work for this paper was developed working with one of them, the Blogs06 Collection, whose creation process is described in detail in [4]. The table 2 is a summary of the Blogs06 collection and main features.

Table 2. Blogs06 collection composition summary

Measure	Value
Number of unique blogs	100,649
RSS	62%
Atom	38%
First crawl date	06/12/2005
Last crawl date	21/02/2006
Number of feeds obtained	753,681
Number of permalinks	3,215,171
Number of homepages	324,880
Feed size (uncompressed)	38.6GB
Permalink size (uncompressed)	88.8GB
Homepage size (uncompressed)	20.8GB
Total size (compressed)	25GB
Total size (uncompressed)	148GB

The Blogs06 collection is suitable for its use in Blog Search research because it is designed to mimic the whole blogosphere features, such as spam inclusion,

the proportion between highly visited blogs and anonymous ones, as well as professional or business blogs opposite to personal blogs. Besides, as a TREC collection, relevance judgements are available for their use in retrieval evaluation, as is shown in section 5.

4.1 Settings and Methodology

A manual approach has been chosen to evaluate the results of the extraction process. Because of the size of the collection, it is not easy neither feasible manually checking every document in it, so it was decided to choose a random sample of a reasonable size and manually evaluate the process on every document in this sample. To achieve this, a helper application was created.

The evaluation application shows to the evaluator the blog web page rendered on the left panel and the text extracts detected for the post and the comments on this page on the right panel. The evaluator may choose between three options, correct, incorrect and no answer (N/A) if it can't be decided, separately for each extraction target (post and comments), as shown in figure 2.

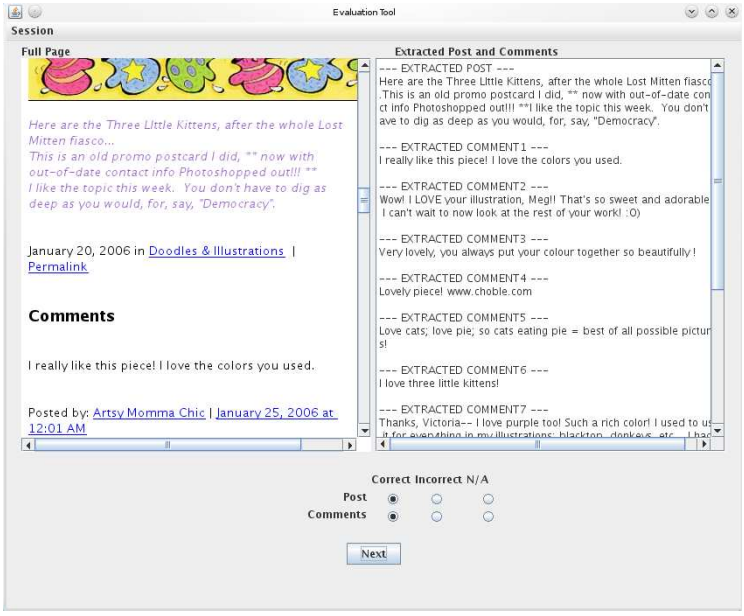


Fig. 2. Screen capture of the evaluation application provided to aid with extraction evaluation

4.2 Evaluation criteria

The human evaluator was instructed to follow the evaluation criteria listed below.

Post extraction is correct whenever one of these conditions is satisfied:

1. The extracted text matches exactly the observed post.
2. The extracted text contains the observed post plus some metadata, such as the posting date, title or the author.
3. The extracted text contains the full observed post plus some marginal text from the page which represents no significant part of the total.
4. The post is neither extracted, nor observed by the evaluator.

Post extraction is incorrect if one of these conditions is satisfied:

1. None of the rightness conditions specified are satisfied.
2. No text is extracted but it is indeed observed by the evaluator.
3. Text is extracted, but the evaluator does not identify a post in the observed page.
4. Marginal text included in the extraction is bigger than the observed post text itself.
5. Comments are included in the extracted post text.

Comment extraction for a document is correct if it satisfies one of these conditions:

1. Extracted text for each comment matches exactly the observed comment.
2. Extracted text for each comment contains the full comment plus metadata, such as posting date, author or title.
3. Extracted comment list contains all observed comments plus some other text conforming fake comments, but these fake comments are less than real comments. An example for this is detecting the comment count as a comment itself, whenever there are one or more observed comments.
4. Every observed comments is detected, although wrongly segmented, i.e. observed comments appear concatenated together as a unique extracted comment.
5. Extracted comment list is empty and the page shows no comments at all.

Comment extraction for a document is wrong if it satisfies one of these conditions:

1. None of the rightness conditions are satisfied.
2. There are extracted comments, but not observed comments.
3. There are not extracted comments, but the comments can be observed on the page.
4. The number of fake comments extracted exceeds the number of proper ones.

Whenever the evaluator is not able to decide whether an extraction satisfies rightness or wrongness conditions, N/A option is assumed as evaluation result.

4.3 Results

The evaluation of the extraction performance was made over a sample of 1,000 documents randomly chosen from the 3,215,171 compounding the Blogs06 collection. The results of the evaluation, obtained following the methodology previously explained, are shown in Table 3.

Table 3. Results of the extraction process for a sample of 1,000 documents. The same sample was used to judge both post and comment extraction.

	Posts	Comments
Correct	821	887
Incorrect	178	112
N/A	1	1
Total	1,000 pages	

As shown in table 3, from the 1,000 random documents, the post was rightly detected in an 82% of them, while the comments were correctly identified on almost an 89%. One of the documents was marked as N/A because it couldn't be evaluated since its markup made the HTML rendering engine used in the application to crash. These figures suggest that the heuristic-based approach chosen is suitable for this problem.

One remarkable fact here is that comment extraction outperformed post extraction. This fact may be due to the fact that commented posts are barely a 31% of the total (997,411 detected, correctly or not, out of 3,215,171) while the extraction reported a post detected for 87% of the documents (2,811,656 of 3,215,171). So, extracting no comments is a good default action since in fact it would be correct around a 70% of the times extraction is attempted, while almost every document in the collection has a blog post. Cases where the extraction process is not able to extract anything usually benefit comment extraction, while penalizing post extraction.

Besides, the extraction process was simultaneously correct for both post and comments in the same document in a 76% of the documents. This reflects a behaviour of some CMS which use a well-defined markup for post but not for comments, or vice versa.

5 Evaluation of the Impact on Retrieval

In order to evaluate how much the extraction process affects depending tasks, we chose to evaluate retrieval over the extracted post and comments and compare it with the retrieval over the page plain text.

5.1 Settings and Methodology

To perform the evaluation, the standard TREC methodology was used, because it is suitable for evaluating large data sets without intervention of a human evaluator. For the TREC Blog Track, a new topic set is provided every year, each of them containing 50 topics. We used the Blogs06 collection as data set, and tested the retrieval with two of these topic sets: topics 851-900 and topics 901-950, topics for year 2006 and year 2007, respectively. Among the fields present for each topic, only the title was used for retrieval, which represents a user query, without any further information added.

For the evaluation task, two indices were created: one holding the full text for each document, obtained extracting the text with a HTML parser from the original pages, and the other containing extracted post concatenated with the extracted comments, obtained through the technique shown in section 3. The index creation was aided by the Lucene [5] library. Our evaluation compares retrieval over these two indices. Higher retrieval effectiveness using the extracted text for post plus comments should point that relevant documents are better found searching over the extracted text and so, meaningful information is condensed within the post and the comments.

We decided to use a high performance state-of-the-art retrieval model: BM25 [6]. The similarity function was computed as follows:

$$R(q,d) = \sum_{t \in q} \frac{tf(t,d)}{k_1((1-b) + b\frac{length(d)}{avg_{dl}}) + tf(t,d))} \times idf(t)$$

Where $tf(t,d)$ is the term t's frequency in document d, $length(d)$ the document length for d, avg_{dl} the average document length across all documents, k_1 is a free parameter, $b \in [0,1]$ as a parameter denoting the degree of length normalization performed and $idf(t)$ is the inverse document frequency of t calculated as $idf(t) = \log \frac{N - df(t) + 0.5}{df(t) + 0.5}$ where N is the number of documents in the colection and $df(t)$ is the document frequency of t.

Using the two topic sets provided, a cross-validation methodology was used for the evaluation. Thus the b parameter for this model was trained using the 851-900 topic set for MAP optimization, yielding the value $b = 0.15$, while the k_1 parameters was left with its recommended value, 2 in this case. Topics 901-950 were used as test set.

To perform a fair evaluation, retrieval on both fields is performed only on those documents for which both post and comments have been extracted. Therefore from 3,215,171 post pages in the Blogs06 collection only 964,019 were selected. This leads to lower effectiveness values when compared to retrieval over the full collection, because part of the relevant documents are not considered.

The measurements used in the evaluation process are those used in the TREC Blog Track for the retrieval task, as seen in [7]: MAP, R-prec and bPref.

5.2 Results

The evaluation results over the Blogs06 collection are summarised in Table 4.

Table 4. Comparison of performance results on retrieval using the full page of the page and the text extracted for the post and comments. Statistical significant improvements according Wilcoxon Signed Rank test ($p\text{-}value < 0.05$) are starred, best values bolded

Topics	*Measure*	*Full text*	*Post + Comments*
	MAP	0.1239	**0.1349*** (+8.88%)
851-900	*R - prec*	0.2093	**0.2193*** (+4.78%)
(training)	*bPref*	0.1974	**0.2186*** (+10.74%)
	MAP	0.1500	**0.1681*** (+12.07%)
901-950	*R - prec*	0.2236	**0.2368*** (+5.90%)
(test)	*bPref*	0.2049	**0.2294*** (+11.96%)

It is shown that retrieval effectiveness over extracted post and comments achieves significant improvements for every measure over the full text of the page, of about a 5% for R-prec and a 10% for MAP and bPref. This experiment demonstrates that using only post and comments as indexing units not only improves the effectiveness of the retrieval but also keeps the indices smaller than when indexing the full text.

6 Conclusions and Future Work

The main objective of this paper was studying an approach to post and comment extraction on blog pages, and how a right extraction may affect depending tasks, such as retrieval.

The obtained results suggest that the followed approach was successful, as it can correctly extract 82% of the blog posts and 89% of its comments, as seen in section 4.

With the extracted post and comments, the retrieval results outperformed the retrieval over the full page text, suggesting that many superfluous text was eliminated and meaningful one was retained.

An interesting future work line on the extraction task should be investigating on CMS-independent text extraction, or systems trained by examples which can learn to identify text regions in a similar way as human eye does. We also plan to do a more exhaustive crowdsourcing [8] evaluation of the extraction process using Amazon Mechanical Turk [9] and our evaluation approach.

Acknowledgments: This work was funded by FEDER, *Ministerio de Ciencia e Innovación* and *Xunta de Galicia* under projects TIN2008-06566-C04-04 and 07SIN005206PR.

References

1. Voorhees, E.M., Harman, D.K.: TREC Experiment and Evaluation in Information Retrieval. MIT Press (2005)
2. Mishne, G., Glance, N.: Leave a reply: An analysis of weblog comments. In: Third annual workshop on the Weblogging ecosystem, Edinburgh, Scotland (2006)
3. Attardi, G., Simi, M.: Blog mining through opinionated words. In Voorhees, E.M., Buckland, L.P., eds.: TREC. Volume Special Publication 500-272., National Institute of Standards and Technology (NIST) (2006)
4. Macdonald, C., Ounis, I.: The TREC Blogs06 collection: Creating and analysing a blog test collection. DCS Technical Report, University of Glasgow (2006)
5. Apache Software Fundation: Lucene, http://lucene.apache.org/. (2010)
6. Jones, K.S., Walker, S., Robertson, S.E.: A probabilistic model of information retrieval: development and comparative experiments. Inf. Process. Manage. **36**(6) (2000) 779–808
7. Ounis, I., de Rijke, M., Macdonald, C., Mishne, G.A., Soboroff, I.: Overview of the TREC-2006 blog track. In: TREC 2006 Working Notes. (2006) 1527
8. Alonso, O., Rose, D.E., Stewart, B.: Crowdsourcing for relevance evaluation. SIGIR Forum **42**(2) (2008) 9–15
9. Amazon: Mechanical Turk, http://www.mturk.com/. (2010)

Contextualización de Búsqueda Web Mediante el Uso de Anotaciones Sociales

Jose M. Conde, David Vallet, Pablo Castells

Universidad Autónoma de Madrid, Madrid, Spain
jose.conde@estudiante.uam.es
{david.vallet, pablo.castells}@uam.es

Resumen. En este trabajo investigamos la adaptación de técnicas implícitas de extracción del contexto del usuario durante la realización de una búsqueda Web. Presentamos varios modelos de obtención del contexto que utilizan la información aportada por servicios Web de anotación de recursos (e.g. Delicious) en vez de la representación textual del documento. Asimismo, presentamos un conjunto de técnicas de reordenación de resultados para adaptar éstos al proceso de búsqueda actual del usuario. Con el objetivo de evaluar las técnicas presentadas, realizamos una evaluación basada en usuarios reales. Los resultados de estas evaluaciones indican que nuestras técnicas de explotación de anotaciones sociales asociadas a documentos son más efectivas para la extracción y aplicación de una representación del contexto del usuario que las basadas únicamente en el contenido textual del documento.

1 Introducción

Con el advenimiento de la Web 2.0, las aplicaciones sociales han crecido de manera exponencial tanto en términos de usuarios como de contenidos. Los usuarios de estos sistemas etiquetan distintos tipos de contenidos con el fin de mejorar su visibilidad respecto a otros usuarios o para permitir una organización propia del contenido. Los dominios de estos sistemas son variados, incluyendo música (Last.fm), fotos (Flickr), o páginas Web (Delicious[1]). Sin embargo, debido al crecimiento exponencial de contenido accesible desde la Web, los usuarios cada vez tienen más problemas para acceder a este contenido, incluso con la ayuda de estas herramientas sociales disponibles en la actualidad. Este problema se conoce como la sobrecarga de información y sigue siendo una línea de investigación abierta dentro del área de la Recuperación de la Información (RI).

El área de la personalización y la contextualización tiene como objetivo aliviar el problema de la sobrecarga de información en la Web. La solución que esta área investiga es tener en cuenta los intereses a largo y corto plazo que puede tener un usuario cuando realiza una tarea de búsqueda, y así adaptar los resultados de la búsqueda a cada usuario. El principal problema al que se enfrentan estos sistemas personalizados es la obtención de los intereses del usuario. Una posible fuente de esta

[1] http://www.delicious.com

información puede ser la actividad de los usuarios en los servicios sociales propios de la Web 2.0. Por ejemplo, la información semántica subyacente asociada a un documento, producto de la etiquetación social que realizan los usuarios sobre cualquier tipo de contenido. Esta información, conocida como *folksonomía*, puede permitir la investigación y desarrollo de nuevos modelos que saquen ventaja de la riqueza de esta información asociada tanto a usuarios como a contenidos. Por ejemplo, las folksonomías se han explotado recientemente para mejorar la búsqueda Web basándose en la popularidad de los documentos [1] o en la personalización [8, 15].

En este trabajo investigamos si sistemas de etiquetado social pueden ser útiles para la captura y explotación implícita (i.e., sin suponer un esfuerzo extra por parte del usuario) de una representación contextual del interés a corto plazo de un usuario. Es decir, se investiga si la información provista por estos servicios puede ser considerada como una nueva fuente del contexto de búsqueda de un usuario. Centramos nuestro análisis en la búsqueda Web y en el uso de la aplicación Web 2.0 Delicious, un servicio de anotación social de recursos Web. Nuestra hipótesis es que sistemas de anotación o etiquetado social como Delicious pueden ser una buena fuente de información a la hora de capturar y explotar la semántica asociada al proceso de búsqueda de un usuario.

El resto de este documento se estructura como sigue. En la sección 2 damos un breve resumen del estado del arte relacionado con nuestro trabajo. En la sección 3 se presenta un nuevo modelo de captura y explotación del contexto del usuario basada en folksonomías. En la sección 4 indicamos una propuesta para la evaluación de nuestro modelo con usuarios reales. En la sección 5 presentamos los resultados obtenidos con nuestros experimentos. Por último, concluimos en la sección 6.

2 Trabajo Relacionado

Los modelos contextuales tradicionales basados en feedback implícito se centran en el procesamiento del contenido asociado a las consultas realizadas por el usuario y el conjunto de documentos accedidos por éste durante la sesión de búsqueda [13]. Esta información se utiliza para construir una representación del contexto actual de búsqueda que puede ser explotado en subsiguientes consultas de la sesión de búsqueda. En este trabajo investigamos la aplicación de dichas técnicas a una representación de documentos Web basada en folksonomías.

El uso de información de anotaciones sociales para la representación y explotación del contexto no se ha investigado en profundidad anteriormente. Una primera iniciativa en la explotación de Delicious como una fuente del contexto del usuario ha sido presentada por Schmidt et al. [9]. En su propuesta, los autores extraen etiquetas relacionadas con documentos abiertos por el usuario, y las aplican a la expansión de consultas, usando factores de decaimiento *ad-hoc* que dan más importancia a documentos abiertos recientemente por el usuario. Sin embargo, este trabajo no presenta una evaluación formal, por lo que la utilidad de usar una representación basada en folksonomías para la representación del contexto aún no se ha demostrado empíricamente. En vez de aplicar técnicas de expansión de consultas [9, 13], en este

trabajo presentamos técnicas de reordenación de resultados. La razón principal es que estás técnicas se pueden aplicar sobre cualquier sistema de búsqueda Web comercial, ya que la reordenación de resultados se realiza sobre cualquier lista de resultados. Además, las técnicas de reordenación de resultados permite centrarnos en la mejora de la precisión en las primeras posiciones de la lista de resultados, ya que en búsqueda Web se considera probado que los usuarios apenas se fijan más allá de la primera página de resultados [3].

Asimismo, en este trabajo investigamos estrategias de representación del contexto más formales [13], tales como el modelo ostensivo [5]. De forma similar a [1, 15], explotamos las anotaciones sociales que alberga Delicious para mejorar la búsqueda Web. Sin embargo, nuestra técnica se basa en la aplicación de un contexto de búsqueda del usuario, o lo que se podría entender como los intereses a corto plazo del usuario, mientras que Bao et al. se centran en la mejora de la búsqueda Web independientemente del usuario [1] y Xu et al. se centran en la personalización de la búsqueda, es decir en los intereses a largo plazo [15]. Por último, cabe precisar que a diferencia del mecanismo de personalización presentado por Xu et al. nuestra técnica no requiere que el usuario tenga un perfil en Delicious, sino que el modelo de contexto se construye en tiempo real, basado en la información de interacción obtenida del usuario durante su proceso de búsqueda y en la información de anotaciones sociales asociada a cada recurso Web provista por los usuarios de Delicious.

3 Un Modelo Contextual Basado en Folksonomías

Definimos una folksonomía $\mathcal{F}$ como una tupla $\mathcal{F} = \{\mathcal{T}, \mathcal{U}, \mathcal{D}, \mathcal{A}\}$, donde $\mathcal{T} = \{t_1, \dots, t_L\}$ es un conjunto de etiquetas que forman parte del vocabulario expresado por la folksonomía, $\mathcal{U} = \{u_1, \dots, u_M\}$ y $\mathcal{D} = \{d_1, \dots, d_N\}$ son, respectivamente, el conjunto de usuarios y el conjunto de documentos anotados por los usuarios, haciendo uso de las etiquetas definidas en $\mathcal{T}$. Finalmente $\mathcal{A} = \{(u_m, t_l, d_n)\} \in \mathcal{U} \times \mathcal{T} \times \mathcal{D}$ representa el conjunto de asignaciones (anotaciones) realizado por un usuario u_m para cada etiqueta t_l a un documento d_n. El perfil de un documento d_n se define como un vector $\overrightarrow{d_n} = (d_{n,1}, \dots, d_{n,L})$, donde $d_{n,l} = |\{(u, t_l, d_n) \in \mathcal{A} | u \in \mathcal{U}\}|$ indica el número de veces que un documento ha sido anotado con la etiqueta t_l. En el escenario de búsqueda Web estudiado en este trabajo, el conjunto de documentos $\mathcal{D}$ representa recursos encontrados en la Web, los cuales se identifican con una URL (*Uniform Resource Locator*).

Para la representación de un proceso de búsqueda de un usuario, definimos el concepto de *trayectoria de consulta*. Este concepto representa la interacción del usuario con un sistema de búsqueda desde el momento que introduce una consulta hasta que introduce una nueva búsqueda o termina de interactuar con el sistema. Definimos por tanto una trayectoria de consulta $tc_i = \{q, d_1, \dots, d_Q\}, d \in \mathcal{D}$, como un conjunto ordenado de elementos en el que el primer elemento es la consulta del usuario (identificada por los términos usados en la consulta) y los subsecuentes indican un conjunto ordenado de documentos a los que el usuario a accedido después

de ejecutar la consulta. La secuencia de documentos está ordenada en función de cuándo el usuario accedió al documento, indicando pues la secuencia temporal de accesos del usuario a resultados de la consulta.

Con objeto de representar una sesión de búsqueda completa, definimos la *trayectoria de sesión* como la secuencia de trayectorias de consulta que realiza un usuario con el fin de satisfacer una necesidad de información dada. De esta forma, una trayectoria de sesión $ts = \{tc_1, \dots, tc_S\}$ se representa como un conjunto ordenado de forma temporal de trayectorias de consulta. En el momento en que el usuario introduce una consulta en el sistema de búsqueda, la trayectoria de sesión representa todas las consultas y accesos a documentos que ha realizado el usuario antes de realizar esa consulta. Esta información es la que puede ser explotada por técnicas implícitas de captura del contexto del usuario basada en folksonomías.

3.1 Captura Implícita del Contexto del Usuario

Definimos una técnica de representación del contexto implícito del usuario como una función $C(ts, \mathrm{t_l})$ que asigna un peso a una etiqueta $\mathrm{t_l}$ en función al grado de relación de esa etiqueta para representar el contexto asociado a la trayectoria de sesión ts. En este trabajo estudiamos un modelo de representación general, basado en el modelo ostensivo de representación del contexto [5]:

$$C(ts, \mathrm{t_l}) = \sum_{i=S}^{1} \alpha^{S-i} \cdot \sum_{d_n \in tc_i} w(d_n, t_l),$$

donde α es un factor de ponderación que considera el orden de las trayectorias de consulta, es decir, cómo de actual es cada trayectoria de consulta respecto al momento actual. La función $w(d_n, t_l)$ asocia un peso a la etiqueta t_l que indica el grado de relación entre la etiqueta y el documento d_n. En este trabajo investigamos dos variaciones de esta función de peso: 1) $tf\text{-}idf(d_n, t_l)$, basada en el valor $tf\text{-}idf$ asociado a un tag, en el que se utiliza $d_{n,l}$ como valor de frecuencia; y 2) $t(d_n) \times tf\text{-}idf(d_n, t_l)$, que multiplica la función anterior por un factor de tiempo $t(d_n)$ que mide el tiempo que ha pasado el usuario observando el documento.

Tabla 1. Modelos de representación implícita del contexto basados en el modelo ostensivo

Nombre	**Valor de α**	**Descripción**
Acumulativo	$\alpha = 1$	Modelo acumulativo, en el que todos los documentos accedidos durante la sesión de búsqueda tienen el mismo peso al representar el contexto del usuario
Ostensivo	$\alpha \in (0,1)$	Modelo ostensivo original [5], en el que los documentos accedidos en momentos más cercanos al actual tienen más peso para representar el contexto del usuario
Comienzo	$\alpha > 1$	Modelo en el que los documentos accedidos en momentos más distantes del actual, dentro de la sesión de búsqueda, tienen más peso para representar el contexto del usuario

Mediante la variación del parámetro α se pueden obtener distintos modelos de representación implícita del contexto [13]. Los modelos estudiados en este trabajo se indican en la Tabla 1. Se incluye el modelo de Comienzo (ver Tabla 1) que fue el que produjo mejores resultados en el estudio realizado en [13].

3.2 Técnicas de Aplicación del Contexto del Usuario

Una vez se ha obtenido la representación del contexto $C(ts, \mathrm{t_l})$ del usuario, se pueden aplicar un número variado de técnicas para contextualizar una lista de resultados. Con el fin de que las técnicas estudiadas se pueden aplicar en cualquier sistema externo de búsqueda Web, en este trabajo nos centraremos en técnicas basadas en la reordenación de resultados.

Un sistema Web no contextualizado S devuelve una lista ordenada de resultados $\mathrm{S}(q) \subseteq \mathcal{D}$ que satisfacen una consulta q introducida por el usuario. Esta ordenación se define de la forma $\tau = [d_1 \geq d_2 \geq \cdots \geq d_k]$, en la cual $d_i \in \mathcal{D}$ y $\geq$ es una relación de ordenación dada por el sistema de búsqueda. Sobre esta ordenación definimos una técnica de contextualización C que provee de una nueva ordenación de documentos $\mathrm{C}(q, ts) \subseteq \mathcal{D}$ mediante la reordenación de los resultados $\mathrm{S}(q)$ de acuerdo a la trayectoria de sesión obtenida implícitamente de las acciones del usuario hasta el momento de realizar la consulta. Formalmente, la técnica de contextualización provee de una ordenación $\tau' = [d_1 \geq d_2 \geq \cdots \geq d_k]$ en la que la relación de orden se define por $d_i \geq d_j \Leftrightarrow \mathrm{sim}(ts, d_i, q) \geq \mathrm{sim}(ts, d_j, q)$, donde $\mathrm{sim}(ts, d, q)$ es una función de similitud entre la representación del contexto actual $C(ts)$ y la representación del documento d, definida como $sim(d_n, C(ts))$, teniendo en consideración la posición original de d en $\mathrm{S}(q)$.

El valor de similitud se puede definir de varias maneras. En la Tabla 2 se indican las funciones de similitud investigados en este trabajo, basadas en un producto escalar, en el cálculo del coseno, y en una adaptación de la medida de BM25 [9].

La reordenación de los resultados por el sistema de búsqueda Web se realiza usando el método de agregación CombSUM con una normalización basada en la posición del documento [11], bien en la ordenación original dada por la búsqueda Web o bien en la ordenación de los documentos en base a alguna de las medidas de similitud con el contexto investigadas.

4 Evaluación del Modelo Contextual en Búsqueda Web

La evaluación de modelos contextuales en sistemas de recuperación de información se reconoce como una tarea compleja de realizar [14]. En el modelo cásico de evaluación de un sistema de RI, conocido como el marco evaluación de Cranfield, se indica que para llevar a cabo una evaluación se requiere de 1) una colección de documentos; 2) un conjunto de descripción de tareas de búsqueda; y 3) un conjunto de juicios de relevancia explícitos que indique qué documentos son relevantes a una tarea de búsqueda dada. En el modelo clásico, tanto la tarea de búsqueda como los juicios de relevancia son independientes del usuario que las realiza. En un modelo de evaluación diseñado para un algoritmo de búsqueda

contextualizada se requiere introducir al usuario como una nueva variable de la evaluación. La información relativa al usuario se introduce en dos puntos distintos. Primero, el punto 3) de juicios de relevancia se altera para que éstos sean dependientes del usuario, esto quiere decir que para una misma tarea de búsqueda (e.g. busca un producto electrónico que te interese) se tendrán distintos documentos relevantes dependiendo del usuario que la realice. Además, se introduce una nueva fuente de información: 4) la interacción del usuario con el sistema, la cual es la fuente de información clave para las distintas técnicas de captura del contexto que evaluamos en este trabajo. Esta información relativa a la interacción del usuario se puede obtener o bien monitorizando a usuarios reales mientras realizan una tarea de búsqueda [12], o bien simulando esta interacción con el sistema, mediante el análisis de logs de consultas [13].

Tabla 2. Medidas de similitud entre el contexto del usuario $C(ts)$ y la representación de un documento d_n

Definición	**Definición**
$escalar(d_n, C(ts))$	$\sum_l d_{n,l} \cdot \mathrm{C}(ts, \mathrm{t_l})$
$coseno(d_n, C(ts))$	$\frac{\sum_l d_{n,l} \cdot \mathrm{C}(ts, \mathrm{t_l})}{\|d_n\|}$
$bm25(d_n, C(ts))$	$\sum_{(l \mid d_{n,l}>0)} idf(t_l) \frac{\mathrm{C}(ts,\mathrm{t_l}) \cdot (k_1+1)}{\mathrm{C}(ts,\mathrm{t_l}) + k_1}, k_1 = 2.0$

Para la evaluación de nuestro modelo de captura implícita y aplicación del contexto, incorporamos las cuatro fuentes de información citadas anteriormente de la siguiente manera: 1) la colección de documentos será cualquier recurso presente en la Web, ya que en este trabajo nos centramos en la aplicación de modelos del contexto en este dominio; 2) adoptamos la descripción de tareas simuladas de Borlund [2], las cuales motivan la necesidad de información que el usuario quiere satisfacer y las pautas que debe seguir para suponer un documento como relevante; 3) usamos usuarios reales en nuestros experimentos, que proveen de juicios de relevancia para cada una de las tareas que realizan; y 4) obtenemos datos de la interacción del usuario del sistema mediante la monitorización del usuario cuando realiza la tarea de búsqueda.

4.1 Preparación de los Experimentos

Para la realización de los experimentos que evalúen las distintas técnicas de captura y explotación del contexto presentadas en este trabajo, reclutamos a 14 usuarios para realizar tres tareas simuladas de búsqueda en la Web. Los usuarios eran en su mayoría estudiantes de doctorado de la institución de los autores. El grupo de estudio se compuso de 10 varones y 4 mujeres, con una mediana de edad de 23 años. El estudio se realizaba en inglés por lo que era un requisito que los usuarios tuvieran un alto conocimiento de este idioma. Los usuarios realizaron cada tarea durante un

máximo de 15 minutos. Se les instruyó a que usaran un navegador Web convencional y usarán el buscador Web Bing[2] de Microsoft para realizar las búsquedas que fueran necesarias de manera que pudieran realizar cada una de las tres tareas.

Cada tarea describe una posible situación real en la que el usuario tiene que utilizar un buscador Web para buscar cierta información. Las tareas se diseñaron para que tuvieran características distintivas, de tal forma que se pudiera estudiar el efecto de la técnica contextual en situaciones diferentes. Las tareas que los usuarios realizaron son las siguientes:

Tarea 1. Encuentra un producto electrónico que te interese (búsqueda dirigida).

Tarea 2. Planifica tus próximas vacaciones (búsqueda exploratoria, multifacética).

Tarea 3. Busca información sobre un tema que te interese (tarea libre).

Las tareas se basan en la clasificación de tareas inicialmente sugerida por Campbell [4]. La primera tarea indica una búsqueda dirigida, en la que se le instruye al usuario que busque información sobre un producto concreto (o una categoría de productos) que le apoye en su decisión de compra: información como especificaciones técnicas del producto, opiniones dadas por revistas electrónicas u otros usuarios, etc. La segunda tarea indica una búsqueda exploratoria con múltiples facetas, en las que el usuario primero debe buscar información del sitio que quiere visitar para sus vacaciones (si no lo tenía decidido anteriormente) y posteriormente debe buscar información suficiente para planificar sus vacaciones en el sitio elegido. Esto requiere buscar información complementaria pero que cubre distintos aspectos tales como el precio del transporte hacia el lugar de vacaciones, qué hotel reservar, qué visitas turísticas se pueden realizar, o recomendaciones sobre el lugar. Por último, en el caso de la última tarea, nos interesa investigar el papel de los modelos del contexto cuando no se le presenta al usuario con una tarea simulada. Por tanto en esta tarea únicamente se le instruyó al usuario que buscara durante el tiempo marcado sobre cualquier tema que le pudiera interesar en ese momento y del que quisiera buscar información. En este caso las tareas de búsqueda de los usuarios fueron muy variadas, desde un usuario que buscó información sobre un concierto de música próximo, hasta un usuario de máster que utilizo este tiempo para profundizar en el estado de arte que tenía asignado en una asignatura. Cada usuario realizaba las tareas siguiendo una ordenación cuadrada latina, de tal forma que el orden de las tareas no influyese en los resultados.

Con el objetivo de obtener juicios de relevancias personalizados para cada usuario y tarea, indicamos a los usuarios que guardasen en los marcadores del navegador aquellas páginas Web que encontraran interesantes para la tarea que estaban realizando. Se evitó indicar que encontrasen el máximo número de documentos relevantes que pudiesen. La intención de estas indicaciones es que nuestra evaluación se desviase lo mínimo posible de una búsqueda Web que pudiera realizar el usuario en una situación normal. Con el fin de obtener un mayor número de juicios de relevancia, al terminar cada tarea se les pedía a los usuarios realizaran juicios de relevancia adicionales sobre una lista agregada de todos los resultados que se les había presentado durante su sesión de búsqueda. Los juicios de relevancia tenían tres niveles: nada relevante, algo relevante y muy relevante.

[2] Microsoft Bing: http://www.bing.com

Esta información, junto con los documentos salvados durante la sesión de búsqueda, conforma los juicios de relevancia utilizados para evaluar cada técnica de contexto.

De modo que se pudiera obtener la información de la interacción del usuario con el navegador y el motor de búsqueda Web, todas las acciones del usuario eran registradas utilizando la barra de herramientas de Lemur[3]. Esta herramienta recoge información tal como qué consulta ha realizado el usuario, el top 50 de documentos devueltos por el motor de búsqueda (en este caso, Bing), qué documentos ha seleccionado el usuario de los resultados, el tiempo que ha pasado el usuario en cada documento, o si ha salvado en sus marcadores el documento. En definitiva, la herramienta nos permitía obtener la información implícita necesaria para ser utilizada por nuestros modelos de captura del contexto. La información sobre anotaciones asociadas a cada página Web se obtuvo a través de la API de Delicious, la cual nos permitía crear la representación del documento basada en folksonomías.

4.2 Procedimiento de Evaluación y Métricas Utilizadas

Debido a que tenemos un alto número de variantes de algoritmos de captura y explotación del contexto, evaluar cada modelo sobre los usuarios reales requeriría un alto número de recursos humanos. Por ello adoptamos un modelo de evaluación en la que monitorizamos a usuarios reales realizando las tareas de búsqueda sobre un sistema base (en este caso un motor de búsqueda Web) y utilizamos los juicios de relevancia para analizar el rendimiento del motor de búsqueda en caso de que sus resultados se reordenaran utilizando las técnicas presentadas.

Los datos de interacción obtenidos de los usuarios son por tanto utilizados para obtener la representación del contexto usando una de las técnicas estudiadas. Por cada consulta que realiza el usuario, se puede comparar las métricas obtenidas con el orden original de los documentos $\mathrm{S}(q) \subseteq \mathcal{D}$ con el nuevo orden $\mathrm{C}(q, ts) \subseteq \mathcal{D}$ producido por la combinación de una técnica de captura del contexto y una técnica de explotación de éste. De este modo el nuevo orden se puede comparar con el caso base u otras técnicas de contextualización de resultados.

La métricas adoptadas para mostrar los resultados son las típicas utilizadas para evaluar sistemas interactivos: MAP (Mean Average Precision), P@10 (Precision hasta la posición 10) y NDCG@50 (Normalized Discounted Cumulative Gain).

5 Resultado de los Experimentos

En esta sección analizaremos el resultado de nuestros experimentos. Primero, analizaremos algunas estadísticas que confirmen que se puede utilizar un servicio de anotación Web como Delicious para extraer el contexto del usuario, aparte de dar información general sobre la interacción de los usuarios con el sistema. Segundo, analizaremos el rendimiento de cada método de captura y explotación del contexto analizado en este trabajo.

[3] Lemur toolbar: http://www.lemurproject.org/querylogtoolbar/

5.1 Interacción del Usuario con el Sistema

En esta sección analizamos los datos de interacción obtenidos de los usuarios del experimento durante la realización de sus tareas. La tabla 3 resume las estadísticas obtenidas, globales para todas las tareas y específicas para cada tarea. En orden, las columnas indican: el número de consultas medio realizado por usuario (consultas), el número de documentos medio accedidos por usuario (documentos), el número de juicios de relevancia medio realizado por cada usuario (juicios), la probabilidad de que un documento accedido por un usuario estuviera anotado en Delicious (prob(D)), la anterior probabilidad condicionada a que el usuario indicó que el documento era relevante (prob(D|R)), y el porcentaje de documentos accedidos por los usuarios que son finalmente marcados como relevantes (ratio(A/R)).

Tabla 3. Estadísticas de la interacción obtenida de los usuarios para todas las tareas y por cada tarea específica.

Tarea	consultas	documentos	juicios	prob(D)	prob(D|R)	ratio(A/R)
Todas	7.16	16.2	13.4	0.379	0.511	0.272
Tarea 1	7.50	17.0	12.1	0.356	0.466	0.243
Tarea 2	7.28	16.6	15.3	0.386	0.493	0.296
Tarea 3	6.71	15.1	12.9	0.398	0.576	0.280

La primera y más importante duda a despejar es si el servicio Delicious es suficiente para obtener la representación del documento basado en folksonomías, es decir, si este servicio tiene la suficiente cobertura para obtener la representación del contexto. Observando los datos de la tabla, todo parece indicar que la cobertura de Delicious es bastante buena. De media, un 38% de documentos accedidos por los usuarios se encuentran en Delicious. Este dato indica sugiere que por sí solo el servicio de Delicious ofrece la cobertura suficiente para ofrecer una contextualización de los resultados basada en folksonomías, hecho que se corrobora en los resultados de los experimentos analizados más adelante. Además, esta probabilidad aumenta cuando el documento es relevante al usuario, con un 51% de media.

Observando los datos desglosados por tarea, podemos concluir que las estadísticas son similares en cada tarea, salvo algunas excepciones. La primera excepción es la tarea 3 que de media tiene ligeramente menos consultas y documentos accedidos que las otras tareas, pero por otro lado la probabilidad de que un documento se encuentre en Delicious condicionada a que éste sea relevante es mayor que las otras tareas. Además cabe destacar que la tarea 2 tiene de media más juicios de relevancia que las otras dos tareas. Este resultado sería consistente con la definición de la tarea, que al ser exploratoria y tener múltiples facetas, requiere evaluar más documentos para cubrir cada faceta de la tarea.

5.2 Rendimientos de los modelos de captura del contexto

En esta sección analizaremos los datos de las métricas obtenidos tras seguir el proceso de evaluación presentado en la sección 4. La Tabla 4 muestra los valores de

MAP para las distintas combinaciones de modelo de captura del contexto basado en folksonomías y función de similitud aplicada en su explotación. El caso base en esta y subsiguientes tablas es la ordenación original devuelta por el motor de búsqueda Bing.

Tabla 4. Valores de MAP para las distintas técnicas del contexto inspeccionadas y distintas funciones de similitud. En negrita la combinación con mejor rendimiento.

Contexto / Similitud	Caso base	Acumulativo $\alpha = 1$	Ostensivo $\alpha = 0.5$	Comienzo $\alpha = 1.5$	ostensivo + tiempo
escalar		0.1005	0.1005	0.0997	**0.1013**
coseno	0.0969	0.0971	0.0962	0.0967	0.0965
bm25		0.1000	0.1000	0.1003	0.0999

Los resultados muestran que usando las medidas de similitud basadas en el producto escalar y BM25 se consigue una mejora con respecto con respecto al caso base. La medida de similitud basada en el cálculo del coseno no parece ser la adecuada para aplicar en este caso una representación del contexto basada en folksonomías. Mientras que las cuatro primeras columnas utilizan la función de peso de etiquetas basada en $tf\text{-}idf$, la última columna muestra el uso de la representación del contexto que sigue el modelo ostensivo y además incorpora el factor de tiempo de visualización de un documento para asignar un peso a cada etiqueta asociada al documento. Esta última medida es la que mejor rendimiento produce, consiguiendo una mejora de en torno el 4.5%. Esta mejora es estadísticamente significativa (Wilcoxon, $p > 0.05$). Para las otras dos métricas contempladas, P@10 y NDCG@50 los valores obtenidos están correlacionados con la medida MAP.

Con el fin de investigar si estos valores varían según el tipo de tarea que realiza el usuario, en la Tabla 5 se desglosan por tarea los valores de MAP para la medida con mejor rendimiento mostrada en la Tabla 4, es decir el modelo ostensivo con factor de tiempo y producto escalar como medida de similitud.

Tabla 5. Valores de MAP por tareas para el modelo ostensivo de captura del contexto y factor de tiempo, con uso de producto escalar como función de aplicación del contexto.

Tarea	1	2	3
Ostensivo+tiempo	0.1071	0.0980	0.0992
Caso base	0.0962	0.1002	0.0951

El desglose por tareas muestra que nuestra técnica de captura y explotación del contexto mejora al caso base en dos de las tres tareas. Es en la primera tarea en el que la contextualización de los resultados produce una mayor mejora, en torno al 11.3%. Esto puede ser debido a que la primera tarea ejemplifica una búsqueda dirigida, en el que el usuario se centra en una única faceta de información- La representación del contexto, por tanto, es relevante en cualquier punto de la sesión de búsqueda y produce una mejora para todas las consultas. Por el contrario, no se mejora el caso base en la tarea 2. Este resultado se puede haber producido por el carácter exploratorio y multi-faceta de esta tarea, en el que en el caso de que un usuario cambie de faceta dentro de la sesión de búsqueda, la representación del contexto obtenida hasta el momento puede no ser relevante. Cabe destacar que la bajada de rendimiento en esta tarea no es

significativa. Por último la técnica de contextualización de resultados obtiene una mejora sobre la tarea 3. Este resultado es muy interesante, ya que a pesar de ser una tarea en el que el usuario podría centrarse en el tema que quisiera, nuestras técnicas siguen mejorando al caso base.

Por último, consideramos importante analizar nuestra técnica de captura del contexto basada en folksonomías con las mismas técnicas usando el contenido textual asociado a cada documento. Este análisis se realiza a fin de observar cual es la ganancia obtenida con el uso de una representación basada en folksonomías, en comparación con una representación textual basada en palabras clave. Para realizar esta comparación modificamos las técnicas de captura del contexto para que extraigan términos asociados al documento en vez de usar el servicio Delicious. En este caso obtenemos una mejor cobertura que Delicious, ya que prácticamente la totalidad de los documentos accedidos por los usuarios tienen texto asociado. Los término asociados a cada documentos son limpiados de palabras comunes, o *stopwords*, y repesados con un valor de $tf\text{-}idf$. La tabla 6 muestra la comparación entre nuestras técnicas de captura y explotación del contexto usando las etiquetas obtenidas a través de Delicious y nuestras mismas técnicas usando el texto asociado al documento.

Tabla 6. Valores de MAP para las distintas técnicas del contexto inspeccionadas usando el producto escalar como valor de similitud y dos distintas fuentes de contenido del documento.

Fuente \ Contexto	Caso base	Acumulativo	Ostensivo	Comienzo	ostensivo + tiempo
Delicious	0.0969	0.1005	0.1005	0.0997	**0.1013**
Texto		0.0895	0.0881	0.0877	0.0870

Los resultados muestran que nuestras técnicas no serían efectivas si se usase el contenido textual del documento para la captura del contexto, ya que en ningún caso se mejora el caso base o a las mismas técnicas basadas en la etiquetación dada por otros usuarios. Este resultado indica que la representación creada mediante la anotación social de los usuarios es más concisa y rica a la hora de extraer y aplicar una representación del contexto del usuario.

6 Conclusiones

En este trabajo se presenta un conjunto de técnicas de captura y explotación implícita del contexto de búsqueda de un usuario basadas en la representación folksonómica de un documento, resultado de la anotación social de los usuarios. Los resultados obtenidos en nuestros experimentos indican que nuestra representación basada en folksonomías asociadas a los documentos es más eficiente que modelos tradicionales, los cuales explotan y representan el contexto del usuario mediante el tratamiento del contenido textual asociado a cada documento.

Los resultados desglosados por tareas indican que nuestra técnica no es efectiva para tareas que comprenden múltiples facetas, ya que la representación del contexto del usuario obtenida durante la sesión de búsqueda no es siempre relevante para las distintas facetas contenidas en este tipo de búsqueda exploratoria. Técnicas que

detecten un cambio de foco durante la sesión de búsqueda del usuario [7] pueden ayudar a mejorar nuestras técnicas en este tipo de tareas.

Agradecimientos

Este trabajo ha sido financiado por el por el Ministerio de Ciencias y Educación (TIN2008-06566-C04-02) y la Comunidad de Madrid (S2009TIC-1542)

Referencias

1. Bao, S., Xue, G., Wu, X., Yu, Y., Fei, B., Su, Z.: Optimizing web search using social annotations. In: Proc. of WWW 2007, pp. 501-510. ACM Press, New York (2007)
2. Borlund, P. The iir evaluation model: a framework for evaluation of interactive information retrieval systems. Information Research, 8(3), paper no. 152 (2003)
3. Broder, A.: A taxonomy of web search. *SIGIR Forum*, 36(2), pp. 3-10 (2002).
4. Campbell, D. J. Task Complexity: A Review and Analysis. In Academy of Management Review 13(1), 40-52 (1988)
5. Campbell, I. and van Rijsbergen, C. J.. The ostensive model of developing information needs. In COLIS'96, pp. 251-268 (1996)
6. Cleverdon, C. W., J. Mills and M. Keen. Factors determining the performance of indexing systems. ASLIB Cranfield project, Cranfield (1966)
7. Huang, X., F. Peng, A. An and D. Schuurmans. Dynamic web log session identification with statistical language models. Journal of the American Society for Information Science and Technology 55(14), 1290-1303. (2004)
8. Micarelli, A., Gasparetti, F., Sciarrone, F., Gauch, S.: Personalized search on the World Wide Web. The Adaptive Web. LNCS, vol. 4321, pp. 195-230. Springer, Heidelberg (2007)
9. Robertson, S. E. Walker, S: Some simple effective approximations to the 2-Poisson model for probabilistic weighted retrieval. In SIGIR '04: Proceedings of the 27th Annual International ACM SIGIR conference, pp. 345-354. Springer (2004)
10. Schmidt, K. U., Sarnow, T., and Stojanovic, L. Socially fil-tered web search: an approach using social bookmarking tags to personalize web search. In SAC'09, pages 670-674, (2009)
11. Shaw, J. A. and Fox, E. A. Combination of multiple searches. In: Text REtrieval Conference, pp. 243-252 (1993)
12. White, R. W. and Kelly, D. A study on the effects of personalization and task information on implicit feedback performance. In CIKM '06: Proceedings of the 15th ACM international conference on Information and knowledge management, pp 297-306 (2006)
13. White, R. W., Ruthven, I., Jose, J. M., and Van Rijsbergen, C. J. Evaluating implicit feedback models using searcher simulations. ACM TOIS., 23(3), pp. 325-361 (2005)
14. Wilkinson, R. and Wu, M. Evaluation experiments and experience from the perspective of interactive information retrieval. In: Proceedings of the 3rd Workshop on Empirical Evaluation of Adaptive Systems, in conjunction with the 2nd International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems, pp. 221-230 (2004)
15. Xu, S., Bao, S., Fei, B., Su, Z., Yu, Y.: Exploring folksonomy for personalized search. In: Proc. of SIGIR 2008, pp. 155-162, ACM Press, New York (2008)

Combining Document and Sentence Scores for Blog Topic Retrieval

Jose M. Chenlo, David E. Losada

Grupo de Sistemas Inteligentes
Departamento de Electrónica y Comunicación
Universidad de Santiago de Compostela, Spain
{josemanuel.gonzalez,david.losada}@usc.es
http://www.gsi.dec.usc.es/ir

Abstract. In recent years topic retrieval has become a core component in blog information retrieval. In this area, non-relevant documents that contain many query terms by chance or in the wrong context might be highly ranked. To address this issue, in this paper we propose some adjustments to effective blog retrieval methods based on the distribution of sentence scores. We hypothesize that we can successfully identify truly relevant documents by combining score features from document and sentences. This helps to detect right contexts related to queries. Our experimental results show that some of our proposed variants can outperform state-of-the-art blog topic retrieval models.

Key words: blog retrieval, query context, sentence retrieval

1 Introduction

With the rise of social networks and blogs, new opportunities and challenges appear to handle properly the opinionated nature of these resources. Our research is focused on one of the most important web sentiment-oriented resources: the blogosphere. People read frequently blogs to be acquainted with others' viewpoints. In order to build an effective information retrieval (IR) system which helps users to find what other people think, we have to consider opinions as a first-class object [1].

In the literature, most blog retrieval tasks are approached by a two-phase process that involves a topic retrieval task and a re-ranking phase based on opinionated features. The performance of the opinion finding system has proved to be strongly dependent on the quality of the initial topic retrieval process [2]. We design here powerful retrieval algorithms able to support successfully the subsequent opinion mining tasks. More specifically, we propose a new method to support blog IR based on the distribution of salient sentences in documents retrieved by state of the art models. A high sentence score is associated to a good matching with the query. Hence, by promoting documents with high score sentences, we could successfully identify documents that contain terms related to the query in the right context. By analyzing the flow of the presumed relevant

sentences in the BLOGS06 TREC collection [3] we were able to distinguish relevance-flow patterns that help us to identify relevant documents.

Our experiments show that this sentence-level information can be useful to push defective documents lower into the rank. Specifically, we consider several features such as the ratio of high-scored sentences in a document (peaks), the median number of unique terms matched for each sentence, the variance of sentence scores in a document and the maximum score of the sentences belonging to a document. These sentence-level features provide valuable information about the way in which a document matches a query.

The rest of the paper is organized as follows. Section 2 presents related work. Standard blog retrieval methods and the methodology followed to combine sentence retrieval scores with document scores are explained in Section 3. Section 4 reports the experiments and analyzes their outcomes. The paper ends with Section 5, where we present the conclusions and future lines of work.

2 Related Work

The main promoter of research in blog retrieval has been the TREC conference. The blog track was introduced as a main task in TREC 2006, continued in TREC 2007, 2008 and 2009 [4,5,2,6] and is running in TREC 2010. One of the lessons learnt from early blog tracks is that opinion-finding is strongly dominated by the underlying document ranking performance (topic-relevance baseline). As a matter of fact, rankings provided by good topic-retrieval baselines perform reasonably well for the opinion-finding task without any opinion-based adjustment. In past blog tracks, many research teams did not manage to improve the opinion-finding performance of their own topic-relevance baselines by applying opinion mining techniques. This motivated some adjustments in the proposed tasks. Since TREC 2008, the blog opinion retrieval task (the main task of blog track) was divided into two independent subtasks. In the first subtask (baseline adhoc retrieval task), traditional IR is used to find topically relevant documents. The second task (opinion-finding retrieval task) consists of finding subjective (positive/negative/neutral) posts and researchers were encouraged to apply their opinion-finding techniques on their own retrieval baselines or on a set of baselines provided by TREC. The idea was to provide the participating groups with an experimental setting where they could assess the impact of their opinion-finding techniques across a set of different topic-relevance baselines. Through this experiment, the Blog track went towards a better understanding of the most effective and stable opinion-finding techniques, by observing their performances on common topic-relevance baselines.

Our work is focused on blog topic retrieval and, more specifically, on the use of sentence-based features to improve these systems. Sentence or passage-level evidence has shown its merits in blog retrieval environments. Lee et al. [7] used a passage-based retrieval model combined with a feedback engine in TREC 2008 and their results demonstrated the strength of their system, being one of the best topic retrieval approaches in the track. Santos et al. [8] applied a novel approach

that takes into account the proximity of query terms to subjective sentences in a document. This led to improvements over state of the art baselines for blog opinion retrieval. We study here a new sentence-level approach based on the relevance flow of sentence scores.

Sentence-level features have also been used in other scenarios. Seo and Jeon [9] built a high precision document retrieval system supported by several sentence-based features. Specifically, they used features based on the position and distribution of the high-scored sentences to learn a probabilistic model that can distinguish relevance-flow patterns for relevant documents. Passage-level evidence has been recurrently applied in passage-based document retrieval. For instance, in [10], Kaszkiel and Zobel proposed several methods to estimate the relevance of documents by aggregating query-passage scores. Finally, the relation between relevance and subjectivity has been explored in other scenarios such as sentence retrieval [11].

3 Blog Topic Retrieval

Before we go into details about our method it is necessary to enumerate the main design points in blog information retrieval systems:

- *Retrieval Unit:* Many researchers choose the documents from blog permalinks [1] as retrieval units of their systems. The permalink document contains complete information about the blog entry: title, post, comments and many noisy components that is necessary to deal with. Other studies opted for fine-grained retrieval units. Recently, Lee et al. have successfully applied retrieval at passage level and, next, the passage scores were aggregated to build a permalink retrieval system [7]. The selection of the retrieval unit is therefore an open issue in blog retrieval systems.
- *Document preprocessing:* There is much noise in the structure of blog pages. In most permalink documents we can find off-topic data such as links to other blog posts, information related to the blog and many advertisements. This data might harm severely retrieval performance by promoting deceptive documents that have query terms in a wrong context. Effective preprocessing to identify the key components of the permalink (i.e. title, post and comments) and discard noisy pieces of information is necessary to obtain a good performance.
- *Topic Retrieval Method:* In past TREC tasks state of the art models (e.g. BM25, Language Models) worked very well in blog information retrieval, being extremely difficult to beat [2].

More details about these stages of the blog retrieval process and our design decisions are explained below.

[1] The permanent link to a specific page within a blog or post that remains unchanged. Permalinks are useful for bookmarking or tagging a specific blog post for future reference.

3.1 Retrieval Unit and Document preprocessing

The first decision was to set the unit of retrieval of our system. We opted for one of the most common units used in past TREC tracks: permalink documents from blog pages. The selection of this piece of information simplifies the evaluation of our system because the TREC blog track demands the construction of a ranking of permalinks (ordered decreasingly by presumed relevance).

Permalink files have useless information within their internal structure (e.g. links and advertisements). To remove noise we have built a preprocessing unit able to identify the main permalink components (i.e. title, post and comments) and discard the rest of the documents' content. This unit uses a common HTML parser [12] to process the structure of permalinks documents and a set of heuristics to find the core components. The main idea is to detect pieces of text in different HTML blocks and then classify them according to their positional information and size. This type of heuristics has been used in other contexts with relative success [13].

3.2 Baseline Retrieval

Once we have defined and correctly processed the documents of our collection, we need an effective information retrieval model. We decided to use the BM25 model [14] because it has been one of the strongest and powerful methods used since 1994 in the information retrieval field [15]. We used the Lemur's implementation of BM25 matching function [16]:

$$w = \log\left(\frac{N - n + 0.5}{n + 0.5}\right) \tag{1}$$

$$BM25(D,Q) = \sum_{t \in Q} w \cdot \frac{(k_1 + 1)\, tf_{t,D}}{k_1\left((1-b) + b \times (L_D / L_{ave})\right) + tf_{t,D}} \frac{(k_3 + 1)\, tf_{t,Q}}{K_3 + tf_{t,Q}} \tag{2}$$

where N is the total number of documents in the collection, n is number of documents that contain the term t, $tf_{t,D}$ is the frequency of t in document D, $tf_{t,Q}$ is the frequency of t in query Q, L_D and L_{ave} are the length of document D and the average document length in the whole collection. b, $K1$ and $K3$ are free parameters that we have to train.

3.3 Obtaining Sentence Scores

The sentence retrieval module is composed of two components: a preprocessing component and a weighting component. Given the collection of blogs, we split the documents into sentences (offline, more details about the splitting process are given in section 4). For efficiency reasons, we store all sentences in an inverted index. Finally, at query time, we only have to retrieve all the sentences of the collection that match at least one query term.

Finally, we have to select the appropriate weighting model for Sentence Retrieval (SR). Traditional SR methods proposed in the literature are often based on a regular matching between the query and every sentence. A vector-space approach, the tfisf model [17], is a simple but very effective SR method [18]. It is parameter-free but performs at least as well as tuned SR methods based on Language Models or BM25. The tfisf matching function is:

$$tfisf(S,Q) = \sum_{t \in Q} \log\left(tf_{t,Q} + 1\right) \log\left(tf_{t,S} + 1\right) \log\left(\frac{n+1}{0.5 + sf_t}\right) \quad (3)$$

where $tf_{t,Q}$ and $tf_{t,S}$ are the number of occurrences of the term t in the query Q and sentence S, respectively; sf_t is the number of sentences where t appears, and n is the total number of sentences in the collection.

3.4 Combining Document and Sentence Scores

Once we have selected document-query and sentence-query matching functions, it is necessary to define a combination method that assigns a single score to every document. For efficiency reasons, our adjustment was modeled as a re-ranking process for each query, in which we only re-rank the top-ranked documents retrieved by BM25 $\left(D_{T_q}\right)$[2].

First, to have the scores in the same scale ([0,1]) we apply the following normalization:

$$BM25_{norm}(D,Q) = \frac{BM25(D,Q)}{max_{D_i \in C} BM25(D_i,Q)} \quad (4)$$

$$tfisf_{norm}(S,Q) = \frac{tfisf(S,Q)}{max_{S_i \in D_{T_q}} tfisf(S_i,Q)} \quad (5)$$

where C is the collection of documents and S_i are the sentences belonging to top retrieved documents. Observe that the normalization in equation 5 considers all sentences in the BM25's top 1000 retrieved documents. In order to obtain these sentences, for each query we rank all sentences in the collection (using our inverted index of sentences) and then we filter out the sentences belonging to documents that are not in the top-ranked list.

Our objective in this paper is to apply sentence-level features able to clarify the pattern of matching between relevant documents and queries. The features used in our study are the following[3]:

- *Ratio of peaks:* The ratio of peaks in a document, being a peak each sentence of the document which has a normalized score greater than 0.5. We calculate the ratio of peaks as $(\#peaks/\#sentences)$. This measure might help us to promote documents with several sentences very similar to the query rather

[2] For each query, we re-rank the first 1000 documents from the BM25 retrieval ranking.
[3] Variance and Median were calculated only for sentences that match at least one query term.

than documents that contain many query terms scattered. We chose this threshold because the number of sentences that could be considered as a peak should be low (as a matter of fact many documents have no peaks). Hence, we hypothesize that the use of this threshold will be useful to detect only the best query-related sentences. The same threshold has been used in other studies to model the importance of sentences in documents [9].

- *Variance:* The variance of non-zero sentence scores in the document. With this feature we can model the tendency of query matching throughout the document. It is interesting to detect these matching trends in relevant documents and to study how they differ from the non-relevant documents trends.
- *MedianU:* The median of the number of unique query terms matched by the sentence in the document. We hypothesize that documents with high median score are potentially relevant because they contain many sentences on topic. In contrast, documents with low median scores have sentences with poor matching with the query. By incorporating this feature we expect to detect right query-context in documents.
- *Max:* The maximum score of the sentences belonging to a document. This measure promotes documents with a sentence that is presumably very related to the query. This is similar to passage-based retrieval methods that estimate relevance using the highest score passage.

The function used to combine the document and sentence-level score features is simply:

$$sim(D,Q) = \alpha \cdot BM25_{norm}(D,Q) + \beta \cdot SF_{norm}(D,Q) \quad (6)$$

where $SF_{norm}(D,Q)$ is one of the scores explained above and α and β are free parameters trained by linear regression [19].

In this paper we only consider the individual incorporation of these features in our model. Combinations of multiple features and more formal combination models (which take into account the possible dependency between document and sentence features) will be studied in the future.

4 Experiments

In the evaluation, we chose the Lemur Toolkit for indexing and retrieval tasks [16]. The remainder of this section details the aspects related with our experimentation methods and the results obtained.

4.1 Collection and Topics

Our experiments are based on the BLOGS06 TREC collection [3], which is one of the most renowned blog test collections with relevance assessments. Each token has been preprocessed with Porter stemmer and standard English stopwords have been removed. Some statistics of the collection are reported in Table 1.

Table 1. The main statistics of the BLOGS06 collection. This collection was utilized in the TREC 2006, TREC 2007 and TREC 2008 blog tracks

Attribute	Value
Number of Unique Blogs	100,649
RSS	62%
Atom	38%
First Feed Crawl	06/12/2005
Last Feed Crawl	21/02/2006
Number of Feeds Fetches	753,681
Number of Permalinks	3,215,171
Number of Homepages	324,880
Total Compressed Size	25GB
Total Uncompressed Size	148GB
Feeds (Uncompressed)	38.6GB
Permalinks (Uncompressed)	88.8GB
Homepages (Uncompressed)	20.8GB

In our experiments we worked with the TREC 2006, TREC 2007 and TREC 2008 blog track test collections. All of them utilize the BLOGS06 document collection as the reference collection for the retrieval experiments. Every year a new set of topics was provided and new relevance judgments were made according to the documents retrieved by participants. Details about these topics are reported in Table 2. We built three realistic and chronologically organised query datasets with these topics. First of all, to train the BM25 free parameters we used the TREC 2006 topics for training and the TREC 2007 and TREC 2008 topics for testing. We applied two different training/testing configurations for our regression model. Details about these configurations are shown in Table 3.

Table 2. Topics provided in the TREC Blog tracks

Blog Track	Topics(#)
2006	851-900 (50)
2007	901-950 (50)
2008	1001-1050 (50)

Table 3. Training and testing configurations used for regression

Label	Training	Testing
TREC 2007	851-900(2006)	901-950(2007)
TREC 2008	851-900(2006), 901-950(2007)	1001-1050(2008)

Each TREC topic contains three different fields (i.e. title, description and narrative) but we only utilized the title field, which is short and the best representation of real user web queries, as reflected in the official TREC Blog track literature [4,5,2].

The measures applied to evaluate the retrieval performance in our system were mean average precision (MAP), Precision at 10 documents (P@10) and Precision at 5 documents (P@5), which are commonly used in blog environments.

Finally, in order to detect all the sentences contained in the documents we used a Java port of the Carnergie Mellon University link grammar parser [20]. This software is included in the MorphAdorner project [21], developed by Northwestern University of Chicago. The link grammar parser is a natural language parser based on link grammar theory. Given a sentence, the system assigns to the sentence a syntactic structure consisting of a set of labeled links connecting pairs of words. The parser also produces a "constituent" representation of a sentence (showing noun phrases, verb phrases, etc.). With this software we were able to build a collection of sentences by tagging the sentences of the original parsed blog text collection. For efficiency reasons, these sentences were stored in a sentence-level inverted index.

4.2 Retrieval Baselines

Our ranking functions are tf-isf (sentence similarity) and BM25 (document similarity). Since tf-isf is parameter-free we only have to tune the BM25 parameters.

BM25 depends on three parameters: $k1$, which controls term frequency; b, which is a length normalization factor; and $k3$, which is related to query term frequency. We fixed $k3$ to 0 (observe that we work with short queries and, therefore, the effect of $k3$ is negligible[4]) and experimented with $k1$ and b values from 0 to 2 (both of them in steps of 0.1). The most robust parameter configuration is reported in Table 4, where we also include the performance obtained with the well-known BM25 suggested configuration ($k1 = 1.2, b = 0.75$). The configuration learnt in the training collection yields better performance than the default BM25 setting. This is somehow surprising because the default BM25 setting has proved to be very robust in many document retrieval experimentations [23]. Our optimal setting fixed $k1$ to 0.7 (instead of 1.2, which is the default value). Still, we found that performance is not very sensitive to $k1$ (the default $k1$ setting leads to quasi-optimal performance). The major difference between the default BM25 configuration and our configuration lies in the b parameter. The typical BM25 value for this parameter is 0.75 but, we observed a significant improvement in performance with parameters smaller than the default value. Specifically, we obtained the best performance with $b = 0.3$. We hypothesize that this is related to the nature of the documents. b is used as a length normalization factor. High b values increase the penalization for long documents in retrieval systems. On the other hand, smaller values apply less length normalization. In the case of BLOGS06 collection, where documents are posts and comments (usually the

[4] As a matter of fact $K3$ is barely used today[22]

main piece of text in a permalink), the distribution of lengths might be less skewed than the distribution found in standard text collection. Anyway, this requires further investigation that is out of the scope of this paper.

Table 4. BM25 results in TREC 2007 (topics 901-950) and TREC 2008 (topics 1001-1050) blog task topics. BM25 parameters were trained with 2006 (topics 851-900) topics. Statistical significance was estimated using the paired t-test (confidence levels of 95% and 99% were marked with $*$ and †, respectively)

	MAP		P@5		P@10	
	default $b = 0.75$ $k1 = 1.2$	trained $b = 0.3$ $k1 = 0.7$	default $b = 0.75$ $k1 = 1.2$	trained $b = 0.3$ $k1 = 0.7$	default $b = 0.75$ $k1 = 1.2$	trained $b = 0.3$ $k1 = 0.7$
2007	.3489	**.4017**$*$†	.6200	**.6600**	.6080	**.6440**
△%		(+15.13%)		(+6.45%)		(+5.92%)
2008	.3237	**.3812**$*$†	.6440	**.6760**	.6340	**.6640**
△%		(+17.76%)		(+4.97%)		(+4.73%)

These initial results are very promising. With a good preprocessing and parameter setting we were able to obtain a strong baseline which is competitive with TREC 2007 and 2008 blog retrieval systems [5,2]. The BM25 parameters learnt in this process ($b = 0.3$ and $K1 = 0.7$) were fixed for the subsequent experiments.

4.3 Experimental Results

In this section, we discuss the results of our combination model with the two datasets discussed above: TREC 2007 and TREC 2008.

Table 5 presents the results of our approach against BM25. The first column reports the baseline (BM25) and the rest of the columns our model with different sentence-level features: *ratio of peaks*, *medianU*, *variance* and *max*. The best value in each row is bolded. Statistical significance was estimated using the paired t-test (confidence levels of 95% and 99% were marked with $*$ and †, respectively) between each combined model and the baseline. An additional row was included for each dataset to report the values of α and β learnt in the training process.

Two features yield to improvements in performance over the blog retrieval baseline: $RatioPeaks$ and $MedianU$. $RatioPeaks$ significantly outperforms the baseline in terms of MAP and also improves consistently the rest of metrics (i.e. P@5 and P@10). Ratio of peaks is therefore the feature that performs the best. On the other hand, $MedianU$ improves consistently MAP and P@5 but is slightly less competitive in terms of P@10.

The combination weights obtained in the training process for both features help to understand the reasons behind the improvements in performance. The $RatioPeaks$ configuration has a negative weight for the feature score, meaning

Table 5. Results with TREC 2007 and TREC 2008 dataset

		$\alpha \cdot BM25_{norm} + \beta \cdot SF_{norm}$			
	baseline	RatioPeaks	MedianU	Var	Max
		TREC 2007			
(α, β)		(2.0751,-0.3484)	(2.0916,0.0642)	(1.9952,1.8750)	(1.9701,0.1716)
MAP	.4017	**.4100**$*$	.4060	.3987	.3965
△%		(+2.07%)	(+1.07%)	(-0.75%)	(-1.31%)
P@5	.6600	**.6960**	.6640	.6120	.6200
△%		(+5.45%)	(+0.60%)	(-7.84%)	(-6.45%)
P@10	.6440	**.6880**$*$	.6360	.6140	.6000
△%		(+6.83%)	(-1.26%)	(-4.89%)	(-7.33%)
		TREC 2008			
(α, β)		(1.5944,-0.2436)	(1.5345,0.1042)	(1.4578,5.0085)	(1.3233,0.4103)
MAP	.3812	.3863$*\dagger$	**.3922**	.3567	.3665
△%		(+1.34%)	(+2.89%)	(-6.87%)	(-4.01%)
P@5	.6760	.6880	**.7040**	.5120	.5680
△%		(+1.78%)	(+4.14%)	(-32.03%)	(-19.01%)
P@10	.6640	**.6900**	.6780	.5460	.5740
△%		(+3.92%)	(+2.11%)	(-21.61%)	(-15.68%)

that we promote documents with few *RatioPeaks*. Observe also that we only re-rank top-retrieved documents and, therefore, these sentence-level features are applied to documents with high query-document similarity score. By promoting top-ranked documents with few peaks we are selecting documents that have the query-document score highly concentrated in a few sentences. On the other hand, documents with many peaks have the score distributed over many places. This seems to indicate that query topics should be discussed in few concentrated places of the document.

MedianU has positive weight for the feature score, meaning that we promote documents with a high *medianU*. By promoting this kind of documents we are giving less weight to documents that contain many sentences with poor overlapping with the query (few query terms), and hence, vaguely related to the query. For example, consider two documents, D_1 and D_2, and a query that matches with two D_1 sentences and six D_2 sentences. If the number of unique query terms matched by D_1 sentences and D_2 sentences are $\{3, 4\}$ and $\{1, 1, 1, 2, 1, 1\}$, respectively [5] then: we can observe that D_2 has more matching sentences than D_1, but D_1 sentences are more highly related to the query. Hence, D_1 is likely more relevant than D_2 (in terms of *medianU* D_1 and D_2 have a score of 3.5 and 1.5 respectively).

The analysis of *RatioPeaks* and *MedianU* suggests that we should prefer documents with a few focalized (and high-quality) sentences on topic rather than documents with many (low-quality) sentences poorly related to the query. Re-

[5] Each document is represented by the number of unique terms matched by their sentences. Note that we only consider sentences that match at least one query term.

garding to the behavior of the rest of features (*var* and *max*), *max* was not able to produce any benefit. This seems to indicate that a single high-quality sentence is not enough (a relevant post should contain a few more on-topic sentences). With respect to the variance, it has shown its merits in past document-retrieval studies [9]. In these works we can observe that relevant documents have higher variance of scores than non-relevant documents, and variance of sentence scores is extremely useful to estimate relevance in topic-retrieval rankings. We believe that a future analysis of feature trends in the collection will help us to find a better way to include this feature into our models.

5 Conclusions and Future Work

In this paper, we have proposed a novel way to incorporate sentence-level features in blog topic retrieval baselines. We worked with an effective BM25 parameter configuration (which outperforms the default BM25 configuration in blog retrieval tasks) and we adapted this retrieval baseline in order to incorporate new document features based on sentence scores.

We have evaluated the effectiveness of our approach by combining four sentence-oriented features in two different datasets. We found two features (ratio of peaks and median of unique terms) that offer a good performance and are able to define combinated models that outperform state-of-the-art models for blog topic retrieval.

Our experiments have demonstrated that individual document features related to the pattern of matching between query and document sentences can outperform effective blog retrieval methods. This work is our first step to understand relevance through analysis at the sentence level. In the future, we plan to explore different methods to combine more than one sentence-level feature.

Acknowledgments

This work was partially supported by FEDER and Xunta de Galicia under projects references 2008/068 and 07SIN005206PR.

References

1. Pang, B., Lee, L.: Opinion mining and sentiment analysis. Foundations and Trends in Information Retrieval **2**(1-2) (2007) 1–135
2. Ounis, I., Macdonald, C., Soboroff, I.: Overview of the TREC 2008 blog track. In: Proc. TREC 2008, the 17th Text Retrieval Conference, Gaithersburg, United States (2008)
3. Macdonald, C., Ounis, I.: The TREC Blogs 2006 collection: Creating and analysing a blog test collection. Technical Report TR-2006-224, Department of Computing Science, University of Glasgow (2006)
4. Ounis, I., Macdonald, C., de Rijke, M., Mishne, G., Soboroff, I.: Overview of the TREC 2006 blog track. In: Proc. TREC 2006, the 15th Text Retrieval Conference. (2006)

5. Macdonald, C., Ounis, I., Soboroff, I.: Overview of the TREC 2007 blog track. In: Proc. TREC 2007, the 16th Text Retrieval Conference, Gaithersburg, United States (2007)
6. Macdonald, C., Ounis, I., Soboroff, I.: Overview of the TREC 2009 blog track. (2009)
7. Lee, Y., Na, S.H., Kim, J., Nam, S.H., Jung, H.Y., Lee, J.H.: KLE at TREC 2008 blog track: Blog post and feed retrieval. In: Proc. TREC 2008, the 17th Text Retrieval Conference, Gaithersburg, United States (2008)
8. Santos, R.L.T., He, B., Macdonald, C., Ounis, I.: Integrating proximity to subjective sentences for blog opinion retrieval. In: Proc. 31st European Conference on Information Retrieval , ECIR 2009. (2009) 325–336
9. Seo, J., Jeon, J.: High precision retrieval using relevance-flow graph. In: Proc. 32nd Annual International ACM SIGIR Conference. (2009) 694–695
10. Kaszkiel, M., Zobel, J.: Effective ranking with arbitrary passages. JASIST **52**(4) (2001) 344–364
11. Fernández, R.T., Losada, D.E.: Using opinion-based features to boost sentence retrieval. In: Proceedings of the ACM 18^{th} Conference on Information and Knowledge Management (CIKM 2009), Hong Kong, China, ACM press (November 2009)
12. Jericho, M.: Jericho HTML parser. `http://jericho.htmlparser.net/docs/index.html` (2009)
13. Parapar, J., Barreiro, A.: An effective and efficient web news extraction technique for an operational newsir system. In: XIII Conferencia de la Asociación Española para la Inteligencia Artificial CAEPIA - TTIA 2007, Salamanca, Spain, Actas Vol II (November 2007) 319–328
14. Robertson, S.E., Walker, S., Jones, S., Hancock-Beaulieu, M., Gatford, M.: Okapi at TREC-3. In: Proc. of TREC-3, the 3th Text Retrieval Conference, Gaithersburg, United States (1994)
15. Armstrong, T.G., Moffat, A., Webber, W., Zobel, J.: Has adhoc retrieval improved since 1994? In: Proc. 32nd Annual International ACM SIGIR Conference. (2009) 692–693
16. The Lemur Project. `http://www.lemurproject.org/`
17. Allan, J., Wade, C., Bolivar, A.: Retrieval and novelty detection at the sentence level. In: Proc. 26th Annual International ACM SIGIR Conference. (2003) 314–321
18. Losada, D.E.: Statistical query expansion for sentence retrieval and its effects on weak and strong queries. Information Retrieval 2010 (2010)
19. Croft, W.B.: 1. In: Combining Approaches to Information Retrieval. Kluwer Academic Publishers (2000) 1–36
20. The Link Grammar Parser. `http://www.link.cs.cmu.edu/link`
21. MorphAdorner Project. `http://morphadorner.northwestern.edu/`
22. Robertson, S.E., Zaragoza, H.: The probabilistic relevance framework: BM25 and beyond. Foundations and Trends in Information Retrieval **3**(4) (2009) 333–389
23. Robertson, S.: How okapi came to TREC. E.M. Voorhees and D.K. Harman (eds.), TREC: Experiments and Evaluation in Information Retrieval (2005) 287–299

Overcoming Spammers in Twitter – A Tale of Five Algorithms[1]

Daniel Gayo-Avello and David J. Brenes

Dept. of Computer Science, University of Oviedo, Calvo Sotelo s/n 33007 Oviedo (SPAIN),
Simplelógica, Fray Ceferino 18 1º Derecha 33001 Oviedo (SPAIN)
dani@uniovi.es,
research@davidjbrenes.info

Abstract. Micro-blogging services such as Twitter can develop into valuable sources of up-to-date information provided the spam problem is overcome. Thus, separating the most relevant users from the spammers is a highly pertinent question for which graph centrality methods can provide an answer. In this paper we examine the vulnerability of five different algorithms to linking malpractice in Twitter and propose a first step towards "desensitizing" them against such abusive behavior.

Keywords: Social networks, Twitter, spamming, graph centrality, prestige.

1 Introduction

Twitter is a service which allows users to publish short text messages (tweets) which are shown to other users following the author of the message. In case the author is not protecting his tweets, they appear in the so-called public timeline and are served as search results in response to user submitted queries. Thus, Twitter can be a source of valuable real-time information and, in fact, several major search engines are including tweets as search results.

Given that tweets are published by individual users, ranking them to find the most relevant information is a crucial matter. In fact, at the moment of this writing, Google seems to be already applying the PageRank method to rank Twitter users to that end [1]. Nevertheless, the behavior of different graph centrality methods and their vulnerabilities when confronted with the Twitter user graph, in general, and Twitter spammers in particular, are still little-known. Hence, the study described in this paper aims to shed some light on this particular issue besides providing some recommendations for future research in the area.

[1] A longer version of this study can be found at http://arxiv.org/abs/1004.0816

2 Literature Review

A social network is any interconnected system whose connections are produce of social interactions among persons or groups. Such networks can be modeled as graphs and, thus, graph theory has become inextricably related to social network analysis with a long history of research. However, for the purpose of this paper it should be enough to briefly sketch the concepts of centrality and prestige.

Both of them are commonly employed as proxy measures for the more subtle ones of importance, authority or relevance. Central actors within a social network are those which are very well connected to other actors and/or relatively close to them. There exist several measures of centrality which can be computed for both undirected and directed graphs. Prestige, in contrast, requires distinguishing inbound from outbound connections. Thus, prestige is only applicable to directed graphs which, in turn, are the most commonly used when analyzing social networks. As with centrality, there are several prestige measures such as indegree (the number of inbound connections), proximity prestige (related to the influence domain of an actor), and rank prestige, where the prestige of a node depends on the respective prestiges of the nodes linking to them. Nonetheless to say, rank prestige is mutually reinforcing and, hence, it requires a series of iterations over the whole network to be computed.

The later is the most commonly used prestige measure and there exist a number of well-known methods to compute one or another "flavor" of such a measure. In the following subsections we will briefly review the popular PageRank and HITS algorithms, in addition to lesser-known techniques such as NodeRanking, TunkRank, and TwitterRank. For the sake of brevity no equations are provided neither for PageRank nor HITS.

2.1 PageRank

PageRank [2] is one of the best known graph centrality methods. It aims to determine a numerical value for each document in the Web indicating the "relevance" of that given document. This value spreads from document to document following the hyperlinks; this way, heavily linked documents tend to have large PageRank values, and those documents receiving few links from highly relevant documents also tend to have large PageRank values. A notable property of the algorithm is that the global amount of PageRank within the graph does not change along the iterations but it is just distributed between the nodes. Thus, if the total amount of PageRank in the Web was arbitrarily fixed at 1 we could see the PageRank for a given document as a proxy for the probability of reaching that given document by following links at random (that's why PageRank is often described as a random surfer model).

2.2 Hyperlink-Induced Topic Search – HITS

HITS [3] is another algorithm to estimate the relevance of a document. The method assumes two different kinds of documents in the Web: authorities and hubs. An

authority is a heavily linked document while a hub is a document collecting links to several authorities. Web pages can exhibit both characteristics and, thus, every document has got two different scores: its authority score and its hub score. HITS was not aimed to be computed across the whole Web graph but, instead, within a query dependent subgraph; however, there is no impediment to apply it to a whole graph.

2.3 NodeRanking

NodeRanking [4] is another installment of the random surfer model. The main differences between NodeRanking and PageRank are two: (1) it is devised for weighted graphs, and (2) the damping parameter is not fixed for the whole graph but is computed for each node and depends on the outbound connections of the node.

The equations underlying this algorithm are shown below. $P_{jump}(p)$ is the equation driving the damping factor for each node: nodes with few outbound links have greater probability of being damped which should be interpreted as the random surfer getting bored because of the limited set of choices. $P_{choose}(p)$ is the probability of a page p to be chosen by the random surfer which, when ignoring weights in the edges, reduces to the same assumption in PageRank, i.e. a web surfer visiting a given page q would continue to any of the p pages linked from q with equal probability.

$$P_{jump}(p) = \frac{1}{1+|L(p)|}, P_{choose}(p) = \frac{1}{|L(q)|}, (q,p) \in E .$$

2.4 TunkRank

TunkRank [5] is one the first prestige ranking methods tailored to the particular circumstances of social networks. It lies on three assumptions: (1) each user has got a given influence which is a numerical estimator of the number of people who will read his tweets. (2) The attention a user pays to his followees is equally distributed. And (3), if X reads a tweet by Y he will retweet it with a constant probability p.

$$Influence(X) = \sum_{Y \in Followers(X)} \frac{1+p \cdot Influence(Y)}{|Following(Y)|} .$$

2.5 TwitterRank

TwitterRank [6] extends PageRank by taking into account the topical similarity between users in addition to link structure. In fact, TwitterRank is a topic-sensitive method to rank users separately for different topics. Additionally, the transition probability among users relies in both the topical similarity between users, and the number of tweets published not only by the followee, but by all the followees the follower is connected to.

These features make of TwitterRank a highly flexible method. However, we feel that it also makes it difficult to scale to the number of users and tweets that are published on a daily basis. Because of this, and for the sake of better comparison with the rest of graph centrality methods, we slightly modified TwitterRank.

The differences are the following: (1) instead of computing a different TwitterRank

value for each user and topic to be later aggregated, we aimed to compute just one TwitterRank value by user. (2) We also changed the topical similarity measure to compare users: instead of computing Jensen-Shannon Divergence between users' topic distributions we decided to apply the much more usual cosine similarity. And lastly, (3) we simplified the way to compute the damping parameter. The following equations provide a description of our implementation of TwitterRank.

TR(u) is the TwitterRank value for user u; γ is the probability of teleportation – constant for the whole graph, we used the commonly applied value of 0.15; $P(u_j,u_i)$ is the transition probability from user u_j to user u_i; $|\tau_i|$ is the number of tweets published by user u_i, and $|\tau|$ is the total number of tweets published by all the users. Lastly, $sim(u_j,u_i)$ is the cosine similarity between the tweets published by users u_i and u_j.

$$TR(u_i) = (1-\gamma)\sum_{u_j \in followers(u_i)} P(u_j,u_i)\cdot TR(u_j) + \gamma \cdot \frac{|\tau_i|}{|\tau|}$$

$$P(u_j,u_i) = \frac{|\tau_i|}{\sum_{a:u_j \boldsymbol{follows} u_a}|\tau_a|} sim(u_j,u_i)$$

3 Research Motivation

3.1 Research Questions

Social networks are increasingly gaining importance and the contents they provide can be exploited to provide up-to-date information (the so-called real-time Web). Because of the ease of publishing any content, anytime by anyone, it is ever more important to have a way to separate trustworthy sources from the untrustworthy ones.

Given the success of applying graph centrality algorithms to the Web, it seems appealing to do the same with social networks. Thus, the main research questions addressed in this study are the following: 1) How vulnerable to link spamming are common graph centrality algorithms when applied to user graphs from social networks? And 2), is it possible to "desensitize" such algorithms in a way that makes no more necessary to detect spammers but, instead, taking into account their presence and minimize their influence?

In addition to the aforementioned methods this author is proposing a variation of the PageRank method less sensitive to link abusing in social networks. Such a method relies on a de-weighting factor computed from the reciprocal links between users, and is described in the following subsection.

3.2 "Desensitizing" Prestige Ranking Methods against Link Spamming

The indegree is one of the simplest centrality measures. Translated to Twitter terms it is the total number of people following a user: the more followers a user has got the more valuable his tweets should be. Users such as Oprah Winfrey, CNN, or TIME are almost expected to have millions of followers: they are opinion-makers and mass media. One could even find reasons to explain the number of followers for Ashton

Kutcher or Britney Spears: they are celebrities. Which is harder to understand is how can spammers have far more followers than average users [7].

There is, however, a simple explanation for this phenomenon. Twitter has seen the emergence of its own etiquette and following back a new follower is considered "good manners". Once spammers took notice of this behavior, it was relatively easy to get followers: spammers just needed to massively follow other users.

Hence, the number of followers is not to be trusted and, indeed, it has been suggested that the follower-followee ratio is what really matters. In fact, such ratio can be interpreted as the user's "value" regarding the introduction of new original information from the "outside world" into the Twitter global ecosystem. Users publishing valuable tweets get followers in spite of being "impolite" (i.e. they do not follow back). That way they have huge number of followers but small numbers of followees and, thus, their ratios tend to be large. On the other hand, users who tend to discuss relatively personal matters with their close group of acquaintances do not get large audiences and, in turn, their ratios tend to be small.

How should we tackle with the spam problem, then? We think the answer lies on reciprocal connections. It seems obvious that those users with huge numbers of followers simply cannot follow-back everybody (not if they want to actually read what their followees are writing). Spammers, however, do not read tweets and, thus, they have no constrain in the number of people to follow. In other words, reciprocal links should be under suspicion and, hence, we define the follower-followee ratio with discounted reciprocity.

$$ratio\_discounted = \frac{followers - reciprocal}{followees - reciprocal}$$

However, putting under suspicion all reciprocal links seems a bit obnoxious; that's why we suggest employing either the follower-followee ratio or the discounted version depending on the possible outcome: if a user would "benefit" of using the original ratio then we use the discounted one, and vice versa. Because of that the complete name for our proposed ratio is in fact *followers to followee ratio with paradoxical discounted reciprocity*:

$$paradoxical\_discounted(p) = \begin{cases} \dfrac{followers(p)}{followees(p)} & if\ followers(p) > followees(p) \\ \dfrac{followers(p) - reciprocal(p)}{followees(p) - reciprocal(p)} & otherwise \end{cases}$$

It must be noticed that this ratio is not aimed to be directly applied to users in the graph but, instead, as a weight within an algorithm such as PageRank:

$$PR(p_i) = \sum_{p_j \in M(p_i)} \frac{PR(p_j)}{|L(p_j)|} \cdot \frac{paradoxical\_discounted(p_j)}{max\_paradoxical\_discounted}$$

Nevertheless, as we will show below we also employed this ratio to obtain a "pruned" version of the user graph. That is, we removed all those users (and their connections) with a zero ratio to then apply standard PageRank to the "pruned" graph.

4 Research Design

The main goal of this study was to compare the performance of different graph centrality algorithms when applied to social networks. To do that, a dataset, and an objective criterion against which to compare the performance of the different methods were needed. The dataset was crawled by the author from Twitter. Then, a subset of "abusive" users was obtained from that dataset. The basic idea is to compare the different algorithms by analyzing the ranking spammers reach with each of them.

4.1 Dataset Description

We relied on the Twitter search API to create the dataset for this study. To that end, we employed a query composed of frequent English stop words. That query was submitted once every minute from January 26, 2009 to August 31, 2009 collecting about 28 million tweets.

Then we obtained followers and followees for each of the 4.98 million users appearing in that collection. For the final graph we did not take into consideration links from/to users not appearing in the sample and we also dropped isolated users. Besides, a substantial amount of user accounts were suspended at the moment of the second crawl. Thus, the finally collected Twitter graph consisted of 1.8 million users and 134 million connections.

4.2 Data Preparation: A Subset of Abusive Users within Twitter

As we have already said, Twitter spammers have both more followees and followers than legitimate users. According to [7], they triple the number of both kinds of connections; these researchers argue that *"spammers invest a lot of time following other users (and hoping other users follow them back)"*.

Thus, we decided to focus on Twitter spammers and a method to detect them was needed. We implemented a version of the method described in [7] achieving similar performance: 87.32% precision versus the 91% reported by them. This spam detection system detected 9,369 spammers in the dataset. By examining a representative sample we found that about 24% of them were already suspended by Twitter.

We obtained a list of distinctive terms from spammers biographies and terms such as `business`, `money`, `internet marketing`, `social media` and `SEO` were at the top of the list. Those terms are not only popular among spammers but among other Twitter users also. As Yardi *et al.* [7] said of them: *"[They] tread a fine line between reasonable and spam-like behavior in their efforts to gather followers and attention"*. We denoted those users as "aggressive marketers", and we decided to expand the target group from pure spammers to also include them. This way we found another 22,290 users which cannot be labeled as spammers but exhibit some common behaviors with them (see Table 1).

4.3 Evaluation Method

We did not assume any prior "correct" ranking for the users, instead we consider a ranking algorithm should be judged by its ability to avoid abusive users achieving undeserved rankings. Hence, the evaluation process was quite straightforward. All of the different methods were applied to the Twitter graph to obtain a user ranking. Then, we compared the positions reached by spammers and marketers in relation to average users. The lower the rankings abusive users reach, the better the method is.

Table 1. Features characterizing the behavior of spammers, marketers, and average users.

	Spammers	**Aggressive marketers**	**Average Users**
Avg. in-degree	**3203.28**	**1338.83**	82.36
Avg. out-degree	**3156.09**	**1245.35**	82.36
Avg. # of tweets over the whole period and Std. Dev.	**41.25 (80.99)**	**12.93 (34.07)**	5.60 (19.45)
% of tweets including URLs	**90.42%**	**32.86%**	18.21%
Avg. # of URLs per tweet including URLs	1.018	1.015	1.014
% of tweets including hashtags	11.54%	8.83%	7.98%
Avg. # of hashtags per tweet including hashtags	1.41	1.42	1.50
% of retweets over total tweets	2.97%	**6.50%**	2.87%
% of "conversations" over total (excluding retweets)	6.86%	**21.48%**	**19.26%**
Avg. # of users referred in conversational tweets (excluding retweets)	1.17	1.13	1.09

Table 2. Amount of the global prestige grabbed by abusive users and top rankings reached by 90% and 50% of the users from each class under different ranking algorithms.

Ranking method	**Spammers**			**Aggressive marketers**		
	% of global prestige	Ranking of 90% of spammers	Ranking of 50% of spammers	% of global prestige	Ranking of 90% of marketers	Ranking of 50% of marketers
PageRank	1.4%	Top-60%	Top-10%	3.3%	Top-80%	Top-20%
HITS	5.2%	Top-40%	Top-10%	11%	Top-60%	Top-20%
NodeRanking	1.62%	Top-60%	Top-10%	3.86%	Top-70%	Top-20%
TunkRank	**0.74%**	**Top-70%**	**Top-20%**	**1.94%**	**Top-90%**	**Top-40%**
TwitterRank	**0.0003%**	Top-30%	Top-10%	**0.00025%**	Top-80%	**Top-40%**
Discounted PageRank	**0.22%**	N/A: 40% spammers tie for the last position	**Top-20%**	**1.05%**	N/A: 55% of the aggressive marketers tie for the last position	
Pruned PageRank	1.84%	Top-50%	Top-10%	4.27%	Top-70%	Top-20%

5 Results

About 50% of the spammers detected in the collection of tweets do not appear in the graph. Those present account for 0.25% of the users. Regarding the aggressive marketers, 98% of them appear in the graph. The acute difference from spammer to marketer presence in the graph gives an idea of the work devoted by Twitter to get rid of spammers. Hence, the whole set of spammers and marketers represent a mere 1.5% of the users. The results obtained by the different ranking methods when confronted to these abusive users are summarized in Table 2, and Figures 1 and 2.

In addition to check the ability of the different algorithms to penalize abusive users it would be interesting to also check the level of agreement between the induced rankings and their implications. Figure 3 shows the agreement between the different methods and PageRank. Such agreements were computed according to the normalized version of Kendall distance with a zero penalty parameter [8][9].

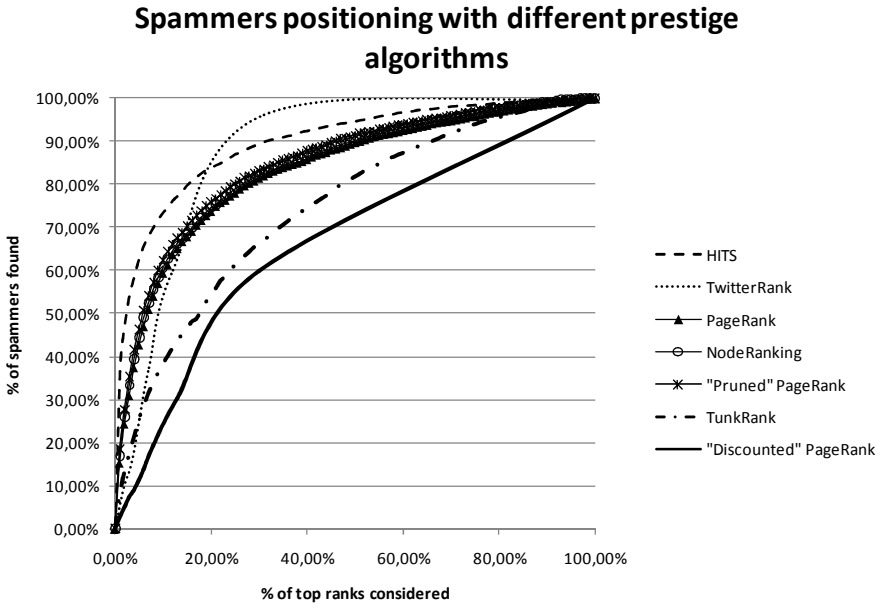


Fig. 1. Percent of spammers found for different slices of the users ranking.

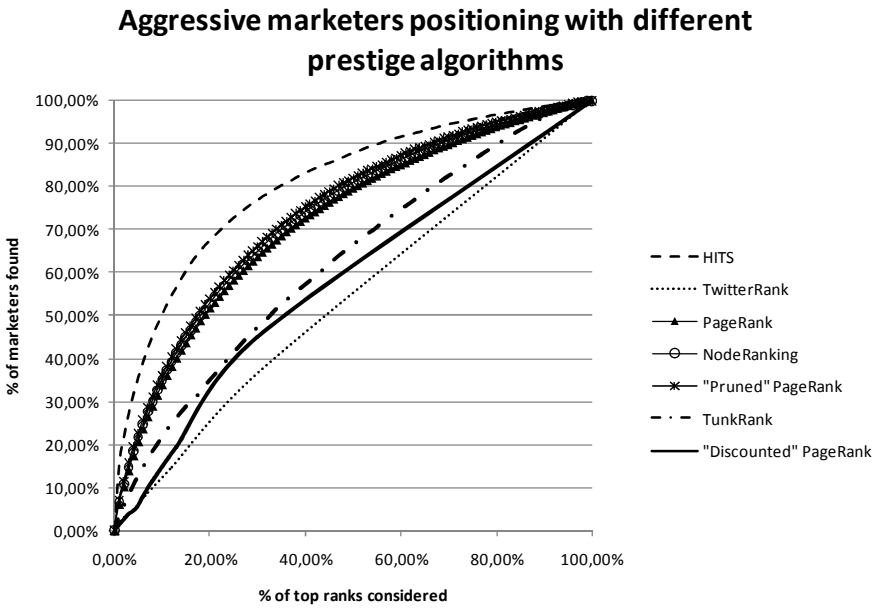


Fig. 2. Percent of aggresive marketers found for different slices of the users ranking.

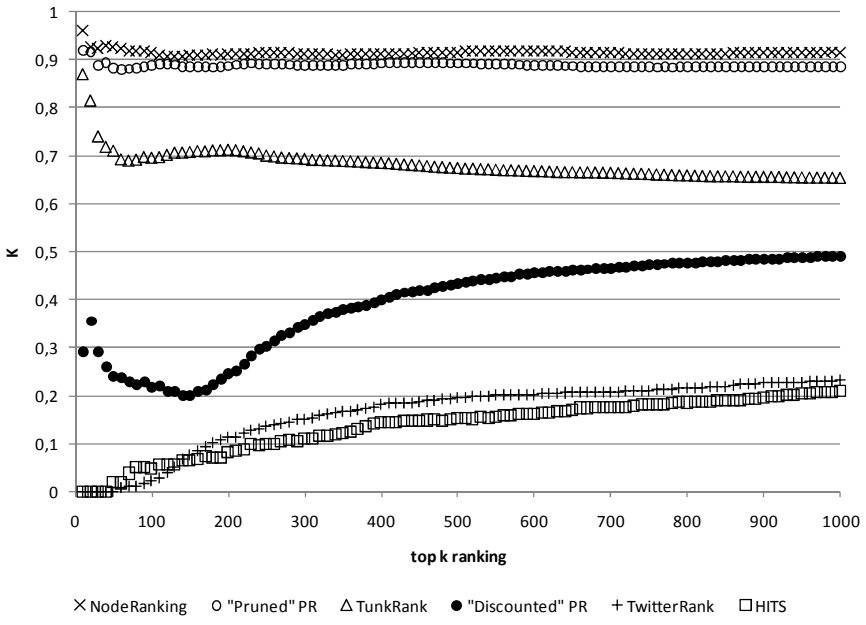


Fig. 3. Agreement between PageRank and the rest of rankings.

6 Discussion of Results

The analysis of the results obtained by PageRank when applied to the Twitter user graph supports our initial concern: users in social networks can easily game rank prestige algorithms –those described in this study. This is the most plausible explanation for spammers being much better positioned than aggressive marketers when the value of the contents provide by the former is virtually negligible.

The similarity between the results obtained by PageRank when applied to the complete graph and to the "pruned" version give support to another point of this author. Remember that the graph was "pruned" by removing those users with zero de-weighting which, in turn, was computed taking into account reciprocal links between users. One of the arguments of this author is that discounting such links is a fine way to separate users contributing value to the network from those with little or no value at all. Because the results obtained with both graphs are virtually the same we can take that as supportive of the goodness of our initial hypothesis.

There are two methods which greatly differ not only from PageRank but also from the other techniques, namely HITS and TwitterRank. Each of them exhibits different problems when applied to the Twitter graph.

HITS underperforms PageRank with respect to both spammers and marketers, and the induced ranking is very different from the other rankings. In fact, because of the very nature of HITS, this algorithm is virtually inoperative when confronted with a relatively small number of users weaving a tight network of reciprocal connections. Hub scores for users who massively follow other users tend to grow very fast and, then, those hub scores are used to compute authority scores for their followees (which are mostly spammers and are following them back). After just a few iterations those users with lots of reciprocal links earn an undeserved amount of authority. Hence, the HITS algorithm is not advisable to rank users within social networks without previously "cleaning" the graph.

Regarding the apparently contradictory results obtained by TwitterRank, they are due to the highly biased way in which it distributes prestige: the top 10 users account for 77% of the total prestige and the top 25 users for 95.5%. Virtually all of the users in the network achieve no prestige at all and, in spite of that, spammers manage to be "one-eyed kings in the land of the blind". This is pretty disappointing but, to be fair, modifying a topic-sensitive method to operate globally is, perhaps, pushing too hard the technique. Anyway, given that even the simplified version used for this research is (1) much more computationally expensive than the rest of methods surveyed, and (2) it requires much more data (namely, the tweets) to obtain the rankings, it seems not recommendable, especially when other available methods (e.g. TunkRank) are faster and provide much better results.

Lastly, there is one method clearly outperforming PageRank with respect to penalization of abusive users while still inducing plausible rankings: TunkRank. It is certainlty similar to PageRank but makes a much better job when confronted with "cheating": aggressive marketers are almost indistinguishable from common users, and spammers just manage to grab a much smaller amount of the global available prestige and reach lower positions than those achieved when using PageRank. In

addition to that, the ranking induced by TunkRank certainly agrees with that of PageRank, specially at the very top of the list, meaning that many users achieving good positions with PageRank should also get good positions with TunkRank. All of this makes TunkRank a highly recommendable ranking method to apply to social networks.

With regards to the proposal of this author, the results are not conclusive. It seems to outperform PageRank –and even TunkRank– because the amount of prestige grabbed by abusive users is smaller and their rankings lower than when applying standard PageRank. Nevertheless, it has two issues which deserve further research.

On one hand the induced ranking could be labeled as "elitist" because 70% of the users tie for the last position. One could argue that this is unsurprising given that 16% of the users from the graph have got a zero de-weighting factor; and, in fact, such results are consistent with the well-known participation inequality [10], and with a recent study revealing that 75% of the users just publish a tweet every 9 days, and 25% of the users do not tweet at all [11]. Thus, this could be considered a minor issue.

On the other hand, "discounted" PageRank exhibits a fairly distinctive curve when comparing its agreement with PageRank (see Figure 3). The agreement is much lower than, for instance, that found between PageRank and TunkRank, but the most striking behavior is the local maximum at the top positions, followed by a relatively large trough, to eventually stabilize. We found several lesser-known users at top ranks and, after studying them, we concluded that most of them have one or more "famous" followers who, in many cases, they manage to outrank. We have denoted this as the "giant shoulders" effect and it explains not only the trough at the head of the list but the smaller agreement for the rest of the ranking: many of the top users from PageRank or TunkRank are a little behind of lesser-known users they are following. This is aesthetically displeasing, at least, and the effect it can exert in the applications of the ranking is still to be explored. Nevertheless, tackling with this and the former issue is left for future research.

7 Conclusions and Future Work

This study makes four main contributions. First, graph prestige in social networks can be "gamed" by means of relationship links. The fact that spammers are always better positioned than marketers supports this assert.

Second, evaluating ranking in itself should not be the point; it should, instead, be evaluated within an objective context. Avoiding abusive users to reach undeserved rankings is a good metric to compare the performance of different algorithms.

Third, TunkRank is an obvious candidate to rank users in social networks. Although highly related to PageRank, TunkRank outperforms it with respect to penalizing abusive users while still inducing plausible rankings. In addition to that, it is simple to implement and computationally cheap –at least as cheap as PageRank.

And fourth, de-weighting the influence of a user by discounting reciprocal links seems to be a good way to separate those users contributing valuable contents to the global ecosystem from those with little to no value at all. This is supported by the fact

that when applying PageRank to both the complete version of the Twitter graph and to a "pruned" version we obtained virtually the same results.

The study opens several lines of research. First, the rankings induced by the different methods should be analyzed in other contexts, for instance, as a way to rank content providers in order to find relevant information within a social network. Second, TunkRank can for sure be manipulated and, thus, its vulnerabilities should be thoroughly studied. And third, a deeper analysis of the role of nepotistic links, in general, and the discounted ratio described in this paper, in particular, is needed.

Acknowledgements

The authors would like to thank F. Zapico and D. Guerra for their help during the Twitter dataset collection, and to M. Fernández for comments on an early draft of this paper. This work was partially financed by grant UNOV-09-RENOV-MB-2 from the University of Oviedo.

8 References

1. Talbot, D. How Google Rank Tweets, http://www.technologyreview.com/web/24353/
2. Page, L., Brin, S., Motwani, R., Winograd, T.: The PageRank Citation Ranking: Bringing Order to the Web, http://dbpubs.stanford.edu/pub/1999-66
3. Kleinberg, J.M.: Authoritative sources in a hyperlinked environment. In: Proceedings of the ninth annual ACM-SIAM symposium on Discrete algorithms, pp. 668—677 (1998)
4. Pujol, J.M., Sangüesa, R., Delgado, J.: Extracting Reputation in Multi Agent Systems by Means of Social Network Topology. In: Proceedings of the first international joint conference on Autonomous agentsand multiagent systems, pp. 467—474 (2002)
5. Tunkelang, D.: A Twitter Analog to PageRank, http://thenoisychannel.com/2009/01/13/a-twitter-analog-to-pagerank/
6. Weng, J., Lim, E.P., Jiang, J., He, Q.: TwitterRank: Finding Topic-sensitive Influential Twitterers. In: WSDM'10: Proceedings of the third ACM international conference on Web Search and Data Mining, pp. 261—270 (2010)
7. Yardi, S., Romero, D., Schoenebeck, G., boyd, d.: Detecting spam in a Twitter network. First Monday, vol. 15, no. 1—4 (2010)
8. McCown, F., Nelson, M.L.: Agreeing to Disagree: Search Engines and Their Public Interfaces. In: Proceedings of the 7th ACM/IEEE-CS joint conference on Digital Libraries, pp. 309—318 (2007)
9. Fagin, R., Kumar, R., Sivakumar, D.: Comparing top k lists. In: Proceedings of the 14th annual ACM-SIAM symposium on Discrete algorithms, pp. 28—36 (2003)
10. Nielsen, J.: Participation Inequality: Encouraging More Users to Contribute. http://www.useit.com/alertbox/participation_inequality.html
11. Heil, B., Piskorski, M.: New Twitter Research: Men Follow Men and Nobody Tweets. http://blogs.hbr.org/cs/2009/06/new_twitter_research_men_follo.html

TagR: Un Sistema de Recomendación de Etiquetas basado en Regresión Logística [*]

J. R. Quevedo, E. Montañés, J. Ranilla, and I. Díaz

Computer Science Department
University of Oviedo,
{quevedo, montaneselena, ranilla, sirene}@uniovi.es

Resumen En este trabajo se propone construir un sistema de recomendación de *tags* basado en técnicas de Aprendizaje Automático, con el propósito de ayudar a los usuarios de una red social en el etiquetado de los recursos que manejan. Para ello, el sistema aprende las etiquetas más adecuadas para etiquetar el recurso que el usuario esté tratando y se las proporciona ordenadas según la relevancia que el sistema le asigne. Como todo sistema de aprendizaje, para producir una recomendación, esto es, un conjunto de etiquetas candidatas, se alimenta de un conjunto de recursos previamente etiquetado por los usuarios de la red social. En este trabajo se estudian diferentes selecciones de conjuntos de entrenamiento, y se construye un conjunto de entrenamiento *ad hoc* para cada recurso a etiquetar. Este trabajo analiza el rendimiento de las aproximaciones resultantes de acuerdo a distintas medidas de evaluación, comprobando que el sistema de aprendizaje mejora los sistemas de recomendación tradicionales

1. Introducción

Las redes sociales, que se han situado recientemente como un mecanismo de comunicación extremadamente dinámico, representan un problema de gran dimensionalidad con contenido textual, cuyo análisis es tan difícil como provechoso. Constituyen un sistema de clasificación abierto, generado por usuarios y sin estructura jerárquica. El potencial de éstas estructuras reside en que el etiquetado de recursos es colaborativo y los usuarios no están obligados a usar una terminología controlada, sino que son libres de etiquetar con el símbolo que quiera (los denominados etiquetas -*tag*-), lo cual no está exento de riesgos, como la desestructuración de la información contenida en una determinada red, dificultando así el objetivo último de la propia red, que es buscar recursos similares, perfiles de usuario parecidos, etc. Las distintas redes sociales se diferencian por el tipo de recursos que contienen. `Flickr`, por ejemplo, permite compartir fotos, `Del.icio.us` permite compartir marcadores y `Bibsonomy` referencias bibliográficas en bibtex y marcadores simultaneamente.

[*] Este trabajo ha sido financiado por el Ministerio de Educación y Ciencia a través del proyecto [TIN2007-61273]

El etiquetado de recursos mediante *etiquetas* da lugar a lo que se conoce como una Folksonomía, que en [7] se define como una colección de recursos introducidos por usuarios en *posts*. Cada *post* es una terna formada por un recurso, un usuario y el conjunto de etiquetas que éste le asigna al recurso. Generalmente el recurso es específico del usuario que lo añadió a la red, pero el resto de usuarios pueden etiquetarlo también, con los *etiquetas* que deseen.

Por tanto, uno de los principales retos en el estudio de estas fuentes de información reside en el desarrollo de sistemas de recomendación de etiquetas, de modo que cuando un usuario colabora en una red social clasificando un determinado recurso, pueda aprovecharse de las interacciones previas que sobre ese recurso, o recursos similares, han realizado otros usuarios.

Se han propuesto diferentes aproximaciones para asistir a los usuarios en el proceso de etiquetar recursos. Algunos realizan recomendaciones analizando el contenido de los recursos [1], la cooccurrencia de *tags* [12] o mediante aproximaciones basadas en grafos [9].

Brooks et al. [3] analizan la eficiencia de los *etiquetas* para clasificar entradas de blog midiendo la similitud de todos los artículos que comparten un tag. Jäschke et al. [9] adaptan un filtrado colaborativo basado en el usuario para solventar este problema. Basile et al. [2] proponen un sistema de recomendación de *tags* capaz de aprender de la interacción previa entre usuarios así como del contenido de los recursos a anotar. Katakis et al. [10] modelan el problema como una tarea de clasificación multietiqueta. Sigurbjornsson et al. [12] caracterizan los *etiquetas* fijándose en la información que contiene el propio *tag*.

Muchos de estos sitemas requieren información asociada al propio recurso [2]. Otros simplemente sugieren un conjunto de *etiquetas* para etiquetar el recurso, pero sería interesante que además indicara cuales de ellos son los más relevantes para el recurso [10].

Este artículo propone una aproximación al problema de recomendación de *tags* basada en un proceso de aprendizaje automático. La hipótesis de partida es que un sistema de aprendizaje puede mejorar el rendimiento con respecto a los sistemas de recomendación de *tags* tradicionales. Nuestra propuesta trata de evitar los puntos débiles de métodos anteriores utilizando un sistema basado en regresión logística que produce un conjunto de *etiquetas* ordenado según la relevancia que el sistema proporcione a cada *tag* con respecto al recurso a etiquetar.

El trabajo está organizado de la siguiente forma: Nuestra aproximación se presenta en las secciones 2 y 3. Las sección 4 muestra las métricas de evaluación utilizadas. Los resultados llevados a cabo sobre colecciones de datos públicas se presentan y analizan en Section 5. Finalmente, la Sección 6 esboza algunas conclusiones y posibles retos que podremos abordar en el futuro.

2. Sistemas de Recomendación de *Tags* (TRS)

Una folksonomía se define formalmente como una terna $F := (\mathcal{U}, \mathcal{T}, \mathcal{R}, \mathcal{Y})$ con $\mathcal{U}$, $\mathcal{T}$ y $\mathcal{R}$ conjuntos finitos, cuyos elementos se llaman respectivamente usuarios,

tags y recursos, e $\mathcal{Y}$ es una relación ternaria entre ellos, i. e., $\mathcal{Y} \subseteq \mathcal{U} \times \mathcal{T} \times \mathcal{R}$, cuyos elementos son los *posts*.

Siguiendo a Marinho [11], un TRS se puede definir como un sistema que recibe una entrada, esto es, un usuario $\in \mathcal{U}$ y un recurso $r \in \mathcal{R}$ y produce un conjunto $\mathcal{T}(u, r) \subset \mathcal{T}$ de *etiquetas* como salida.

Existen algunos TRS simples pero bastante utilizados hoy en día [9] que se basan en proporcionar una lista de *tags* extraida del conjunto de *posts* relacionado con la anotación en curso:

- MPT (Most Popular *Etiquetas*): Se calcula la frecuencia con la que se usa cada *tag* t_i para anotar los recursos de la red, y se recomiendan los más usados.
- MPTR (Most Popular *Tags* by Resource): Dado un recurso r_i, se cuenta la frecuencia de coocurrencia de cada *tag* con r_i, y se recomiendan aquéllos que coocurran con más frecuencia.
- MPTU (Most Popular *Tags* by User): Dado un usuario u_i, se cuenta la frecuencia de coocurrencia de cada *tag* con u_i. Se proponen los *tags* que coocurren con más frecuencia con u_i.
- MPTRU (Most Popular *Tags* by Resource or User): Dado un recurso r_i, se cuenta la frecuencia de coocurrencia de cada *tag* con r_i o con u_i y se proponen los *tags* que coocurren con más frecuencia con r_i o u_i.

3. Aprendiendo a Recomendar

A continuación, vamos a comprobar si la introducción de un mecanismo de aprendizaje mejora la eficacia de estos TRS.

En esta sección se detalla el procedimiento propuesto en este trabajo para proporcionar un conjunto de *tags* ordenado según su relevancia. Estas recomendaciones se basan en un proceso de aprendizaje que se alimenta de lo que cada usuario ha etiquetado previamente. El núcleo del sistema es un algoritmo de aprendizaje basado en SVM, cuya salida es probabilística [4]. Lo aspectos más importantes del sistema se detallan a continuación.

- El conjunto de test no está prefijado.
- Para cada ejemplo de test seleccionado, se construye un conjunto de entrenamiento específico, luego se construye un clasificador *ad hoc* para cada ejemplo de test.
- El sistema de aprendizaje utilizado es LIBLINEAR [4]. Este sistema proporciona una distribución probabilística antes de clasificar. Esta distribución de probabilidad se utilizará para ordenar los *etiquetas*, tomando como *tag* más relevante para el recurso a etiquetar aquél cuya proabilidad sea más alta.
- Los *tags* recomendados serán todos los que aparecen en las categorías del conjunto de entrenamiento, lo que conlleva que algunos de los *tags* del conjunto de test que sean relevantes, no se considerarán.

3.1. Definición del Conjunto de Test

Muchos trabajos construyen sus métodos siguiendo la división tradicional en conjunto de entrenamiento y test. De esta forma, aprenden a partir de un conjunto de entrenamiento prefijado y recomiendan un conjunto de *tags* para cada *post* del conjunto de test [10]. La aproximación adoptada en ente trabajo es bastante diferente en el sentido de que para cada *post* del conjunto de test, se construye un conjunto de entrenamiento específico.

Una folksonomía está compuesta por un conjunto de *posts*. Cada *post* está formado por un usuario, un recurso y un conjunto de *etiquetas*, es decir

$$p_i = (u_i, r_i, \{t_{i_1}, \ldots, t_{i_k}\})$$

Cada *post* de test p_i se transforma en tantos ejemplos de test como etiquetas tenga asignadas de la siguiente forma

$$e_1 = (u_i, r_i, t_{i_1}) \ldots e_k = (u_i, r_i, t_{i_k}) \tag{1}$$

Ejemplo 1 Supongamos la siguiente folksonomía

post	$fecha$	$Usuario$	$Recurso$	$Tags$
p_1	d_1	u_1	r_1	t_1
p_2	d_2	u_1	r_2	t_2
p_3	d_3	u_2	r_1	t_1
p_4	d_4	u_3	r_1	t_3
p_5	d_5	u_2	r_2	t_4
p_6	d_6	u_2	r_1	t_2, t_3
p_7	d_7	u_2	r_2	t_2, t_5
p_8	d_8	u_3	r_2	t_1

(2)

y sea $p_7 = (u_2, r_2, \{t_2, t_5\})$ un *post* de test seleccionado aleatoriamente en el instante d_7. Entonces el conjunto de test es el siguiente

ejemplo	$fecha$	$Usuario$	$Recurso$	Tag
e_1	d_7	u_2	r_2	t_2
e_2	d_7	u_2	r_2	t_5

(3)

3.2. Definición del Conjunto de Entrenamiento

Cualquier sistema de aprendizaje depende fuertemente del conjunto de entrenamiento que se utilice para aprender, por lo que la selección de un conjunto de entrenamiento adecuado es fundamental para el éxito del aprendizaje. Por esa razón se selecciona un conjunto de entrenamiento específico para cada conjunto de test que se quiera etiquetar. En este trabajo se selecciona un conjunto de entrenamiento de forma dinámica a partir de los N *posts* añadidos antes que el *post* de test. El parámetro N se fijará experimentalmente.

Esta selección del conjunto de entrenamiento hace que se propongan los *tags* más *modernos* de la folksonomía además de producir un sistema más escalable, ya que el número de *posts* del conjunto de entrenamiento no crece según se vayan añadiendo *posts* al sistema.

A continuación se establecen las condiciones para que un ejemplo se incluya en el conjunto de entrenamiento. Distinguiremos 4 aproximaciones distintas, 3 de ellas se definen a continuación.

Aproximación 1. TR Sea $p_i = (u_i, r_i, \{t_{i_1}, \ldots, t_{i_k}\})$ un *post* de test. Sea $\mathcal{R}_{u_i}$ el subconjunto de *posts* asociados al recurso r_i y

$$\mathcal{R}^t_{r_i} = \{p_i / p_i \in \mathcal{R}_i \textit{ añadido antes del tiempo } t\}.$$

La aproximación TR selecciona como conjunto de entrenamiento los N *posts* más actuales de $\mathcal{R}^{d_i}_{r_i}$, siendo d_i la fecha en la que p_i se añadió a la folksonomía.

Aproximación 2. TU Sea $p_i = (u_i, r_i, \{t_{i_1}, \ldots, t_{i_k}\})$ un *post* de test. Sea $\mathcal{P}_{u_i}$ la personomía (el conjunto de *posts* añadido pos un usuario) asociada al usuario u_i y

$$\mathcal{P}^t_{u_i} = \{p_i / p_i \in \mathcal{P}_{u_i} \textit{ añadido antes del tiempo } t\}$$

La aproximación TR selecciona como conjunto de entrenamiento los N *posts* más actuales de $\mathcal{P}^{d_i}_{u_i}$, siendo d_i la fecha en la que p_i se añadió a la folksonomía.

Aproximación 3. TRU Los conjuntos de entrenamiento que se obtienen siguiendo alguna de las aproximaciones anteriores, no tienen en cuenta el hecho de que el sistema de aprendizaje se aplica cuando ya se ha establecido una relación entre el usuario y el recurso. Por tanto puede ser interesante añadir como ejemplos de entrenamiento los concernientes tanto al recurso como al usuario del *post* de test

El conjunto de entrenamiento asociado a p_i será

$$UR^{d_i}_{u_i,r_i} = \{\mathcal{P}^d_i \cup \mathcal{R}^d_i\} \backslash \{p_j / p_j = (u_i, r_i, \{t_1, ..., t_n\})\}$$

Ejemplo 2 Consideremos $\mathcal{R}^{d_7}_{r_2} = \{p_2, p_5\}$ y $\mathcal{P}^{d_7}_{u_2} = \{p_3, p_5, p_6\}$. En este caso el conjunto de entrenamiento es

$$UR^{d_7}_{u_2,r_2} = \{\mathcal{P}^{d_7}_{u_2} \cup \mathcal{R}^{d_7}_{r_2}\} \backslash \{p_j / p_j = (u_i, r_i, \{t_1, ..., t_n\})\} =$$

$$\{\{p_3, p_5, p_6\} \cup \{p_2, p_5\}\} \backslash \{p_5\} = \{p_2, p_3, p_6\}$$

Entonces, el conjunto de entrenamiento se define como sigue:

ejemplo	*fecha*	*Usuario*	*Recurso*	*Tags*
e_2	d_2	u_1	r_2	t_2
e_3	d_3	u_2	r_1	t_1
e_{6_1}	d_6	u_2	r_1	t_2
e_{6_2}	d_6	u_2	r_1	t_3

(4)

3.3. Representación de los Ejemplos

Una vez que se ha seleccionado el conjunto que se cree más adecuado para realizar el entrenamiento, es preciso establecer las características que van a determinar los ejemplos de entrenamiento y test así como la clase de cada examen. Consideramos como atributos de los ejemplos los *tags* que se han utilizado previamente para etiquetar el recurso en la folksonomía. De esta forma, cada ejemplo se representa por un vector booleano V de tamaño M (que es el número de *tags* de la folksonomía), de modo que $v_j = 1$ si y sólo si t_j se ha utilizado para etiquetar el recurso con anterioridad y 0 en otro caso. La clase del ejemplo será el *tag* con el que el usuario ha etiquetado el recurso en un cierto momento.

Por otra parte, es posible que eliminar atributos no relevantes o ruidosos pueda resultar provechoso para mejorar tanto la eficacia como la eficiencia del sistema. Además, esta representación de los ejemplos permite realizar una selección de atributos sencilla, que consiste en considerar sólo los atributos que representen el test. Esta aproximación puede llevarse a cabo, en primer lugar, porque construimos un conjunto de entrenamiento para cada *post* de test, y, en segundo lugar, porque los pesos de los atributos que no representan el *post* de test no contribuyen a la puntuación de éste post, y, por tanto, pueden ser considerados irrelevantes a priori. Esta aproximación afecta no a la selección de los ejemplos en sí misma, sino a la representación de dichos ejemplos. Cuando la aplicamos a un conjunto de entrenamiento obtenido siguiendo la aproximación TRU, la denominaremos TRUTR.

Ejemplo 3 Consideremos el ejemplo de test y el conjunto de entrenamiento obtenido en el ejemplo anterior. Los atributos del *post* de test son t_2 y t_4. Por tanto, el conjunto de entrenamiento se representará utilizando a los sumo estas dos etiquetas. Inicialmente se representaba del siguiente modo :

$$\begin{array}{ccccc} ejemplo & fecha & recurso & atributo & categoria \\ e_2 & d_2 & r_2 & \emptyset & t_2 \\ e_3 & d_3 & r_1 & t_1 & t_1 \\ e_{6_1} & d_6 & r_1 & t_1, t_3 & t_2 \\ e_{6_2} & d_6 & r_1 & t_1, t_2, t_3 & t_3 \end{array} \quad (5)$$

Pero eliminando de la representación las etiquetas que no representaban el *post* de test, obtenemos la nueva representación

$$\begin{array}{ccccc} ejemplo & fecha & recurso & atributos & categoria \\ e_2 & d_2 & r_2 & \emptyset & t_2 \\ e_3 & d_3 & r_1 & \emptyset & t_1 \\ e_{6_1} & d_6 & r_1 & \emptyset & t_2 \\ e_{6_2} & d_6 & r_1 & t_2 & t_3 \end{array} \quad (6)$$

3.4. Sistema de Aprendizaje

Dado que uno de los objetivos que nos hemos planteado, consiste en proporcionar un conjunto de etiquetas que se adpaten tanto al usuario que las va a utilizar como al recurso que se va a etiquetar, sería beneficioso que el sistema de aprendizaje puntuara las etiquetas, de modo que el usuario supiera, de entre todos las etiquetas que se les proporcionan, aquéllas que se adaptan mejor al recurso que quiere etiquetar. Por esta razón buscamos un sistema que nos proporcione una ordenación total del conjunto de etiquetas que proporcione como salida.

Inicialmente se pensó en utilizar un sistema multicategoría, dado que el problema lo es. Sin embargo estos sistemas no proporcionan una relación de orden total. Se pueden adaptar para proporcionar un orden parcial, diferenciando entre las etiquetas que proporcione el sistema y las que no, pero no es lo que se buscaba.

El orden total que estamos buscando se puede conseguir utilizando un sistema multi-etiqueta, ya que muchos de estos sistemas ordenan antes de resolver el problema de clasificación multietiqueta ([13]). Elisseeff and Weston [5] proponen un sistema multi-etiqueta basado en SVM, que establece un orden total en las categorías, pero la complejidad de este sistema es cuadrática después de muchas optimizaciones, lo que hace que el método no se pueda llevar a la práctica sobre conjuntos de datos reales.

Con respecto al problema de recomendación de *etiquetas*, Godbole y Sarawagi en [6] evolucionan SVM simplemente extendiendo el conjunto de datos original, resolviendo un problema multicategoría.

La aproximación seleccionada aquí está basada en la librería LIBLINEAR [4] (disponible en [1]) basada en SVM, y que representa una nueva alternativa para resolver problemas multi-etiqueta utilizando regresión logística. LIBLINEAR obtiene la distribución de probabilidad que siguen las categorías antes de clasificar, lo que utilizaremos para ordenar dichas categorías (en nuestro caso, etiquetas), tomando como mejor aquella cuya probabilidad sea más alta. Para realizar la experimentación utilizaremos la configuración por defecto de LIBLINEAR después de modificar ligeramente la salida.

La evaluación se realiza cuando un usuario desea etiquetar un recurso. En este caso, todas las etiquetas que aparezcan como categoría del conjunto de entrenamiento se ordenan utilizando LIBLINEAR, salvo las que ya hayan sido asigandas por ese usuario a ese recurso. Si el recurso no ha sido etiquetado previamente, se recomiendan las etiquetas que haya utilizado el usuario previamente ordenadoas según su frecuencia de uso

4. Evaluación del Método

No existe consenso respecto a qué métrica es la adecuada para evaluar un TRS [9]. Algunos trabajos no incluyen una evaluación cuantitativa de sus métodos [14]. En [11] y [9] seleccionan un *post* aleatoriamente para cada usuario y

[1] http://www.csie.ntu.edu.tw/∼cjlin/liblinear/

proporcionan un conjunto de *tags* para él utilizando toda la folksonomía salvo ese post, y, a partir de ahí, calculan la precisión, la cobertura y la F_1 (([10])) Sin embargo, esta forma de evaluar no nos es útil puesto que proporciona la misma valoración para todos los conjuntos de *tags* que difieren en el orden de los elementos que lo componen.

Obviamente, un TRS que retorne las etiquetas positivas en la parte alta de la jerarquía es preferible que uno que las devuelva al final. Por eso, se hace necesario establecer una métrica que nos permita cuantificar tanto las etiquetas que se recomiendan, como el orden en el que se recomiendan.

La métrica NDCG [8] calcula la ganancia acumulada que obtiene un usuario examinando los resultados de la recuperación hasta una determinada posición. Además aplica un factor de descuento sobre la relevancia para devaluar los documentos recuperados en último lugar y calcula una puntuación relativa con respecto a la ganancia acumulada ideal.

Sin embargo, ninguna de estas métricas se adapta bien al problema de la evaluación de los sistemas de recomendación de *tags* dado que las ordenaciones de eqtiquetas a comparar pueden no contener las mismas etiquetas, y todas las métricas anteriores están pensadas para evaluar permutaciones de un mismo conjunto.

Por esta razón en este trabajo se propone una alternativa que trata de evitar estos inconvenientes. Consiste en calcular la medida F_1 para todos los posibles cortes de la ordenación que retornan un determinado *tag* y elegir la más alta. Se denotará F_1^+ y se define como

$$F_1^+ = \max_{0<=i<=r} (F_1)_i \tag{7}$$

donde r representa el tamaño del conjunto (el número de *tags* que retorna el sistema), $(F_1)_i$ es la F_1 de la clasificación, asumiendo que el sistema ha clasificado los primeros i *tags* como positivos y el resto como negativos. Este índice varía entre 0 (el sistema no devolvió ningún *tag* como positivo) y r (los devolvió todos).

Dado que un *tag* negativo no mejora el rendimiento del sistema, sólo se calcula F_1^+ donde se añaden tags positivos. Nótese que el valor de F_1^+ no varía si se añaden *tags* negativos al final del ranking, como le ocurre a AP. Si embargo, tiene en cuenta todos los *tags* positivos en vez de considerar los *tags* positivos que aparecen en la parte alta del ranking, como hace AUC.

5. Experimentos

5.1. Colección

Los experimentos se han llevado a cabo utilizando la colección `bt08`, que es un conjunto de datos formado por *posts* bibtex, recopilados por el ECML PKDD Dicovery Challenge 2008 [2] y extraídos de Bibsonomy [3].

[2] http://www.kde.cs.uni-kassel.de/ws/rsdc08/
[3] http://www.bibsonomy.org

Antes de utilizar estos conjuntos de datos, se ha realizado una limpieza de dichos conjuntos siguiendo las pautas del PKDD Discovery Challenge 2008. El preprocesamiento incluye la eliminación de etiquetas inútiles (e.g., system:: unfiled), la transformación de todas las letras a minúsculas y la eliminación de caracteres no alfanuméricos. El número de usuarios, *etiquetas*, recursos y *posst* se muestran en el Cuadro 1.

Dataset	users	*tags*	resources	*posts*
`bt08`	1,206	29,739	96,616	278,008

Cuadro 1. Datos estadísticos de la colección de prueba

5.2. Discusión de los Resultados

En esta sección se detallan los experimentos llevados a cabo. Para comprobar los métodos, se han seleccionado aleatoriamente 1000 *posts* de test, para cada uno de los cuales se ha seleccionado un conjunto de entrenamiento *ad hoc* que depende de

- Como se haya construido el conjunto de entrenamiento
- La cardinalidad del conjunto de entrenamiento (N): $N = i * 500$ con $i = 1, 2, \ldots, 10$. De este modo, se analizará la influencia del tamaño del conjunto de entrenamiento

El Cuadro 2 muestra el comportamiento de los TRSs de referencia según las métricas detalladas en la Sección4. Parece que los sitemas `MPTU` y `MPTRU` son considerablemente mejores que el resto de los TRS de referencia para ambos conjuntos de datos.

	F_1^+	*AUC*	AP	NDCG
`MPT`	6.7 %	5.8 %	4.6 %	6.6 %
`MPTR`	7.8 %	5.2 %	5.7 %	7.7 %
`MPTU`	37.2 %	56.3 %	28.8 %	42.5 %
`MPTRU`	38.2 %	57.9 %	29.7 %	43.7 %

Cuadro 2. Rendimiento de los TRSs de referencia para bt08

La Figura 1 muestra la F_1^+, AUC, AP y NDCG de cada conjunto de entrenamiento obtenido cuando el número de ejemplos de entrenamiento varía entre 500 y 5000. Claramente, considerar como ejemplos de entrenamiento los que el usuario del *post* de test ha posteado previamente, aumenta sensiblemente la eficacia del sistema. Si además añadimos al conjunto de entrenamiento los *posts* en

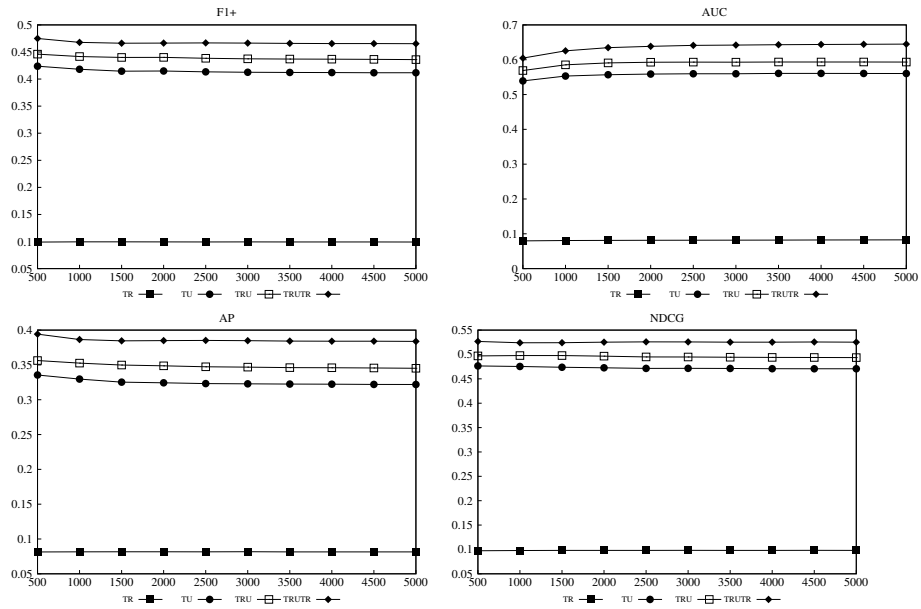


Figura 1. F_1^+, AUC, AP and NDCG de la colección bt08

los que ha estado involucrado previamente el recurso del *post* de test, también mejoramos la eficacia del método. Es más, si utilizamos para representar los ejemplos de entrenamiento, sólo las etiquetas que se utilizan para representar el *post* de test, también mejoramos la eficacia del TRS.

Aunque todas las medidas de evaluación coinciden en valorar mejor a las mismas aproximaciones, el comportamiento de éstas con respecto al número de *post* de entrenamiento difiere de una aproximación a otra.

6. Conclusiones y Trabajos Futuros

Este trabajo propone un TRS que aprende a ordenar un conjunto de *tags* candidatos a etiquetar un recurso añadido por un usuario. Este sistema se alimenta de distintos conjuntos de *posts*, dependiendo de la aproximación utilizada.

Además, se ha propuesto una medida de evaluación nueva, llamada F_1^+, que tiene en cuenta las posiciones que ocupan los *tags*, evitando alguno de los problemas de las otras medidas de evaluación utilizadas.

El TRSs propuesto se compara con el mejor de los TRS de referencia (`MPTRU`). Los resultados muestran una mejora significativa de nuestros TRS con respecto a `MPTRU`, siendo (`TRUTR`) el mejor de nuestras 4 versiones.

Por otro lado, la cardinalidad del conjunto de entrenamiento ha variado entre 500 y 5000. Los resultados muestran que el número de ejemplos apenas afecta a la eficacia del sistema. En cualquier caso, lo que sí podemos asegurar es que es

posible mantener la eficacia del sistema sin añadir una gran cantidad de ejemplos de entrenamiento, lo que ralentizaría el proceso.

Por tanto, podemos concluir que la introducción de un sistema de aprendizaje mejora la eficacia de los sistemas de aprendizaje.

Sin embargo este método puede ser mejorado analizando en profundidad la representación de los ejemplos realizando agrupamientos de *tags* a priori, lo que reduciría sensiblemenente la complejidad del mismo. También sería interesante analizar el contenido de los recursos, lo que sin duda ayudaría a a establecer relaciones entre recursos.

Referencias

1. B. Adrian, L. Sauermann, and T. Roth-Berghofer. Contag: A semantic tag recommendation system. In *Proceedings of I-Semantics' 07*, pages 297–304, 2007.
2. P. Basile, D. Gendarmi, F. Lanubile, and G. Semeraro. Recommending smart tags in a social bookmarking system. In *Bridging the Gep between Semantic Web and Web 2.0 (SemNet 2007)*, pages 22–29, 2007.
3. Christopher H. Brooks and Nancy Montanez. Improved annotation of the blogosphere via autotagging and hierarchical clustering. In *WWW '06: Proceedings of the 15th international conference on World Wide Web*, pages 625–632, New York, NY, USA, 2006. ACM.
4. S.S. Keerthi C. J. Lin, R. C. Weng. Trust region newton method for logistic regression. *Journal of Machine Learning Research*, (9):627–650, 2008.
5. Andre Elisseeff and Jason Weston. A kernel method for multi-labelled classification. In *In Advances in Neural Information Processing Systems 14*, pages 681–687. MIT Press, 2001.
6. Shantanu Godbole and Sunita Sarawagi. Discriminative methods for multi-labeled classification. In Honghua Dai, Ramakrishnan Srikant, and Chengqi Zhang, editors, *PAKDD*, volume 3056 of *Lecture Notes in Computer Science*, pages 22–30. Springer, 2004.
7. A. Hotho, R. J.ªschke, C. Schmitz, and G. Stumme. Trend detection in folksonomies. pages 56–70. 2006.
8. K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446, 2002.
9. R. Jäschke, L. Marinho, A. Hotho, L. Schmidt-Thieme, and G. Stumme. Tag recommendations in folksonomies. In Alexander Hinneburg, editor, *Proceedings of LWA 2007*, pages 13–20, sep 2007.
10. I. Katakis, G. Tsoumakas, and I. Vlahavas. Multilabel text classification for automated tag suggestion. In *Proceedings of ECML PKDD Discovery Challenge (RSDC08)*, pages 75–83, 2008.
11. L. Marinho and L. Schmidt-Thieme. Collaborative tag recommendations. In Springer Berlin Heidelberg, editor, *Studies in Classification, Data Analysis, and Knowledge Organization*, pages 533–540, 2008.
12. B. Sigurbjörnsson and R. van Zwol. Flickr tag recommendation based on collective knowledge. In *Proceedings of WWW '08*, pages 327–336, New York, 2008. ACM.
13. G. Tsoumakas and I. Katakis. Multi label classification: An overview. *International Journal of Data Warehousing and Mining*, 3(3):1–13, 2007.
14. Z. Xu, Y. Fu, J. Mao, and D. Su. Towards the semantic web: Collaborative tag suggestions. In *Proceedings of WWW2006*, Edinburgh, Scotland, 2006.

La Temporalidad en los Sistemas de Recomendación: Una revisión actualizada de propuestas teóricas

Pedro G. Campos[1,2] and Fernando Díez[1]

[1] Escuela Politécnica Superior, Universidad Autónoma de Madrid, 28049, Madrid, España
[2] Departamento de Sistemas de Información, Universidad del Bío-Bío, Avda. Collao 1202, Concepción, Chile
pgcampos@ubiobio.cl, fernando.diez@uam.es

Resumen Recientemente se ha observado que la dimensión temporal juega un rol preponderante en la calidad de las predicciones realizadas por sistemas de recomendación, cómo muestran diversos estudios. Existen diversas propuestas en la literatura que incorporan esta información para mejorar las recomendaciones obtenidas mediante filtrado colaborativo. Este trabajo revisa algunas de las principales contribuciones realizadas en esta área, aportando una visión general respecto de las decisiones de diseño implicadas en su implementación. Adicionalmente, se realiza una propuesta inicial de elementos necesarios para llegar a establecer un marco de trabajo en esta dimensión.

1. Introducción

El desarrollo de mejores algoritmos de recomendación es actualmente un activo campo de investigación y desarrollo. Además del interés científico, existe un importante interés por parte de la industria en este ámbito. Un ejemplo de ello es el *Netflix Prize* [1], que recientemente pagó US$1.000.000 a los desarrolladores de un algoritmo que mejoró sustancialmente el rendimiento del sistema de recomendación (SR) de la empresa de alquiler y venta de vídeos en DVD *Netflix*. La búsqueda de mejoras ha llevado al estudio de diversos aspectos, destacándose la temporalidad de la información [2], que implica reconocer que los gustos de los usuarios cambian en el tiempo. El propósito de este artículo es describir algunas de las propuestas más importantes existentes al día de hoy sobre cómo abordar el aspecto temporal y analizar las decisiones de diseño más relevantes en su implementación, de manera que pueda servir como punto de partida tanto a nuevos investigadores y estudiantes del área interesados en esta importante perspectiva como a desarrolladores de SR interesados en incorporar esta dimensión.

En este trabajo nos hemos centrado en propuestas que se basan en Filtrado Colaborativo (FC)[3,4], por un doble motivo: además de ser muy populares, son las que permiten incorporar mayor cantidad de información temporal, y por lo tanto es el área en la que existe mayor desarrollo del tema.

El resto del artículo se organiza de la siguiente forma: La sección 2 describe de manera general diversas propuestas para considerar la variación en el tiempo existentes en la literatura. La sección 3 profundiza en algunas de las propuestas más relevantes, detallando diseños alternativos que pueden ser utilizados para implementar sistemas de recomendación que aborden el aspecto temporal. La sección 4 propone los elementos iniciales para construir un marco de trabajo sobre temporalidad en SR. Finalmente, la sección 5 presenta algunas conclusiones.

2. Temporalidad en la literatura de Sistemas de Recomendación

2.1. Primeras aproximaciones

El aspecto temporal hasta hace poco tiempo había sido considerado de forma limitada en el ámbito de SR. Uno de los primeros trabajos que menciona este aspecto es la propuesta del 2001 de Adomavicius y Tuzhilin, aunque dicho trabajo se enfoca en la posibilidad de extender el modelo usuario-objeto para realizar recomendaciones multidimensionales [5]. Terveen et al. en el 2002 utilizaron la historia personal para determinar las preferencias actuales de usuarios con respecto a música, en un sistema visual interactivo llamado *HistView* [6]. Sin embargo, uno de los primeros trabajos centrados en el aspecto temporal es el de Tang et al. en el 2003, en el que se utiliza el año de producción de películas para reducir la carga de un SR de FC para recomendar películas por medio de un "truncamiento" de datos, eliminando información de películas consideradas "antiguas" [7]. En dicho trabajo se obtienen predicciones más precisas al utilizar este enfoque. Aunque el tratamiento de la temporalidad es bastante limitado (no se consideran muchos de los aspectos que influyen, por ej. el momento en que se realiza la valoración), los resultados muestran la importancia que puede tener la temporalidad. Otro trabajo relacionado es de Sugiyama et al. quienes en el 2004 exploran un tipo de FC temporal en el que realizan un análisis detallado de la historia de navegación (en internet) durante un día [8].

2.2. Mayor peso a información reciente

En el 2005, Ding y Li son de los primeros autores en utilizar un enfoque de pesos en función del tiempo dando mayor peso a valoraciones recientes [9], con el cual en el 2006 se desarrolla un algoritmo de FC basado en información reciente (*recency-based* FC)[10].

En el 2008, Lee et al. proponen un algoritmo de FC basado en información implícita, con un modelo que se puede considerar más general de información temporal, ya que incluye dos dimensiones: el momento de lanzamiento del objeto en el sistema (momento en que el objeto se añade al catálogo del sistema) y el momento en que el usuario realiza la valoración, asignando pesos decrecientes según la antigüedad en estas dimensiones. A partir de la información temporal se ponderan las valoraciones realizadas por los usuarios, otorgando un mayor peso

tanto a las valoraciones recientes como a los lanzamientos recientes. La asignación de pesos sigue un esquema muy sencillo (ver sección 3.1). Sus resultados muestran, con respecto a FC puro, mejoras de hasta 47 % para FC basado en usuario y hasta 25 % para FC basado en objeto [11]. Es destacable de este trabajo el hecho que se basa en información implícita, ya que las valoraciones consideradas consisten sólo en la preferencia (o no) manifestada en la compra (o no) de un producto. Posteriormente, los mismos autores realizan un análisis más elaborado, probando con diferentes esquemas de asignación de pesos de acuerdo a la información temporal [12]. Sus resultados muestran que considerando cualquiera de las dos dimensiones por separado mejora la precisión de las predicciones, y que al considerar las dos dimensiones en conjunto los resultados mejoran más aún, aunque no aditivamente.

Si bien estos resultados fueron prometedores, al igual que en el caso de "truncamiento" de valoraciones, el enfoque de pesos en función del tiempo es limitado, puesto que se pierde (o subvalora) información que podría ser valiosa [2].

2.3. Influencia del tiempo en algoritmos *kNN*

Otra forma de considerar la temporalidad la proporciona Neal Lathia y sus colaboradores [13,14]. En el 2008 analizaron el comportamiento de algoritmos *kNN* desde una perspectiva temporal, considerando al algoritmo como un proceso que genera una red social implícita en que los nodos corresponden a usuarios y los arcos a relaciones entre los usuarios [13]. Entre los hallazgos interesantes de este trabajo, están el que las vecindades formadas por *kNN* son dinámicas, es decir, varían en el tiempo, aunque convergen finalmente hacia un conjunto estable. La velocidad de convergencia es dependiente de la forma de medir la similitud entre usuarios. Además, detectaron lo que llamaron "usuarios poderosos" (*power users*), que son usuarios frecuentemente seleccionados como vecinos por otros muchos usuarios, por lo que tienen una alta influencia en las predicciones realizadas para esos otros usuarios. En el 2009 plantean que los sistemas de recomendación en producción deben ajustar continuamente sus parámetros a medida que agregan nueva información al *dataset* [14]. Sin embargo, observan que al aplicar *kNN* sobre un *dataset* iterativamente (simulando actualizaciones de los datos cada cierto tiempo), no se obtienen resultados igualmente buenos que al aplicarlos sobre un dataset estático. A partir de estas observaciones, se propone un método de actualización de tamaños de vecindarios para cada usuario, que de acuerdo a los experimentos que muestran, obtiene mejores resultados que *kNN*. Posteriormente realizan un análisis crítico respecto de la forma en que se evalúan los SR hoy en día, centrado sólo en la precisión (accuracy)[3], y se plantea una forma de evaluar el FC a través del tiempo, a través de la métrica que denominan *Time-Averaged RMSE* (RMSE ponderado en el tiempo), la cual corresponde al RMSE entre las predicciones realizadas y valoraciones obtenidas hasta un instante de tiempo determinado (en lugar de calcularse sobre todo el *dataset*, ver sección 4)

[3] Diversos autores han criticado esto, como Adomavicius y Tuzhilin[15] y Herlocker et al.[16]

[17]. Al aplicar esta métrica se observan diferencias en los resultados de precisión a través del tiempo. Adicionalmente, se plantea la medición de la diversidad entre dos listas de recomendación generadas en instantes distintos, utilizando para ello la medida de similitud de Jaccard. Los resultados de su aplicación muestran que los métodos más precisos, llevan a una menor diversidad.

2.4. Incorporación de Parámetros temporales en el modelo

Una última familia de propuestas para considerar la temporalidad corresponde a la incorporación de parámetros específicos en los modelos que den cuenta de las variaciones en el tiempo tanto en los gustos de los usuarios, como en las tendencias de los objetos. La incorporación de parametros que dan cuenta de efectos globales en las valoraciones tales como tendencias generales de los usuarios o los objetos (por ejemplo, usuarios que usualmente valoran más altamente los objetos, u objetos que tienden a recibir valoraciones bajo el promedio) [18,19,20] dio origen a esta línea de investigación. Del mismo modo que para considerar estos efectos estáticos, se incorporan parámetros específicos para dar cuenta de los efectos dinámicos (temporales). Xiang y Yang (2009) consideran cuatro tipos principales de efectos temporales[21]: i) tendencia (*bias*) temporal, que corresponde al cambio en los gustos de la sociedad en el tiempo (por ejemplo, cambios en la moda); ii) cambio en la tendencia de usuario, ya que un usuario puede cambiar sus hábitos de valoración en el tiempo (por ejemplo, puede ser más exigente a medida que tiene mayor experiencia); iii) cambio en la tendencia de un objeto, ya que la popularidad de los objetos cambia en el tiempo (por ejemplo, un objeto particular puede estar asociado a una moda pasajera); y iv) cambios en las preferencias de usuario, ya que un usuario puede cambiar su actitud respecto a algunos tipos de objetos (por ejemplo, por una experiencia desagradable). Se mencionan otros tipos de efectos temporales, que tendrían un menor impacto. Entre estos últimos están las posibles tendencias asociadas al momento de la recomendación (estación, mes del año, día de la semana, etc.), diferencias entre usuarios nuevos y antiguos, y diferencias entre objetos nuevos y antiguos. La mayoría de las propuestas en este sentido se basan en la construcción de un modelo a partir de los datos (usualmente con técnicas como factorización de matrices), en el cual se agregan parámetros adicionales que dan cuenta de cada uno de los tipos de efectos temporales que pueden influir en las tendencias enumeradas anteriomente. Los resultados obtenidos con estos modelos muestran mejoras en el RMSE gracias la consideración de estos efectos, particularmente los 4 principales efectos temporales mencionados.

En esta familia, una de las propuestas más completas corresponde a Koren et al. [18,19,2,22]. Bell y Koren en el 2007 consideran los efectos temporales como efectos globales, considerando las interacciones entre el tiempo y los usuarios u objetos como una variación lineal a la raíz cuadrada del número de días transcurridos desde la primera valoración realizada (por el usuario o al objeto) [18]. En el 2008 proponen utilizar parámetros asociados a intervalos de tiempo, agrupando de esta forma las valoraciones de acuerdo a la fecha de valoración, para modelar de mejor manera el cambio en el tiempo de las tendencias de usuarios y

objetos. Dado que los cambios de tendencia en objetos son relativamente lentos, se requieren pocos parámetros de este tipo, y existen suficientes datos para su ajuste. Sin embargo, en el caso de usuarios esto resulta más complejo, ya que pueden tener cambios de tendencia bruscos (como tan solo un día), por lo cual se requieren muchos parámetros, y pueden existir pocos datos para su adecuado entrenamiento [19]. Con mayores detalles, Koren [2] describe la propuesta, la que incluye diversos efectos temporales (ver sección 3.3), con lo cual se consigue disminuir considerablemente el RMSE en el dataset de Netflix (de 0.8911 a 0.8799 con un algoritmo de FC basado en factorización mediante SVD) tanto con un algoritmo basado en vecindario como con un esquema basado en factorización [2], siendo estas mejoras las que les permitieron en buena parte obtener el primer lugar en el *Netflix Prize*[22].

3. Algunas consideraciones sobre las propuestas revisadas

De acuerdo a la revisión de la literatura del área realizada, se pueden identificar tres grandes enfoques para considerar los efectos temporales en SR, los cuales van en un orden incremental respecto al nivel de detalle de los modelos empleados. El primero de ellos corresponde a la asignación de pesos en función del tiempo, en el cual usualmente se da un mayor peso a las valoraciones más recientes. El segundo corresponde al ajuste de parámetros (*tunning*) en algoritmos "tradicionales", que den cuenta de los cambios en las tendencias en el tiempo. El tercer enfoque corresponde a la incorporación de parámetros temporales a los modelos, que dan cuenta de los efectos temporales, utilizando para su modelado diferentes funciones variantes en el tiempo. A continuación se presentan los elementos más relevantes de cada uno de estos enfoques.

3.1. Pesos en función del tiempo

Este enfoque considera que los usuarios cambian sus preferencias a medida que transcurre el tiempo, y por lo tanto, las valoraciones más recientes reflejan de mejor forma sus gustos actuales. Uno de los puntos relevantes es determinar cuál o cuáles dimensiones temporales se utilizarán. Se puede considerar información temporal asociada tanto a los usuarios como a los objetos, y a su vez, existen diferentes aspectos temporales asociados a cada uno de ellos. El más común en el caso de usuario es la fecha de valoración [9,10], mientras que en el caso de objetos se puede considerar la fecha de incorporación al catálogo [11], fecha de creación del objeto [7], etc. Otro punto importante a considerar en este esquema es la forma en que se asignarán los pesos. Si bien el esquema común es incrementar los pesos de la información más reciente, la función de asignación puede variar considerablemente. Esto además puede depender del tipo de FC utilizado. Ding et al. [10] utilizan un algoritmo de FC basado en objetos, en el cual, cuando se determina el vecindario de un objeto, se asigna mayor peso a aquellos otros objetos cuyas valoraciones son más recientes, asumiendo que llevarán a un menor error en las predicciones. Para el cálculo de la similitud utilizan tanto la similitud

por coseno como la similitud por correlación de Pearson. Una vez determinada la similitud entre objetos, y para calcular la predicción, además de considerar la similitud, se añade un peso a cada objeto, dependiendo de su cercanía con la valoración más reciente. La ecuación 1 muestra el cálculo realizado:

$$\hat{r}_{i,j} = \frac{\sum_{c=1}^{k} r_{i,c} \times sim(I_j, I_c) \times W_c}{\sum_{c=1}^{k} sim(I_j, I_c) \times W_c} \quad (1)$$

donde $\hat{r}_{i,j}$ es la predicción calculada para el usuario i sobre el objeto j, k es el número de objetos en el vecindario, $r_{i,c}$ es la valoración existente del usuario i sobre el objeto c, $sim(I_j, I_c)$ es la similitud entre el objeto j y el objeto c, y W_c es el peso asignado al objeto c. Para el cálculo de los pesos se utiliza la ecuación 2.

$$W_c = \left(1 - \frac{|r_c - r_r|}{|R|}\right)^{\alpha} \quad (2)$$

donde r_c es la valoración del usuario objetivo sobre el objeto c, r_r es la valoración más reciente del usuario objetivo sobre el vecindario de objetos, $|R|$ representa la escala de valoración, y α es un parámetro de ajuste. El valor de los pesos está en el rango $[0, 1]$, y el peso de la valoración más reciente es 1. A medida que una valoración se aleja más de la valoración más reciente, el peso del objeto correspondiente es menor.

Por otro lado, Lee at al.[11] utilizan información implícita para recomendar *wallpapers* para teléfonos móviles a través de estos mismos dispositivos, considerando dos dimensiones temporales: la fecha de inclusión de un nuevo objeto en el SR (a la que llaman lanzamiento), y la fecha en que el usuario realiza la valoración. En este caso, no se cuenta con valoraciones explícitas, ya que las valoraciones corresponden a la compra por parte del usuario de un determinado objeto. Con esta información, se efectúa una clasificación por cada una de estas dimensiones temporales, y se genera una matriz de pseudo-valoraciones, que contiene valoraciones ponderadas a partir de la información de las dos dimensiones temporales mencionadas. El esquema seguido por los autores consiste en dividir cada dimensión d en un conjunto de intervalos temporales, asignando un peso diferenciado a cada intervalo. Los pesos en este estudio son incrementados linealmente: $w_d = peso\_inicial_d + \Delta peso_d \times posición_d$, donde w es el peso asignado, *peso_inicial* es un valor constante, $\Delta peso$ es el valor de incremento (constante) y *posición* es el intervalo temporal al que corresponde la valoración en cuestión en la dimensión correspondiente. El peso asignado a una valoración es $W = \sum_{d=1}^{n} w_d$ (para n dimensiones, en esta caso $n = 2$). Se debe notar que en este trabajo las valoraciones son 0 ó 1, equivalentes a compra o no compra, por lo tanto las pseudo-valoraciones serán equivalente a 0 ó al valor W calculado para la valoración respectiva.

3.2. Ajuste de parámetros en función del tiempo

En este enfoque, lo que se busca es adaptar parámetros de algoritmos de FC, con el fin de que se puedan seguir los cambios de tendencia temporales. Esta

puede ser una idea muy atractiva, por cuanto abre la perspectiva para generar un esquema de ajuste automático de parámetros en función de los cambios temporales. En este caso, se debe identificar qué elementos del modelo utilizado serán actualizados, y de qué forma. La aproximación más directa es actualizar parámetros periodicamente, buscando la minimización de un criterio basado en el error obtenido en (una ventana de) tiempo más reciente, de manera que se dé cuenta de los cambios de tendencia recientes. El trabajo más representativo de este esquema es de Lathia et al. [14], en el que se ajusta el parámetro k (tamaño del vecindario) en un algoritmo *kNN*, de forma "personalizada"para cada usuario, a medida que transcurre el tiempo. El criterio considerado es encontrar el parámetro que maximice la mejora respecto del error en el tiempo actual, como muestra la ecuación 3:

$$\forall u : k_{u,t+1} = \max_{k_i \in P}(e_i - RMSE_{t,P_i}) \tag{3}$$

donde se busca el mejor valor de k para cada usuario u, el cual tiene asociado un error (RMSE) e correspondiente a su perfil personal, P es un conjunto de posibles valores de k, t representa el intervalo de tiempo, y $RMSE$ es el RMSE sobre todos los perfiles de usuarios. La actualización del valor de k se realiza cada Δt unidades de tiempo (días por ejemplo).

3.3. Incorporación de parámetros temporales en el modelo

En este enfoque, se consideran una serie de efectos temporales, en la forma de efectos globales. Un efecto global es una desviación o tendencia que se presenta, ya sea para un usuario, objeto o su interacción, de forma sistemática y que puede ser separado como un parámetro en el modelo de predicción, de manera que se puedan normalizar los datos (remover estos efectos) antes de entrenar el modelo o aplicar una heurística de predicción. Por tanto, en este caso esencialmente se deben seleccionar qué efectos se desea considerar, y de qué forma serán modelados. Bell et al. [18] considera i) la variación de las valoraciones de un mismo usuario en el tiempo ($User \times Time(user)$); ii) la variación de las valoraciones de un usuario con respecto al tiempo desde que la película fue valorada por primera vez (por cualquier usuario) ($User \times Time(movie)$); iii) la variación de las valoraciones de una misma película desde la primera valoración de dicha película ($Movie \times Time(movie)$); y iv) la variación de las valoraciones de una película desde que el usuario realizó valoraciones por primera vez ($Movie \times Time(user)$), siendo todos estos modelados como parámetros que cambian linealmente en proporción a la raíz cuadrada del número de días transcurridos desde la primera valoración (realizada por el usuario o recibida por la película, según sea el caso).

Recientemente Koren, recogiendo lo planteado en sus trabajos anteriores más algunas extensiones, considera diveros aspectos temporales [2]. La base para incorporar estos parámetros es el uso de un predictor base, que sirve para encapsular los efectos globales de usuario y objeto. Si μ representa la valoración promedio general, un predictor base b_{ui} que incluye tales efectos para una valoración r_{ui} es:

$$b_{ui} = \mu + b_u + b_i \tag{4}$$

Los parámetros b_u y b_i representan las desviaciones observadas del usuario u y el objeto i respectivamente. Al considerar que las desviaciones pueden cambiar en el tiempo (por ejemplo, la popularidad de un objeto puede cambiar en el tiempo), los parámetros correspondientes no serán constantes, sino funciones dependientes del tiempo. Así, un predictor base sensible al tiempo sería:

$$b_{ui}(t) = \mu + b_u(t) + b_i(t) \tag{5}$$

De acuerdo con Koren [2], que realiza sus estimaciones sobre valoraciones de películas, la función $b_{ui}(t)$ representa una estimación básica para la valoración de u para i en el día t. Dado que en este caso es esperable que los objetos (películas) cambien su tendencia de valoraciones más lentamente que el cambio de la tendencia en usuarios (en este último caso, el cambio podría llegar a ser diario), los respectivos parámetros se modelan de manera diferente. En el caso de la variación de objetos, utiliza una partición de parámetros en periodos de tiempo (*bins* temporales) Así, cada día t se asocia a un entero $Bin(t)$, de manera que la tendencia en las películas se divide en una parte estacionaria y una parte variable en el tiempo:

$$b_i(t) = b_i + b_{i,Bin(t)} \tag{6}$$

En el caso de los usuarios es más complejo, puesto que se requiere de una resolución más fina para detectar efectos temporales de corta duración. Por otro lado, es poco probable que existan suficientes valoraciones por usuario para realizar estimaciones confiables en cada *bin*. De esta forma, se considera una función lineal para capturar posibles cambios graduales en la tendencia de usuario:

$$b_u(t) = b_u + \alpha_u \times desv_u(t) + b_{u,t} \tag{7}$$

con

$$desv_u(t) = sign(t - t_u) \times |t - t_u|^{\beta} \tag{8}$$

donde t_u representa la fecha media de valoración y $\beta = 0{,}4$. $desv_u(t)$ representa la desviación temporal de una valoración. *alpha* es un parámetro estático del usuario. El último término en la ecuación 9 captura variaciones de muy corta duración (en este caso, un día). Finalmente, el predictor base toma la forma:

$$b_{ui}(t) = \mu + b_u + \alpha_u \times desv_u(t) + b_{u,t} + b_i + b_{i,Bin(t)} \tag{9}$$

Todos estos parámetros se agregan al modelo de predicción utilizado (pudiendo ser un modelo de factorización o de vecindario). Sin embargo, estos parámetros no permiten capturar el cambio temporal en la interacción usuario-objeto (hasta el momento los factores son solo de usuario o de objeto). Para capturar estos últimos efectos, se siguen diferentes estrategias según se trate de un modelo de factorización o de vecindario. Para el primer caso, Koren trata los vectores de factores de usuario como dependientes del tiempo (lo que significa que los

vectores serán distintos para cada día). Para el caso de vecindarios, dado que su análisis se basa en determinar la similitud entre objetos, resulta importante buscar que las similitudes encontradas sean estables y no cambiantes en el tiempo. Para ello, utiliza una función de decaimiento exponencial en el tiempo que da mayor peso a valoraciones con menor distancia temporal, asumiendo que si un usuario valora de buena forma dos objetos en un corto periodo de tiempo, éstos tienen mayor relación que si las valoraciones están separadas por un intervalo temporal mayor.

4. Hacia un marco de trabajo sobre temporalidad en Sistemas de Recomendación

Es importante tratar de aprovechar el potencial de incluir la utilización de la dimensión temporal en los Sistemas de Recomendación. Sin embargo, las diferentes y variadas propuestas existentes en la literatura hacen necesario contar con un marco de referencia que permita determinar la mejor combinación de elementos para aprovechar este potencial, así como conocer los requisitos adicionales que debe cumplir un sistema para este cometido, de acuerdo a las técnicas utilizadas. En esta sección se delinean los principales elementos que deben ser estudiados para establecer este marco de trabajo. Esta propuesta es meramente enumerativa, y cada uno de estos elementos deberán ser estudiados con mayor detalle en trabajos posteriores:

1. *Elementos temporales (y datos) a considerar*: Si bien lo más común es considerar como variable temporal el momento (usualmente fecha) de valoración por parte del usuario, existen otras variantes que se pueden utilizar, tales como:
   - *Usuario*: Momento de incorporación al sistema, frecuencia de realización de valoraciones.
   - *Objeto*: Momento de incorporación al sistema, momento de creación, frecuencia de recepción de valoraciones.
   - *Interacción usuario-usuario* (en sistemas sociales): momento de establecimiento de relación, frecuencia de comunicación, sincronización/no sincronización de acciones.
   - *Interacción* objeto-objeto: Recepción de valoraciones sincronizada (objetos que reciben valoraciones similares por parte de un mismo usuario en un intervalo de tiempo pequeño).
   - *Interacción usuario-objeto*: Momento de valoración. Puede existir más de un tipo de objeto, y además por cada tipo de objeto, más de un tipo de valoración. Por ejemplo, un usuario podría evaluar películas y cines, y podría evaluar distintas dimensiones de una película (guión, reparto, etc.)

   La cantidad de elementos temporales considerados influirá sobre la cantidad de datos que el sistema debe almacenar, pero más importante aún, en la complejidad del algoritmo utilizado. En muchos casos se contará con SR que

estén en funcionamiento durante un largo tiempo, y por tanto los elementos temporales susceptibles de ser utilizados ya estarán limitados; sin embargo, si a pesar de esto aún existe más de un elemento temporal posible de ser utilizado, será necesario determinar las técnicas temporales que puedan ser utilizadas sobre dicho datos, frente a las dimensiones que se desee analizar.

2. *Técnica de filtrado*: Si bien el fin de la técnica de filtrado (basado en contenido, colaborativa basada en memoria, colaborativa basada en modelo, híbrida) no es directamente abordar la dimensión temporal, la técnica de filtrado utilizada podría limitar las posibles técnicas temporales aplicables. Por tanto, este elemento es también importante en este marco de trabajo.
3. *Tecnica de filtrado temporal*: Las técnicas de filtrado temporal pueden ser clasificadas de diferentes formas, dependiendo del punto de vista con que se analice. Por ejemplo, desde el punto de vista de manejo de la información, se pueden identificar dos familias: i) aquellas basadas en el análisis de información en intervalos temporales; y ii) las que son capaces de trabajar directamente sobre todo el conjunto de datos. La principal diferencia de ellas es que las primeras requieren de la generación de conjuntos diferenciados de información, mientras que las segundas pueden trabajar directamente sobre el conjunto de datos completo. Como ejemplos de la primera categoría se pueden mencionar el enfoque basado en el tiempo de Lee et al. [11,12] y el FC temporal con vecindarios adaptativos de Lathia et al. [14]. Ejemplos de la segunda categoría son el FC basado en información reciente de Ding et al. [10] y el FC con dinámica temporal de Koren [2].
   Otras formas de clasificar estas técnicas pueden ser de acuerdo a si permiten incorporar varias dimensiones temporales o no, si se asocian a técnicas de FC basadas en memoria o modelo, si modifican la forma de calcular la similitud entre usuarios (u objetos), o de acuerdo a la forma en que la información temporal se incluye en los cálculos (modificación de cálculo de similitud, ponderación de valoraciones, incorporación de parámetros globales en función del tiempo).
4. *Forma de evaluar los resultados y métricas de Evaluación*: Dependiendo de la forma en que se aplique la técnica de filtrado temporal, podría ser necesario tener que adaptar la forma en que se evalúan los resultados obtenidos por los SR, particularmente si se trabaja sólo con un subconjunto del total de valoraciones registradas en el sistema. Por ello, aunque hoy en día las métricas dominantes para evaluar el rendimiento de SR son *MAE* y *RMSE*, se han definido nuevas métricas. Por ejemplo Lathia et al. [17] propone un RMSE promediado en el tiempo (*TA-RMSE - Time-Averaged RMSE*) que es el RMSE obtenido sólo hasta el tiempo t:

$$TA - RMSE_t = \sqrt{\frac{\sum_{\hat{r}_{u,i} \in R_t}^{n} (\hat{r}_{u,i} - r_{u,i})^2}{|R_t|}} \tag{10}$$

   donde R_t es el conjunto de valoraciones hasta el tiempo t, y $r_{u,i}$ y $\hat{r}_{u,i}$ son la valoración del usuario u para el objeto i y la estimación obtenida del SR respectivamente (hasta el tiempo t). En el caso en que se trabaja

sobre una ventana temporal de valoraciones, R_t se debe reeemplazar por $r_{u,i} \in R_{t_0}, ...R_{t_n}$, con R_{t_i} como el conjunto de valoraciones realizadas en el intervalo de tiempo t_{i-1}, t_i.
Del mismo modo, se deben explorar métricas temporales asociadas a otras dimensiones, como por ejemplo la novedad o cobertura.

5. Conclusiones

En el presente trabajo se han revisado diversas propuestas para abordar la dimensión temporal en SR, y se han descrito con mayor detalle algunas consideraciones de diseño de las principales técnicas de filtrado colaborativo temporal existentes en la literatura de SR. Aunque se trata de una aplicación relativamente nueva, los resultados obtenidos con estas técnicas han sido favorables, mejorando la precisión de las recomendaciones obtenidas, por lo cual planteamos que contar con una visión general de estas diferentes propuestas puede ser de gran utilidad para la comunidad de SR. Sin embargo, la mayoría de los algoritmos han sido probados en conjuntos de datos muy particulares. Es por ello que se plantea la necesidad de crear un marco de trabajo para el estudio de la temporalidad en SR, que permita por un lado a quienes consideran incorporar esta información contar con alternativas de diseño posibles de implementar, y por otro lado facilite la creación de nuevos y mejores algoritmos.

Agradecimientos

El trabajo que se describe en este artículo ha sido subvencionado por el Ministerio de Ciencia e Innovación, Plan Nacional I+D+i, como parte del proyecto número TIN2008-06566-C04-02. El primer autor agradece el apoyo del Gobierno de Chile a través del programa Becas-Chile.

Referencias

1. Bell, R.M., Bennett, J., Koren, Y., Volinsky, C.: The million dollar programming prize. IEEE Spectr. **46**(5) (2009) 28–33
2. Koren, Y.: Collaborative filtering with temporal dynamics. In: KDD '09: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, New York, NY, USA, ACM (2009) 447–456
3. Herlocker, J.L., Konstan, J.A., Borchers, A., Riedl, J.: An algorithmic framework for performing collaborative filtering. In: SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval, New York, NY, USA, ACM (1999) 230–237
4. Su, X., Khoshgoftaar, T.M.: A survey of collaborative filtering techniques. Adv.in Artif.Intell. **2009** (2009) 2–2
5. Adomavicius, G., Tuzhilin, A.: Extending recommender systems: A multidimensional approach. In: In Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-01), Workshop on Intelligent Techniques for Web Personalization (ITWP2001). (2001) 4–6

6. Terveen, L., McMackin, J., Amento, B., Hill, W.: Specifying preferences based on user history. In: CHI '02: Proceedings of the SIGCHI conference on Human factors in computing systems, New York, NY, USA, ACM (2002) 315–322
7. Tang, T.Y., Winoto, P., Chan, K.C.C.: On the temporal analysis for improved hybrid recommendations. In: WI '03: Proceedings of the 2003 IEEE/WIC International Conference on Web Intelligence, Washington, DC, USA, IEEE Computer Society (2003) 214
8. Sugiyama, K., Hatano, K., Yoshikawa, M.: Adaptive web search based on user profile constructed without any effort from users. In: WWW '04: Proceedings of the 13th international conference on World Wide Web, New York, NY, USA, ACM (2004) 675–684
9. Ding, Y., Li, X.: Time weight collaborative filtering. In: CIKM '05: Proceedings of the 14th ACM international conference on Information and knowledge management, New York, NY, USA, ACM (2005) 485–492
10. Ding, Y., Li, X., Orlowska, M.E.: Recency-based collaborative filtering. In: ADC '06: Proceedings of the 17th Australasian Database Conference, Darlinghurst, Australia, Australia, Australian Computer Society, Inc (2006) 99–107
11. Lee, T.Q., Park, Y., Park, Y.T.: A time-based approach to effective recommender systems using implicit feedback. Expert Syst.Appl. **34**(4) (2008) 3055–3062
12. Lee, T.Q., Park, Y., Park, Y.T.: An empirical study on effectiveness of temporal information as implicit ratings. Expert Syst.Appl. **36**(2) (2009) 1315–1321
13. Lathia, N., Hailes, S., Capra, L.: knn cf: a temporal social network. In: RecSys '08: Proceedings of the 2008 ACM conference on Recommender systems, New York, NY, USA, ACM (2008) 227–234
14. Lathia, N., Hailes, S., Capra, L.: Temporal collaborative filtering with adaptive neighbourhoods. In: SIGIR '09: Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval, New York, NY, USA, ACM (2009) 796–797
15. Adomavicius, G., Tuzhilin, A.: Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE Trans.on Knowl.and Data Eng. **17**(6) (2005) 734–749
16. Herlocker, J.L., Konstan, J.A., Terveen, L.G., Riedl, J.T.: Evaluating collaborative filtering recommender systems. ACM Trans.Inf.Syst. **22**(1) (2004) 5–53
17. Lathia, N., Hailes, S., Capra, L.: Evaluating collaborative filtering over time book (2009)
18. Bell, R.M., Koren, Y.: Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In: ICDM '07: Proceedings of the 2007 Seventh IEEE International Conference on Data Mining, Washington, DC, USA, IEEE Computer Society (2007) 43–52
19. Bell, R.M., Koren, Y., Volinsky, C.: The bellkor 2008 solution to the netflix prize. Technical report (2008)
20. Koren, Y.: Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: KDD '08: Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, New York, NY, USA, ACM (2008) 426–434
21. Xiang, L., Yang, Q.: Time-dependent models in collaborative filtering based recommender system. In: WI-IAT '09: Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology, Washington, DC, USA, IEEE Computer Society (2009) 450–457
22. Koren, Y.: The bellkor solution to the netflix grand prize. Technical Report Report from the Netflix Prize Winners (2009)

Knowledge-Based Thesaurus Recommender System in MobileWeb Search*

Mario Arias[1], José M. Cantera[2], Pablo de la Fuente[1], César Llamas[1], and Jesús Vegas[1]

[1] GRINBD, Universidad de Valladolid
Camino del Cementerio S/N, 47010 Valladolid (Spain)
[2] Telefónica I+D
Parque Tecnológico Boecillo S/N, 47151 Boecillo (Spain)

Abstract. Search tools carry out a crucial role in the expansion of the mobile web. Despite the overwhelming evolution of mobile devices, user input still fails to provide a seamless user experience. To improve this situation we propose a thesaurus as knowledge-based recommender system, which acts as aid tool in query construction phase by providing semantic suggestions and autocompletion. As a result, the number of required key-presses is minimized and users can find new serendipitous options. Our evaluation reveals that this mechanism reduces user frustration and outstrips former exhaustive-typing approaches.

1 Introduction

Mobile devices have evolved in many ways, including bigger full-color screens, increased processing power and faster and permanent broadband Internet connections. These technologies have brought the World Wide Web to mobile devices introducing new possibilities, requirements and expectations. A vast majority of current web pages and search engines are usually designed to be accessed from desktop computers, which are much more powerful than mobile devices. For that reason, current mobile search experience is still far from being friendly. Search engine analysts have detected this problem and they have designed mobile views to provide the same services from a smaller interface. Their approach deals with one of the major limitations of mobile web, screen size, but it does not improve usability with reduced user input and it does not take advantage of all the benefits of a mobile environment to provide more accurate results.

Mobile web search introduces newer peculiarities that were not present in traditional web search. Nowadays users always carry modern cell phones which allow them to be permanently online wherever they go, at any time. A typical mobile web search scenario consists of a user who has an information need and is probably outdoors, so he cannot access a desktop computer easily. At this point he takes his phone and uses a web search engine to find an answer to his question. Furthermore, he is probably doing something else at the same time,

* This work is partially funded by MICINN (grant TIN2009-14009-C02-02).

like walking or talking to a friend. In this everyday situation the user needs a short, fast but also accurate answer to his query.

The first limitation the user has to face when he uses a mobile search engine is the deficient user input. According to Kamvar and Baluja studies [5], users need an average of 40 key-presses with a 12-key keypad, and about 40 seconds to enter a query with a cell phone. The required effort is too big and users get easily annoyed. This leads to the *mismatched query problem*, mobile queries become even shorter and more ambiguous than the same queries in a static desktop environment, and they are not able to guess accurately what the user intentions are. As a consequence the returned results are poor, and the user is forced to visit a lot of links and to navigate through several result pages in order to fulfill his information need. This search flow, which may be acceptable when surfing the web from a desktop computer, becomes an extremely arduous task when attempted from a mobile device. Indeed, the overwhelming number of indexed pages and results per query, implies that most of them are not relevant for that specific situation [7].

To cope with this problem, we propose an aid tool to simplify the query construction phase and to gather as much information as possible with the minimum user effort. All this information can be used in conjunction with traditional information retrieval techniques to provide more appropriate search results.

Recommender systems appear as a natural improvement to the limited user input problem, as they allow users both to save typing time and to find new serendipitous options. In mobile search the user will typically attempt to perform a few well-known tasks, like finding places or services nearby. A task-specific thesaurus can be tailored to include the most common concepts together with their synonyms and relationships. For example, if someone is outside looking for a place to have dinner, he can directly type "dinner", but what is he exactly looking for? Some possibilities are a restaurant or a supermarket nearby, or a place to order takeaway dinner. The thesaurus could include the concept "dinner" related to "food", "restaurant" and "supermarket". A traditional term-based approach is not able to guess these relationships hence a thesaurus based solution seems more valuable. In this paper we propose this thesaurus as skeleton for a knowledge-based recommender system.

In order to personalize web search to each user and situation, and due to the intrinsic mobility of this kind of devices, environment modeling becomes a crucial factor to guess what the user intentions are in those circumstances. There are some scenarios where further tasks are clearly obvious if we know the details; for instance, if someone has just landed from his flight he probably wants to find a taxi or book a hotel. This fact which may appear quite evident, is completely ignored when using a traditional search engine. We have introduced a Delivery Context definition which models all significative properties about user and environment. These properties can be used to provide personalized application behavior, and then to extend the knowledge of our recommender system and make it context-aware.

We have incorporated the Delivery Context definition and the thesaurus recommender system into a working prototype to test their feasibility together, and to analyze how can they improve the quality of the mobile web search experience.We have also designed a novel user interface to support these new features and to satisfy all mobility requirements. In this work we focus on the operation of the recommender system.

The rest of this paper is organized as follows. Section 2 briefly describes the background and related work regarding personalization and recommender systems. The architecture description of our system and the explanation of our proposal can be found in section 3. The evaluation and results are shown in section 4. Finally, we summarize the conclusions and the guidelines for future work in section 5.

2 State of the Art

People find hard typing their own queries to express intentions, but they are good at recognizing them when they are already written. For that reason, recommender systems are introduced as an aid tool to simplify the search input process. They are usually based on a predefined vocabulary, probably generated by an information retrieval engine or a knowledge background, and they provide suggestions that better match user interests from a large data repository.

Recommender systems have been a decisive element of modern online shopping because of the huge amount of products available [8]. On the other side, there are only a few mobile recommender systems and they are typically designed for bigger devices like PDA [10]. Previous works [6] study the viability of applying recommender systems to simplify mobile web search. They have detected that the users who were asked to enter queries on a search interface with query suggestions saved nearly half of the key presses, rated their workload lower and their enjoyment higher.

There are several recommendation paradigms. *Content-based* recommendation finds documents which description matches user profile. *Collaborative filtering* tries to analyze previous user ratings to identify patterns and apply them to future queries. Finally, *Knowledge-based Recommendation* uses a knowledge model to infer which suggestions are more suitable to the user in a logical sense.

We need to gather user preferences and interests for each user to provide personalized suggestions. All this information is collected in a user profile. The first way to construct it is by explicitly asking what his interests are with a small survey form. This method is known to be *high quality* if the survey is correctly designed, but in general users dislike filling out forms, especially when the benefits are not immediately obvious. In addition, users do not feel safe by submitting personal information to untrusted servers [7].

Collaborative filtering techniques analyze user behavior to create implicit user profiles. They take into account which links are more frequently clicked or the time spent visiting each one, to extrapolate user interests without them even realizing. Users with similar interests can also be grouped in classes, once

the individual has been classified, assumptions about his interests can be made based on those of the whole group. This introduces the concept of social profiling, which can be valuable when a new user connects to the application for the first time and not many details about him are available. There are studies which show that, from a statistical point of view, implicit measures of interest are equivalent to user explicitly collected data when navigation history is large enough [9].

Knowledge-based recommender systems are an interesting approach because they use an ontology to guide suggestions. They do not need a learning phase like collaborative filtering and the user does not have to fill any form, so they can be used from the connection of the first user. On the other hand, knowledge must be previously tailored based on some data source. Usually this construction phase must be manually accomplished by analysts, thus requiring a great effort, but it provides a formal definition of reality and allows to infer new facts about the domain [1].

Using a thesaurus as a recommender system is a particularization of a knowledge-based recommender system. It uses a thesaurus to model knowledge by using concepts, synonyms of each concept and different kind of relationships among them. All concepts and synonyms can be used as vocabulary for suggestions, and given one concept, discovering related ones is as easy as following that concept relationships.

Besides their advantages, each of the paradigms has its drawbacks. For example content-based and collaborative filtering suffer from the *ramp-up* problem [2]. They need a big amount of users and ratings to provide valuable suggestions. In addition, when a new document or item is added to the corpus, the system needs a learning phase to gather information about them. On the other hand, *knowledge-based* mechanisms need a previous knowledge acquisition phase.

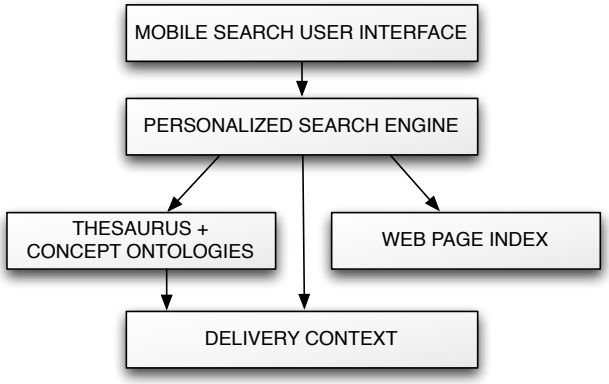


Fig. 1. Mobile Web Search Framework.

To minimize some of the disadvantages of each recommendation paradigm, a hybrid solution can be constructed [2]. The suggestions from different rec-

ommendation paradigms can be merged by ranking aggregation, and then the best-scored ones are suggested. Another option is switching between one recommender system mechanism and another depending on the situation, or suggesting the best items of both algorithms at the same time. Yet another option is applying them in cascade, the first algorithm does a first filter of the item domain, and the next application does a fine-grain selection among those options.

3 Our Proposal

To alleviate previously described mobile web search limitations we propose a framework with novel features in mind, which is depicted in Figure 1. In this paper we are focusing on the recommender system as an aid in query construction.

The Delivery Context module is in charge of modeling all knowledge about user interests and environment. This module provides a framework to make personalization and context awareness possible. Each context-aware feature can state what conditions must the delivery context satisfy for that feature to be activated.

To simplify user input in both time and effort dimensions, we propose to use the thesaurus as base for the recommender system. This thesaurus is used to model semantic relationships among concepts, which can be useful for several applications. For instance, once the user has selected one thesaurus concept, we can suggest him related ones, not only from a syntactic but also from a semantic point of view.

Finally, the user interface generates web pages to interact with the user. It uses the thesaurus and ontologies to help the user construct the query. The objective is not only to help user with query construction but also gather all information possible to let the personalized search engine provide better results.

Now that we have an overall view of our solution, we will proceed to describe in detail the operation of our recommender system.

3.1 Recommender System operation

We propose a thesaurus knowledge-based recommender system as a natural approach to semantically guide the query construction process. The thesaurus is a set of concepts or subjects, which are tied together with semantic relationships composing a lattice. Each of the thesaurus concepts also contain synonyms in several languages to provide a better description of that subject. Internationalization is easy, to add a new language only a translation of the terms is required, the thesaurus hierarchy remains unaltered.

How to profit from a thesaurus as a recommender system is quite straightforward. Given the first typed letters, the suggestion system can find matching thesaurus concepts, and at the same time show their relationships as shown in Figure 2.

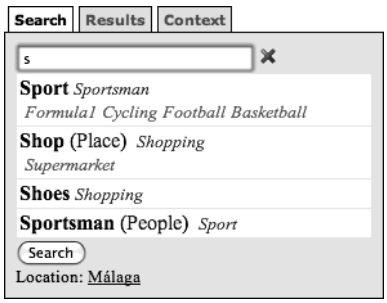


Fig. 2. Thesaurus-based suggestions tree.

In addition, a thesaurus based solution has other advantages. For example, it can be used to solve sense ambiguity within queries. Once one thesaurus concept is selected, we know which is the exact meaning of the word, because the user selected that one directly. He is able to distinguish that concept among others because he is reading its related concepts at the same time. The search engine can use this information to ensure that results are referring to that specific sense of the word, therefore achieving a search guided by concepts instead of terms. A common example of term ambiguity is the word *jaguar*, it could refer to the animal, the car manufacturer or the Apple Operating System among others.

The thesaurus is complemented with an additional ontology describing each concept. For example if the user has selected the concept "Restaurant", the ontology models different kind or types of restaurant to let the user select in a form which options he wishes. This information can be used not only to obtain a more complete description of user intentions, but also as input to the user profiling engine. In fact, these options are yet another type of explicitly selected user interests. In each query, from one to three short questions are asked. Since these questions are directly related to the subject, the user feels encouraged to answer them to improve the search. This process is fast and the user obtains instant feedback on the benefits of the extra work by receiving more accurate results. Users feel less annoyed when filling a small, query-related form at search time, than answering a huge survey to specify all his preferences the first time they use the application when they have no idea of what use they will do of the application in the future.

Search engine internals may also use the thesaurus to gather more information about that specific concept. For example, if too many results were found the engine could try extending the query to disambiguate and get more precise results. In the opposite case, when no results available it could search again any synonym or related term so results may be less accurate but it saves user time and avoids disturbing him unnecessarily.

One of the most interesting features is that thesaurus suggestion mechanism can be easily made context-aware. If the thesaurus is extended with the preferred conditions for each concept, we can increase priority to those concepts which are

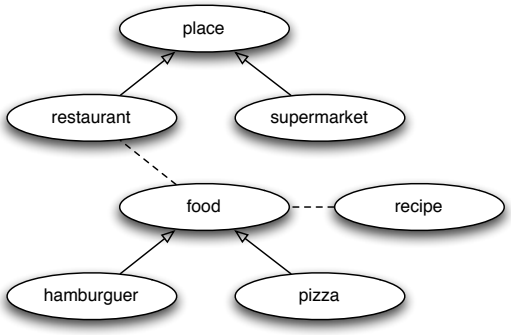


Fig. 3. Thesaurus concept lattice example.

more suitable for that situation and they will appear on top. This way, the system is able to guess user intentions much sooner, fewer key-presses are required to guess user intentions.

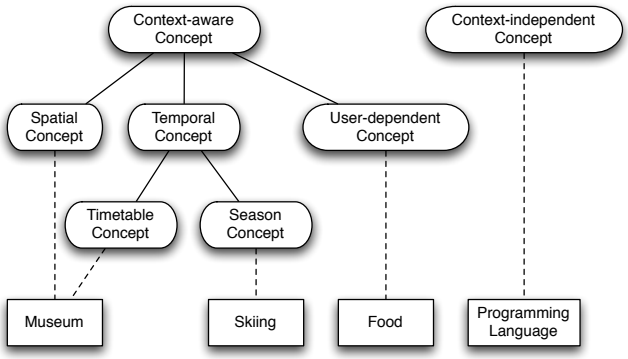


Fig. 4. Thesaurus concept classification.

To simplify context awareness, we classify the thesaurus concepts in several classes (Figure 4). For example, the thesaurus concept *programming language* does not depend on user context, so it is classified in the *Context-Independent Concept* group. In contrast the concept *transport* strongly relies on user location, and should be classified in the *Context-Aware Concept* group. In the same manner, all the context aware concepts can be grouped according to the type of property they depend on. For instance, *Spatial Concept*, *Temporal Concept* and *User-Dependent Concept*. In this last case, obviously the same concept can belong to several classes.

Our Delivery Context definition is described as an OWL ontology including SWRL [3] rules to provide more abstract layers of knowledge. To make our the-

saurus context-aware we define an abstract property *isSuitableInContext* which is described for each concept or classification class as follows:

$$concept(?c) \wedge context(?t) \wedge \{conditions(?t)\} \rightarrow isSuitableInContext(?c, ?t)$$

where $conditions(?t)$ is a logical equation which refers to the state of the delivery context to specify that the concept is valuable in that situation.

A pure knowledge-based recommender approach does not automatically reflect evolution in user needs or intentions, so we decided to add some of the collaborative filtering ideas to our recommender system. After applying rules to discard not applicable concepts, the remaining ones are candidates to be suggested. The system will use a collaborative filter algorithm in cascade to reorder them according to previous usage patterns and scores.

The handicap of a knowledge-based approach is that the thesaurus must be constructed and populated by a human to successfully model task-specific concepts, relationships and context conditions. This is an important analysis and maintenance job but only humans have the expert knowledge to intelligently model reality. To simplify this task, a semi-automatic acquisition tool to help thesaurus analysts can be developed. For example the underlying Information Retrieval system can supply most important term lists to guide knowledge engineers in which areas they should focus to cover the most common tasks sooner. A fully unsupervised version is also feasible, based on a general purpose thesaurus like WordNet. Once terms are identified by Information Retrieval engine, the module tries to associate them, identifying synonyms and concepts that are near within WordNet synset graph. Obviously, the results obtained by this module cannot be as accurate as human developed ones, but may be enough to increase search engine scope.

In this first approach, we have focused on the most common tasks for mobile devices; like transport, leisure or services. We have restricted our thesaurus concepts to these situations in order to cover them sooner, and to ease in-depth study. This first approach serves as cold-start for the prototype to launch collaborative training.

Our thesaurus implementation has been defined by using RDF metadata format with SKOS [4] vocabulary. It contains 100 concepts and about 200 relationships. We can divide them in three kind of relationships: broader, narrower and related. For example, a place is a broader term of restaurant or museum, and food is related to restaurants. This way we can differentiate inheritance from other kind of relationships and compose a concept hierarchy, as shown in Figure 3.

3.2 User Interface Design

User interfaces for mobile devices must be carefully designed to successfully accomplish all their requirements. Our objective is to maintain traditional search

Fig. 5. Thesaurus selected concept with additional options from ontology.

engine user interaction flow, but at the same time increase its functionality by using our thesaurus recommender system.

The search presentation contains an input box where the user can type his query in a traditional way. While he is typing, the system proposes below all suggestions that match those letters in a tree view. Each concept has its own branch, including the broader concept to know which is the sense of the word, and other concepts which are related or narrower to the main concept. To distinguish the kind of relationship each one is shown with a different color.

At any time the user can select one of the suggested concepts, and then additional options about it are shown (see Figure 5). The thesaurus tree is kept on screen focused on the selected term, including all related concepts. This way the user can navigate through the thesaurus just by clicking without typing. At any time he can visit the results page to check results, or go back to the query tab to select different options. The options are taken directly from the ontology.

We have taken advantage of the AJAX capability of most powerful devices to minimize downloaded data, by only obtaining needed resources through an asynchronous HTTP request instead of downloading the whole page over and over again.

4 Evaluation

As a first approach to evaluate our novel introduced ideas, we did an exploratory analysis by employing qualitative market research techniques. Firstly, we used group discussion studies to analyze which features were more interesting and should be developed. Afterwards, to ensure that the application satisfies all kind of audience, 12 people were invited to evaluate the prototype divided in three groups. The group A was composed by students in the first year of MsC in Computer Science; the second group, B, was integrated by IT technicians, and the third group, C, was comprised of users with no specials skills in the mobile search process. We asked them to play with our interface for a few minutes until they felt comfortable with its usage. Then we proposed to find the answers to

several questions by using our search engine. Finally they filled a small survey with questions like how friendly does the interface look like, if they think that it is faster than manual typing the whole query and what new features they would introduce in our system.

In general, users were able to use the system without bigger problems. They mostly complained about the system not providing certain suggestions or results of their interest, but this is a consequence of the reduced scope of our thesaurus. Indeed if the searched concepts do not belong to the thesaurus they cannot be suggested. A production version or subject-specific deployment will require to adjust thesaurus for its purpose.

The users found our suggestion system useful and intuitive. They all agreed that a recommender system reduces typing time and allows to use the system with less effort.

5 Conclusions and Future Work

We have detected that one of the most severe limitations that prevents mobile web search from taking off is the inadequate user input. As solution we propose a task-specific thesaurus recommender system to enhance functionality and appease the nuisance. We emphasize the importance of the context in order to provide personalized suggestions and search results. Our results show that these techniques are valuable to ensure that the users are really able to enjoy a faster and easier mobile web search.

The introduced thesaurus-based recommender system successfully copes with our limited user input problem. With just a few characters typed and a few clicks the system is able to gather much more information, improving user experience and minimizing query introduction time. In the worst case if the user is searching for something that is not modeled in the thesaurus, the system will behave like a traditional search engine.

Our prototype is currently in an early phase and there is still a lot of research to do. We began developing a framework which supports most modern ideas, and we started writing simple algorithms to let the search engine work. Now we must study each of the modules separately to learn how to obtain best results from them. Our future work must focus on how to extend thesaurus with minimum effort to provide a wider scope without decreasing accuracy. Then, we must study how to apply user profiling and collaborative filtering techniques to better reflect user intentions and behavior and to know which are the essential rules for context-awareness. Finally, we must study how to integrate all gathered facts with traditional information retrieval techniques to provide personalized results.

6 Acknowledgements

We would like to thank Jorge Cabrero, Guido García and Álvaro Zubizarreta for their help.

References

1. R. Burke. Knowledge-based recommender systems. In A. Kent, editor, *Encyclopedia of Library and Information Systems*, volume 69, Supplement 32, New York, 2000. Marcel Dekker.
2. R. Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, 2002.
3. I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean. SWRL: A semantic web rule language combining owl and ruleml. W3C member submission, W3C, may 2004. http://www.w3.org/Submission/SWRL/.
4. A. Isaac and E. Summers. SKOS simple knowledge organization system primer. W3C working draft, W3C, feb 2008. http://www.w3.org/TR/2008/WD-skos-primer-20080221/.
5. M. Kamvar and S. Baluja. Deciphering trends in mobile search. *Computer*, 40(8):58–62, 2007.
6. M. Kamvar and S. Baluja. Query suggestions for mobile search: understanding usage patterns. In *CHI '08: Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, pages 1013–1016, New York, NY, USA, 2008. ACM.
7. K. Keenoy and M. Levene. Personalisation of web search. In *Intelligent Techniques for Web Personalization, IJCAI 2003 Workshop, ITWP 2003, Acapulco, Mexico*, pages 201–228, 2003.
8. Q. N. Nguyen and F. Ricci. User preferences initialization and integration in critique-based mobile recommender systems. In *Artificial Intelligence in Mobile Systems 2004, in conjunction with UbiComp 2004*, pages 71–78, Nottingham, UK, 2004.
9. R. White, I. Ruthven, and J. M. Jose. The use of implicit evidence for relevance feedback in web retrieval. In *Advances in Information Retrieval, 24th BCS-IRSG European Colloquium on IR Research Glasgow, UK, March 25-27, 2002 Proceedings*, pages 93–109, 2002.
10. R. T. A. Wietsma and F. Ricci. Product reviews in mobile decision aid systems. In E. Rukzio, J. Hkkil, M. Spasojevic, J. Mntyjrvi, and N. Ravi, editors, *PERMID*, pages 15–18. LMU Munich, 2005.

Un método de indexación en línea para recuperación de información multimedia*

Luis G. Ares, Nieves R. Brisaboa, Alberto Ordóñez, Óscar Pedreira

Laboratorio de Bases de Datos, Facultade de Informática, Universidade da Coruña
Campus de Elviña s/n, 15071, A Coruña
{lgares,brisaboa,alberto.ordonez,opedreira}@udc.es

Resumen. La búsqueda por similitud es una de las operaciones básicas en recuperación de información multimedia. Los espacios métricos constituyen un marco formal para el planteamiento de soluciones eficaces a este problema. Los métodos existentes tienen en común la necesidad de realizar una fase de indexación off-line que puede resultar relativamente costosa, tanto en términos de espacio ocupado como de tiempo de proceso necesario. Estos condicionantes hacen que algunas de las soluciones existentes sean inviables en el tratamiento de grandes volúmenes de datos. En este trabajo presentamos un método de indexación en línea que no requiere el proceso de indexación inicial. Esta cualidad resulta determinante para su aplicación a casos en los que no se puede realizar el procesado previo de los elementos, lo que sucede frecuentemente en aplicaciones que presentan demandas exigentes. Este método resulta especialmente adecuado en escenarios en los que se produce una secuencia de aparición de los objetos, sin disponer de la totalidad de ellos desde el inicio, como ocurre en los procesos de distribución de contenidos mediante streaming.

1 Introducción

Hoy en día, en numerosos campos, como por ejemplo la medicina, la biología y la comunicación audiovisual, se manipulan datos complejos como imágenes, video, sonido, datos geográficos, etc., y se demandan aplicaciones cada vez más sofisticadas como son la recuperación de información basada en el contenido [1], los sistemas de recomendación [2], y la detección de documentos duplicados [3], entre otros. En estos contextos, el planteamiento tradicional de la búsqueda exacta resulta inadecuado debido a la ausencia de estructura de los datos, además de que su relevancia es muy reducida, al tener como objetivo únicamente la obtención de elementos completamente iguales a otros, cuando lo que resulta interesante es encontrar los que son parecidos según algún criterio.

La búsqueda por similitud o búsqueda de los vecinos más cercanos, se convierte así en un problema común en aplicaciones que procesan colecciones de

* Este trabajo ha sido parcialmente financiado por el Ministerio de Ciencia e Innovación (PGE y FEDER) ref. TIN2009-14560-C03-02.

datos no estructurados, como son los contenidos multimedia. La implementación trivial de esta operación consiste en comparar el objeto de consulta con todos los objetos de la base de datos. Sin embargo, esta no es una opción válida en la práctica, debido a que la comparación de dos objetos puede ser computacionalmente muy costosa y a que las colecciones a procesar suelen tener una gran cantidad de objetos.

Existen varias propuestas para implementar la búsqueda por similitud en los entornos multimedia. La más elemental consiste en utilizar información adicional que describe el contenido de los objetos, generalmente mediante palabras clave o etiquetas, que además de suponer un largo y tedioso trabajo, puede presentar sesgos debido a la subjetividad de quien lo realiza [1], por lo que no resulta adecuada para grandes cantidades de datos. Otra alternativa es la de utilizar los propios datos multimedia para realizar una búsqueda basada en contenido. Se trata de una propuesta con la que se han obtenido resultados eficientes, pero su aplicación suele estar condicionada por las características del dominio.

Las aplicaciones que necesitan realizar búsquedas por similitud sobre contenidos multimedia presentan una serie de características comunes, como son la existencia de un universo de objetos sobre los que actuar y la necesidad de evaluar la semejanza entre esos objetos. Estas características resultan adecuadas para su tratamiento mediante los espacios métricos, en los que se tiene un conjunto de objetos y se define una función de distancia, no negativa, simétrica y que cumple la desigualdad triangular, que permite establecer un criterio riguroso para determinar la similitud entre los objetos. Esta alternativa es de aplicación más general, ya que los métodos de búsqueda por similitud en espacios métricos se basan en una formalización del problema en la que no se tienen en cuenta los detalles internos de los objetos de la colección, ni la implementación de la función de distancia utilizada para compararlos [4]. Las características de los datos multimedia pueden convertirse en valores de elementos de un espacio vectorial n-dimensional, que es un caso particular de espacio métrico, al que se le aplica una función de distancia adecuada.

El principal problema que presentan los métodos basados en espacios métricos es que la evaluación de la función de distancia resulta, en general, muy costosa, por lo que estos métodos indexan la colección. Esto, unido a la utilización de la desigualdad triangular para descartar los objetos que están a una distancia mayor que la deseada, permite reducir considerablemente el número de comparaciones necesarias para efectuar una consulta.

Otro problema importante presente en muchos métodos de este tipo es el tamaño que puede llegar a tener el índice, que puede convertirlos en inadecuados para procesar grandes colecciones reales. Por otra parte, la creación del índice se realiza previamente a la operación de búsqueda, en un procesamiento *off-line* cuyo coste y viabilidad reducen el campo de aplicación de los métodos, como ocurre en los entornos en los que no es posible disponer con antelación del índice sobre los objetos, debido por ejemplo a que se produce una secuencia de aparición de los mismos. Esto ocurre, por ejemplo, en la distribución de contenidos mediante streaming, en los casos en los que se presentan problemas

de memoria debido al tamaño del índice [5] y en situaciones en las que la colección de objetos presenta cambios frecuentes que hacen que el mantenimiento del índice sea inviable.

En este trabajo proponemos un nuevo método de búsqueda en espacios métricos que no requiere la construcción del índice con antelación, sino que la efectúa en línea a medida que realiza la búsqueda. La efectividad del método queda patente en los resultados obtenidos tras la evaluación experimental, realizada tanto sobre colecciones sintéticas como reales.

El resto del artículo está estructurado de la siguiente forma: la Sección 2 ofrece una introducción a los métodos de búsqueda en espacios métricos, y revisa las propuestas existentes para optimizar la búsqueda por similitud sin utilizar índices o con índices en línea construidos a medida que se van resolviendo las consultas. La Sección 3 presenta el método que proponemos en este artículo. En la Sección 4 presentamos los resultados obtenidos en la evaluación experimental del método, y finalmente, la Sección 5 presenta las conclusiones extraídas y las líneas de trabajo futuro.

2 Antecedentes y Trabajo Previo

2.1 Búsqueda en Espacios Métricos

El problema de la búsqueda por similitud puede formalizarse utilizando el concepto matemático de espacio métrico. Un espacio métrico es un par (X, d), donde X es un *universo* de objetos válidos y $d : X \times X \longrightarrow \mathbb{R}^+$ es una *métrica*, es decir, una función que determina la distancia o disimilitud $d(x, y)$ entre dos objetos cualesquiera $x, y \in X$, y que cumple las siguientes propiedades: es estrictamente positiva ($d(x, y) \geq 0$), simétrica ($d(x, y) = d(y, x)$), y cumple la desigualdad triangular ($d(x, y) \leq d(x, z) + d(z, y)$. La *colección* de objetos es un subconjunto finito $U \subseteq X$ del universo de objetos válidos.

Aunque existen varios tipos de consulta, las más comunes son la búsqueda por rango $R(q, r)$, que recupera los objetos a una distancia de la consulta q menor o igual que el radio de búsqueda r, y la búsqueda de los k vecinos más cercanos a la consulta, $kNN(q)$. La búsqueda por rango es la más general y la búsqueda de k vecinos más cercanos puede implementarse en términos de esta.

Los espacios vectoriales son un caso particular de espacios métricos. El par formado por un espacio vectorial y la distancia Euclídea, $(\mathbb{R}^l, L_2)$, es un ejemplo de espacio métrico. Un conjunto de palabras que se comparan utilizando la distancia de edición es también un ejemplo de espacio métrico.

El objetivo de los métodos de búsqueda en espacios métricos es indexar la colección para reducir el número de comparaciones necesarias para resolver la consulta. La única información que pueden utilizar estos métodos es la que queda reflejada en la formalización del problema. Así, un mismo método puede aplicarse en cualquier dominio que cumpla las restricciones del modelo.

Los métodos de búsqueda en espacios métricos se clasifican en dos grupos: basados en clusters y basados en pivotes. En ambos casos, se distingue entre una

primera fase de indexación *off-line*, y una fase de búsqueda en la que se utiliza el índice para optimizar las consultas.

Los métodos basados en clusters crean una partición de Voronoi del espacio de búsqueda. Para cada cluster, el índice almacena el identificador del objeto utilizado como centro, c, y el radio cobertor del cluster, r_c (distancia del centro al objeto más alejado). En general, la partición se construye de forma recursiva, por lo que los índices tienen estructura de árbol. Dada una consulta $R(q, r)$, se compara q con cada centro de cluster, y se descartan directamente aquellos clusters para los que la región definida por (c, r_c) tienen intersección vacía con la región definida por (q, r). Algunos de los métodos basados en clusters más reconocidos son: GNAT [6] o List of Clusters [7].

Los métodos basados en pivotes seleccionan un conjunto de objetos de la colección como referencias o *pivotes*. En este caso, el índice almacena las distancias $d(x_i, p_j)$ de los objetos de la colección x_i a los pivotes p_j. Dada una consulta $R(q, r)$, se compara q con los pivotes para obtener las distancias $d(p_j, q)$. Aplicando la desigualdad triangular sabemos que podemos obtener una cota inferior (l_b, *lower bound*) de la distancia real $d(q, x)$ entre el objeto x y la consulta q como:

$$l_b = |d(p, q) - d(p, x)|$$

Si el valor de la cota inferior es superior al radio de búsqueda, $l_b > r$, el objeto x puede ser descartado del resultado sin compararlo con la consulta q. Como propuestas representativas de métodos basados en pivotes, podemos citar FQT [8], AESA [9], o LAESA [10].

2.2 Motivación

Todos los métodos que hemos citado funcionan en dos fases: una primera fase de indexación *off-line* de la colección y una fase de búsqueda en la que se utiliza el índice para optimizar las consultas. Sin embargo, hay situaciones en las que esta aproximación no es posible [11]:

- *Frecuencia de cambios elevada*: en muchas aplicaciones, se insertan nuevos objetos en la colección de forma constante y, en general, esos nuevos objetos eliminan a otros que ya no se van a utilizar. Es el caso de *stream databases* [12], o series temporales, entre otros, en los que el contenido de la colección cambia constantemente. Como ejemplo, podemos pensar en una colección de imágenes recibidas constantemente de múltiples satélites y sobre las que hay que buscar determinados patrones. En un caso como este, no es viable tener una fase de indexación *off-line*.
- *Consultas aisladas*: en ciertas aplicaciones de minería de datos, la búsqueda por similitud se ejecuta muy pocas veces, como un paso más de un proceso de análisis más complejo. En estos casos, el coste de construcción de la totalidad del índice antes de las consultas, es mucho mayor que el coste de resolver las consultas con una búsqueda secuencial.

– *Función de distancia arbitraria*: en algunas aplicaciones, la función de distancia no es constante, sino que va cambiando con el tiempo (distancias que implican aprendizaje, definidas por el usuario, dependientes de la consulta, etc.). El coste de una fase de indexación *off-line* se perdería en el momento en que cambie la función de distancia.

2.3 Trabajo Previo

Una posible solución a estas situaciones es hacer una búsqueda secuencial y tratar de optimizarla en la medida de lo posible. Existen propuestas en esta línea para el caso de espacios vectoriales. El método VA-File [13], almacena versiones comprimidas de los vectores y, durante la búsqueda, obtiene valores aproximados de las distancias reales utilizando los vectores comprimidos, descartando así todos los objetos que pueda utilizando las distancias aproximadas.

En el caso general de búsqueda por similitud en espacios métricos, existe una única propuesta para este problema en particular hasta la fecha, denominada D-File [11]. En lugar de optimizar una búsqueda secuencial, D-File construye un índice relativamente pequeño en memoria, utilizando la información que obtiene al resolver las consultas.

La primera consulta se resuelve haciendo una búsqueda secuencial, y esa información se almacena temporalmente para optimizar las siguientes consultas. Básicamente, se construye un pequeño índice basado en pivotes con dos características: los pivotes ya no son objetos de la base de datos, sino consultas ya procesadas y las distancias se almacenan en una caché con un tamaño limitado, de forma que se van reemplazando a medida que llegan nuevas consultas.

Cuando hay que procesar una consulta, se compara con las consultas ya procesadas, que actúan ahora como pivotes, y se sigue el mismo proceso que en cualquier otro método para descartar objetos directamente sin compararlos con el objeto de consulta.

3 OCMI: Online Cacheable Metric Index

En este trabajo proponemos un nuevo método de indexación y búsqueda en espacios métricos cuya característica principal es que no necesita una fase de procesamiento previo de la colección para crear el índice. Para optimizar las búsquedas, se crea un pequeño índice en memoria reutilizando la información obtenida al procesar cada consulta. Esto convierte a nuestra propuesta en una opción de gran utilidad en entornos en los que la secuencia de cambios en la colección hace que no sea posible la construcción de un índice completo.

La idea de nuestro método es, al igual que el D-File [11], crear un pequeño índice que se pueda almacenar en memoria principal y que pueda ser construido, con un coste cercano a cero, a partir de la información que se obtiene al resolver las consultas. Además, su reducida complejidad espacial permite que toda o gran parte de su estructura pueda almacenarse en memoria caché.

3.1 Descripción de la Estructura del Índice

Nuestro método se basa en construir, durante las búsquedas, un pequeño índice basado en pivotes que se actualiza dinámicamente con la información que se obtiene al resolver las consultas. Al contrario que el D-File, que almacena en una matriz dispersa las distancias entre los objetos de la colección y una serie de consultas, nuestro método almacena la distancia de cada objeto a la consulta que, realizando la función de pivote, tiene más posibilidades de descartarlo en búsquedas futuras. En Celik [14] se demostró que dicho pivote era aquel que más cerca estaba del objeto. Así, la estructura de nuestro índice está formada por un vector que almacena, para cada objeto indexado, la distancia a su pivote más prometedor, es decir, a su consulta ya procesada más cercana y un puntero a dicho pivote. En la Figura 1 se puede ver una representación gráfica del índice, formada por los siguientes componentes:

- Un vector de *Pivotes/Consultas*: para almacenar las consultas que van a desempeñar la función de pivotes.
- Un vector de pares $(piv(x_i), d(x_i, piv(x_i)))$, donde:
  - $piv(x_i)$ es un puntero a la consulta que ha sido seleccionada para realizar la función de pivote más prometedor para el objeto x_i.
  - $d(x_i, piv(x_i))$ es la distancia entre el objeto x_i y su pivote más prometedor $piv(x_i)$.

Así, mientras que el D-File presenta una complejidad espacial $O(n \times m)$, siendo n el número de objetos indexados y m el número de consultas utilizadas como pivotes, nuestro método tiene complejidad espacial $O(n+m)$. Con nuestra propuesta el índice puede incluso almacenarse, en parte o en su totalidad, en memoria caché. De esta propiedad se deriva su nombre: *OCMI, Online Cacheable Metric Index*. Además, al almacenar menos información para cada objeto, se abre la posibilidad de elegir más consultas como pivotes, por lo que aumenta la probabilidad de encontrar un pivote mejor para cada objeto.

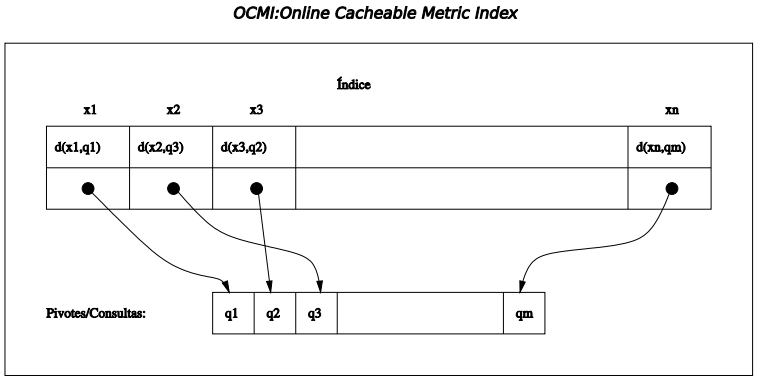


Figura 1. Estructura del índice

3.2 Indexación y Búsqueda

En esta sección describimos cómo se actualiza la información que contiene y cómo se utiliza dicha información para realizar el filtrado de los objetos que no son de interés para una consulta dada.

Búsqueda. En este apartado nos centramos en la implementación de la búsqueda por rango, ya que es la más general y otras operaciones pueden ser implementadas en función de ella [4].

El algoritmo que se propone es muy similar al de la mayoría de métodos de búsqueda por similitud basados en pivotes, constando de los siguientes pasos:

1. Se calcula la distancia entre la consulta q y todos los pivotes, con lo que se obtiene el siguiente vector de distancias:

$$\forall j \in [0, m), dqp[j] = d(p_j, q)$$

2. Para cada objeto x_i indexado se calcula la cota inferior l_b de la distancia entre x_i y la consulta q de la siguiente forma:

$$l_b = |d(x_i, piv(x_i)) - dqp[piv(x_i)]|$$

Según el valor que se obtenga para la cota inferior l_b y, según el radio de búsqueda r determinado, se pueden dar dos situaciones diferentes:

1. Si la cota inferior l_b es mayor que el radio de búsqueda, el objeto x_i se puede descartar sin ser comparado directamente con la consulta.
2. Si por el contrario la cota inferior es menor o igual que el radio de búsqueda r, se tendrá que comparar el objeto x_i directamente con la consulta q, a fin de determinar su grado exacto de similitud.

De esta forma, utilizando conjuntamente la información almacenada en el índice y la desigualdad triangular, se consigue pasar de una complejidad $O(n)$ para la búsqueda secuencial a otra de la forma $O(n^\alpha), \alpha \in (0, 1]$ [15] utilizando el índice.

Indexación. Una de las característcas más importantes de nuestro método es que no requiere ningún tipo de procesamiento previo de la colección. La construcción del índice está basada en la reutilización de las comparaciones que se llevan a cabo durante las búsquedas. Por este motivo decimos que la construcción del índice tiene coste cero, ya que no se realiza ninguna evaluación de la función de distancia más que las estrictamente necesarias para resolver las consultas. Además, en este método se propone reutilizar los objetos de consulta para realizar la función de pivotes, a diferencia de los métodos anteriores que seleccionaban como pivotes objetos de la colección. En este sentido seguimos la misma idea que se propone para el D-File [11].

Sin embargo, a diferencia de lo propuesto por este último, en el que se almacenan las distancias entre todos los objetos y un subconjunto de pivotes/consultas

(complejidad espacial $O(n \times m)$), nuestro método almacena la distancia entre cada objeto y un único pivote/consulta. Determinar cuál es el mejor pivote para cada elemento de la colección, es decir, el pivote que maximiza la probabilidad de descartar ese objeto durante una búsqueda, es un tema ya estudiado en numerosos trabajos ([14], [16], entre otros). De ellos se deriva que uno de los criterios más eficaces consiste en elegir como pivote *más prometedor* para cada objeto, aquel que más cerca está de él.

Así, una vez determinado el criterio que aporta los mejores pivotes, se propone una extensión del algoritmo de búsqueda para construir y actualizar el índice aprovechando únicamente los cálculos que se realizan durante las consultas, es decir, se plantea una forma de construir el índice a coste prácticamente cero.

En el Algoritmo 1 se resume el procedimiento de búsqueda y actualización del índice. El primer paso consiste en calcular las distancias entre la consulta q y todos los pivotes seleccionados hasta el momento. Acto seguido, utilizando dichas distancias y la información almacenada en el índice, se determina, para cada objeto x_i, la cota inferior de su distancia real a la consulta, pudiendo darse dos casos: que el objeto sea descartado o que el objeto deba compararse directamente con la consulta. En esta última situación, se aprovecha dicha comparación para determinar si la actual consulta q está más próxima al objeto x_i que su actual consulta o pivote más cercano ($piv(x_i)$). Si esto último se cumple, la consulta q pasará a ser el pivote más prometedor para el objeto x_i. De esta forma se consigue mejorar el índice de manera iterativa sin necesidad de realizar más evaluaciones de la función de distancia que las estrictamente necesarias para resolver cada consulta.

4 Resultados Experimentales

4.1 Entorno de Pruebas

Para llevar a cabo la evaluación de nuestra propuesta, realizamos experimentos con varios espacios métricos que representan colecciones de datos reales y cuyas métricas son representativas de un amplio espectro de complejidad algorítmica. En concreto, las colecciones de prueba que utilizamos son:

- NASA: contiene 40.151 imágenes extraídas de repositorios de imágenes y vídeo de la NASA, cada una de ellas representada por un vector de características de 20 componentes. Como métrica para comparar las imágenes se utilizó la distancia euclídea ($O(n)$, siendo n el número de componentes de cada vector).
- COLOR: contiene 112.682 histogramas de color codificados como vectores de 112 dimensiones. Como métrica se utilizó la distancia euclídea L_2.

Algoritmo 1: Algoritmo de búsqueda por rango y actualización del índice

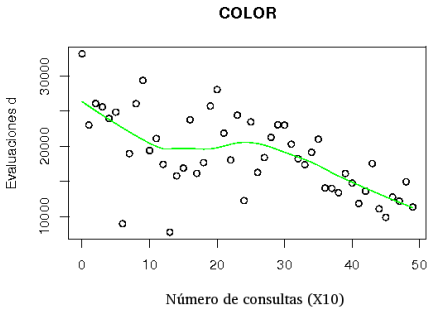
```
   Input: q (query object), r (search radius)
   Output: ResultSet
 1 for j ← 0 to m do
 2     dqp[j] ← d(q, q_j);
 3 end
 4 ResultSet ← {};
 5 for i ← 1 to n do
 6     lb ← |d(x_i, piv(x_i)) − dqp[piv(x_i)]|;
 7     if lb ≤ r then
 8         real ← d(x_i, q);
 9         if real ≤ r then
10             ResultSet ← ResultSet ∪ {x_i};
11         end
12         if d(x_i, q) ≤ d(x_i, piv(x_i)) then
13             piv(x_i) ← q
14         end
15     end
16     return ResultSet
17 end
```

- ENGLISH: contiene 69.069 palabras extraídas de un diccionario de inglés. Se utilizó la distancia de edición, calculada como el número de símbolos a insertar, eliminar o substituir en una cadena para convertirla en otra ($O(m \times n)$, sindo m y n la longitud de cada cadena).
- GERMAN: contiene 75.086 palabras extraídas de un diccionario de alemán. Como métrica, al igual que para la colección ENGLISH, se utilizó la distancia de edición.

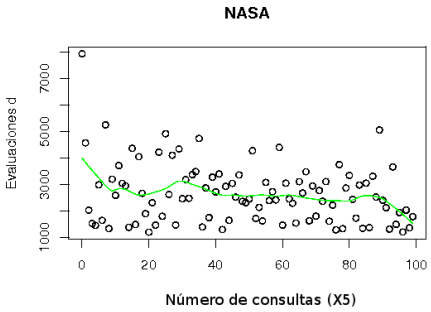
Todas las pruebas se realizaron sobre una máquina con procesador AMD Athlon(tm) 64 Processor 3800+ (2,4 GHz), 8GB de memoria RAM y 512Kb de memoria caché, sistema operativo Ubuntu 8.04.4 LTS (kernel Linux 2.6.24-26-generic (x86_64)). La compilación se realizó con GNU C (gcc 4.2.4) con el flag -09 para optimización activado.

4.2 Evolución del Rendimiento del Índice

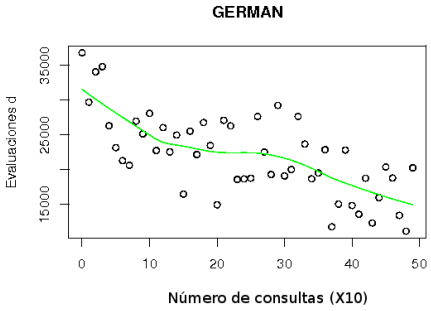
En las Figuras 2(a), 2(b), 2(c) y 2(d) se muestra la evolución del rendimiento del índice, en términos del número de evaluaciones de la función de distancia para resolver cada búsqueda, a medida que se va incrementando el número de consultas utilizadas como pivotes. En todas las gráficas referenciadas, se puede apreciar un notable incremento en la capacidad del índice para descartar objetos según se van realizando nuevas búsquedas. Este hecho se debe a que el índice está en un proceso de mejora continua ya que, con cada consulta, existe la posibilidad de sustituir el pivote más prometedor de cada objeto indexado por otro cuya capacidad para descartar objetos es aún mayor.



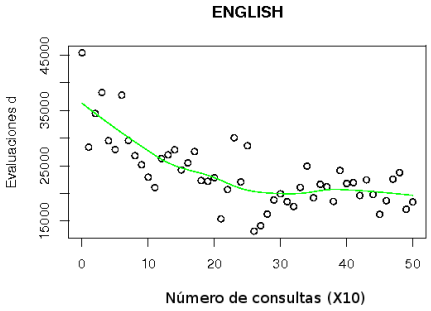
(a) Resultados para la colección COLOR



(b) Resultados para la colección NASA



(c) Resultados para la colección GERMAN



(d) Resultados para la colección ENGLISH

4.3 Eficiencia

La Tabla 1 muestra la siguiente información:

- Colección: nombre de la colección indexada.
- Tamaño(n): número de objetos de la colección.
- Indexado: número de elementos de la colección que han sido indexados. Se corresponden con el 90% del tamaño total n.
- #d: número medio de evaluaciones de la función de distancia necesarias para resolver una grupo de consultas (el número de consultas por cada grupo se corresponde con el 10% del total de las colecciones indexadas).
- %Descartados: indica el tanto por cien de objetos descartados por el índice en cada consulta.
- t_{OCMI}: tiempo que necesita la propuesta *OCMI* para resolver cada grupo de consultas.
- t_{sec}: tiempo necesario para resolver cada grupo de consultas en el caso de no utilizar índice, es decir, aplicando la búsqueda secuencial.

De dicha tabla se extrae que la capacidad de nuestra propuesta para descartar objetos, pese a no requerir un procesamiento *off-line* de las colecciones para crear

el índice, varía entre un 70% y un 87%. También son significativas las diferencias en cuanto a tiempo necesario para resolver cada grupo de consultas, ya que la búsqueda secuencial implica un tiempo de ejecución unas cinco veces mayor que en caso de utilizar el índice *OCMI*. Los resultados, en cuanto al porcentaje de

Colección	**Tamaño(n)**	**Indexado**	**#d**	**%Descartados**	t_{OCMI}	t_{sec}
ENGLISH	69.069	62.162	18.621,17	70,00%	32,03	113,07
GERMAN	75.086	67.578	18.235,11	73,00%	82,42	331,77
COLOR	112.682	101.422	12.952,30	87,00%	56,68	307,47
NASA	40.151	36.135	5.298,23	85,00%	2,76	6,95

Tabla 1. Tabla de resultados

objetos descartados por el índice, son mejores que los resultados obtenidos por D-File en [11]. Además, nuestro método utiliza mucho menos espacio en memoria para almacenar el índice ya que pasamos de una complejidad $O(n \times m)$ en D-File a $O(n + m)$ en nuestro método.

5 Conclusiones

En este artículo presentamos un nuevo método de indexación en línea para búsqueda por similitud en espacios métricos. Su principal diferencia con los métodos existentes es que no separa las fases de indexación *off-line* y búsqueda, sino que el índice se construye dinámicamente reutilizando la información que se obtiene al procesar cada consulta. De esta forma, el método puede utilizarse en problemas en los que la colección de datos se ve afectada por inserciones y eliminaciones constantes, en los que no es viable la construcción de un índice completo antes de empezar a procesar las consultas.

El método propuesto presenta mejores resultados que el método D-File [11] en cuanto al número de objetos descartados durante la búsqueda. Además, nuestro método necesita mucho menos espacio por lo que podría almacenarse completamente en la memoria caché, incluso para colecciones del orden de millones de objetos. Además, el coste de construcción del índice es cercano a cero, ya que no se realizan más evaluaciones de la función de distancia que las estrictamente necesarias para resolver cada consulta.

El hecho de que sea un método para búsqueda por similitud en espacios métricos, y no para un problema particular, permite aplicarlo de la misma forma a diferentes ámbitos, como recuperación de imágenes por contenido, detección de documentos duplicados, o sistemas de recomendación, entre otros.

Referencias

1. Blanken, H., Hiemstra, D., Jonker, W., Petkovic, M., de Jong, F., Feng, L.: Multimedia Retrieval. Springer (2007)

2. Shapira, B., Rokach, L.: Building Effective Recommender Systems. Springer (2010)
3. Bar-Yossef, Z., Keidar, I., Schonfeld, U.: Do not crawl in the dust: Different urls with similar text. ACM Transactions on the Web **3**(1) (2009) 1–31
4. Chávez, E., Navarro, G., Baeza-Yates, R., Marroquín, J.L.: Searching in metric spaces. ACM Computing Surveys **33**(3) (September 2001) 273–321 ACM Press.
5. Bustos, B., Navarro, G.: Improving the space cost of k-nn search in metric spaces by using distance estimators. Multimedia Tools and Applications **41**(2) (2009) 215–233
6. Brin, S.: Near neighbor search in large metric spaces. In: Proc. of the 21st Conf. on Very Large Databases (VLDB'95). (1995) 574–584
7. Chávez, E., Navarro, G.: A compact space decomposition for effective metric indexing. Pattern Recognition Letters **26**(9) (2005) 1363–1376
8. Baeza-Yates, R., Cunto, W., Manber, U., Wu, S.: Proximity matching using fixed-queries trees. In: Proc. of 5th Symp. on Combinatorial Pattern Matching (CPM'94). LNCS(807), Springer (1994) 198–212
9. Vidal, E.: An algorithm for finding nearest neighbors in (approximately) constant average time. Pattern Recognition Letters **4** (1986) 145–157 Elsevier.
10. Micó, L., Oncina, J., Vidal, R.E.: A new version of the nearest-neighbor approximating and eliminating search (aesa) with linear pre-processing time and memory requirements. Pattern Recognition Letters **15** (1994) 9–17 Elsevier.
11. Skopal, T., Bustos, B.: On index-free similarity search in metric spaces. In: Proc. of 20th Conf. on Database and Expert Systems Applications (DEXA'09). LNCS(5690), Springer (2009) 516–531
12. Aggarwal, C.C.: Data streams: models and algorithms. Springer (2006)
13. Weber, R., Schek, H.J., Blott, S.: A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. In: Procs. of 24rd Int. Conf. on Very Large Data Bases (VLDB'98), ACM Press (1998) 194 – 205
14. Celik, C.: Priority vantage points structures for similarity queries in metric spaces. In: Proc. of EurAsia-ICT 2002: Information and Communication Technology. LNCS(2510), Springer (2002)
15. Navarro, G.: Analyzing metric space indexes: What for? In: Proc. of 2nd Int. Workshop on Similarity Search and Applications (SISAP'09), IEEE CS Press (2009) 3–10
16. Ares, L.G., Brisaboa, N.R., Esteller, M.F., Pedreira, O., Places, A.S.: Optimal pivots to minimize the index size for metric access methods. In: Proc. of 2nd Int. Workshop on Similarity Search and Applications (SISAP'09), IEEE CS Press (2009) 74–80

Estado del arte en Wordspotting aplicado a los sistemas de extracción de información en contenidos de voz

Javier Tejedor[1], Doroteo Torre[2], and José Colás[1]

[1] HCTLab, Universidad Autónoma de Madrid, España,
javier.tejedor@uam.es
[2] ATVS, Universidad Autónoma de Madrid, España

Resumen. La gigantesca cantidad de datos almacenados en formato audio, y en particular en lenguaje oral, hace necesario el desarrollo de tecnología para posibilitar su acceso. Aunque en el año 2000, Garofolo afirmó que el problema de la extracción de información en contenidos de audio estaba resuelto (TREC-99) gracias a los reconocedores de habla continua de gran vocabulario (LVCSR), desafortunadamente no es así, debido a las palabras fuera de vocabulario. Los términos más usados en las consultas contienen nombres propios, extranjerismos, acrónimos..., que no suelen estar en el vocabulario de los reconocedores de habla y que pueden representar un 12% del total de los términos buscados (Logan [6]). Por ello, nuevos sistemas de "Word Spotting" o "reconocimiento de palabras clave" están emergiendo para un eficiente acceso a la información. En este trabajo se realiza una recopilación del estado del arte en sistemas de extracción de información en contenidos de voz, con particular énfasis en las técnicas de "Word Spotting". Hasta la fecha, la combinación de LVCSR con sistemas de "Word Spotting" basados en sub-unidades de palabra (fonemas, sílabas...) proporciona los mejores resultados en la búsqueda de información.

1 Introducción

El continuo crecimiento en el volumen de datos que se almacenan en formato audio ha provocado el desarrollo de numeros sistemas para el acceso a dicha información [1–5]. Tras las jornadas celebradas en la "Text Retrieval Conference" (TREC-99) a través de la evaluación de "Spoken Document Retrieval" en el año 2000, se afirmó que el problema de la extracción de información en contenidos sonoros estaba resuelto. Tal solución pasaba por el desarrollo de sistemas de reconocimiento de voz de habla continua de gran vocabulario (LVCSR), de tal forma que una simple búsqueda lineal (mediante un detector de palabras clave) en la salida de los mismos de las palabras que forman parte de la consulta del usuario sería suficiente para proporcionar al usuario el contenido deseado, tal y como se detalla en la Fig. 1. Sin embargo, con posterioridad se comprobó que tal proceso no era del todo eficiente. El problema radica en el hecho de que dichos

sistemas únicamente pueden proporcionar al usuario contenidos sonoros que hagan referencia a las palabras que conforman el diccionario (vocabulario) de los mismos. De este modo, por ejemplo, un reconocedor de voz que no tenga incorporado en su diccionario el nombre del presidente del gobierno de un determinado país, hace imposible un acceso a las grabaciones donde aparezca el nombre de dicho presidente. Normalmente los reconocedores de habla continua de gran vocabulario suelen contener en torno a 65.000 palabras en su diccionario, y entre ellas no se suelen encontrar nombres propios, acrónimos, extranjerismos, etc, que además son las más importantes a la hora de procesar la consulta del usuario. Estas palabras son conocidas como palabras "Out-Of-Vocabulary" (OOV) y su procesamiento es indispensable en cualquier motor de búsqueda de contenidos de audio. En un estudio realizado por Logan [6], se afirmó que en torno al 12% de las palabras que forman parte de una consulta típica del usuario son OOV. Además, debido a la inevitable aparición de nuevas palabras continuamente para un determinado idioma (en torno a 20.000 por año [7]), el problema de las palabras OOV siempre existirá independientemente del número de palabras contempladas en el vocabulario de los LVCSR. Debido a todo esto, el campo de word spotting, aplicado a la extracción de un conjunto de palabras clave dentro de repositorios de audio, necesita del desarrollo de sistemas que resuelvan de una forma eficiente la consulta del usuario.

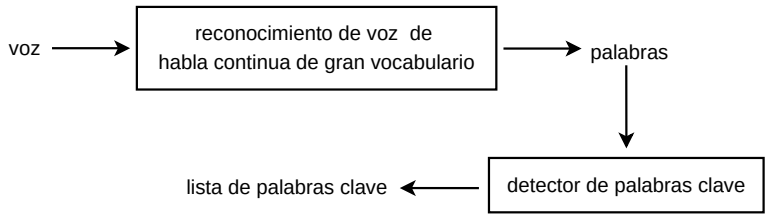


Fig. 1. *La voz es procesada a través de un reconocedor de habla continua de gran vocabulario y las palabras que forman parte de la consulta del usuario son detectadas por el* detector de palabras clave *a partir de la salida del reconocedor de voz.*

Los sistemas de word spotting miden su rendimiento en términos de aciertos y falsas alarmas. Un acierto ocurre cuando la palabra propuesta por el sistema aparece en el audio mientras que una falsa alarma se produce cuando la palabra propuesta por el sistema no tiene su correspondencia en el audio.

En este trabajo se realiza un estudio del estado del arte de los sistemas que se han desarrollado para la recuparación de información de contenidos de voz, centrándose en los sistemas de word spotting. Dichos sistemas pertenecen principalmente a uno de estos tres tipos, amén de la posible combinación que se puede realizar entre los mismos: 1) Word spotting basado en LVCSR, 2) Word spotting basado en modelos de relleno y 3) Word spotting basado en reconocedores de voz de sub-unidades de palabra. El primero de ellos es el que mejor funciona en caso de que todas las palabras que forman parte de la consulta del usuario

formen parte del vocabulario del sistema, lo cual no siempre ocurre como se explicó anteriormente. Los sistemas basados en modelo de relleno proporcionan la segunda mejor tasa de resultados, pero tienen el problema de que si cambia alguna palabra a buscar es necesario reprocesar todo el audio (según se explicará en la Sección 3), por lo que son inmanejables en situaciones donde el vocabulario de las mismas cambia frecuentemente, quedando su uso limitado a aplicaciones tipo call-centers, consulta de itinerarios, gestión de reservas, etc, donde el vocabulario se mantiene prácticamente constante. Finalmente los sistemas de word spotting basados en reconocedores de sub-unidades de palabra no necesitan reprocesar el audio cuando el vocabulario de la aplicación cambia, al realizar la decodificación acústica con un vocabulario compuesto por fonemas, grafemas, sílabas, etc, cuyo número siempre se mantiene constante en un idioma. Sin embargo, este tipo de sistemas, al no considerar ningún modelo de palabra durante el proceso de reconocimiento, es el que peor resultado obtiene, si bien a veces este hecho queda compensado por la mayor velocidad de procesamiento del audio y la mayor flexibilidad para manejar vocabularios que cambian con el tiempo. La Tabla 1 muestra la tasa de resultados presentada por Szoke et.al [8] de cada uno de estos tres tipos de sistemas de word spotting. Cada uno de estos tres tipos de sistema tiene sus ventajas y sus inconvenientes y depende de la aplicación final el uso de uno u otro o incluso de la combinación entre ellos. Como veremos en la Sección 5, la combinación de los sistemas basados en LVCSR y los basados en reconocedores de voz de sub-unidades de palabra son los que actualmente se usan en aplicaciones que versan sobre la extracción de información en contenidos de audio, tal y como se representa en la Fig. 2.

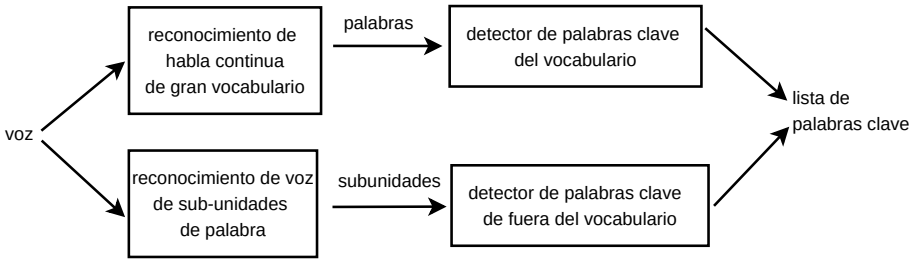


Fig. 2. *Las palabras de la consulta del usuario que forman parte del vocabulario del LVSCR son buscadas en la salida del reconocedor de habla continua de gran vocabulario, mientras que las palabras OOV son accedidas a través del reconocedor de voz de sub-unidades de palabra.*

El resto del artículo se divide en las siguientes secciones: En la Sección 2 se realizará una breve introducción a los sistemas de word spotting basados en LVCSR aplicados a la extracción de información. En la Sección 3 se detallarán los principales sistemas de word spotting basados en modelos de relleno, en la Sección 4 se describirán los principales sistemas de word spotting basados en

Sistema	FOM
LVSCR	66.95
Modelos de relleno	64.46
Sub-unidades	58.90

Tabla 1. Tasa de resultados de los tipos de word spotting. *LVCSR* es el word spotting basado en reconocimiento de habla continua de gran vocabulario. *Modelos de relleno* se refiere al sistema de word spotting basado en modelos de relleno y *Sub-unidades* se refiere al sistema de word spotting basado en sub-unidades de palabra. La métrica de resultados es la Figure-of-Merit (FOM), definida como número de aciertos por cada 5 falsas alarmas por palabra clave por hora.

reconocedores de voz de sub-unidades de palabra y en la Sección 5 se detallará la combinación de dichos sistemas aplicados a la extracción de información en contenidos de audio. Finalmente en la Sección 6 se concluye este trabajo.

2 Word spotting basado en reconocedores de habla continua de gran vocabulario

El reconocimiento de habla continua de gran vocabulario se caracteriza por perseguir el reconocimiento de habla pronunciada de manera natural y cubriendo un vocabulario extenso (del orden de decenas de miles de palabras). Posiblemente la mejor manera de seguir la evolución de esta tecnología es a través de las evaluaciones de reconocimiento de voz realizadas por el *National Institute of Standards and Tecnhonogy* (NIST) de EE.UU. [9, 10]. Una revisión del estado en sistemas LVCSR queda, simplemente por cuestiones de espacio, completamente fuera de los objetivos de este artículo, máxime cuando existen excelentes libros que cubren esta necesidad [11, 12].

Inicialmente, una de las mayores preocupaciones al emplear sistemas LVCSR para recuperación de información de voz era la tasa de error de los sistemas LVCSR, que resultaba especialmente elevada si las condiciones del audio y la voz sobre las que trabaja el reconocedor no coinciden con las condiciones de entrenamiento del mismo. Este problema, el problema de la *robustez* de los reconocedores de voz frente a variaciones del tipo de audio o del tipo de habla, sigue siendo una de las mayores limitaciones para el uso de los sistemas LVCSR en recuperación de información, y lo que hace prácticamente imprescindible que el sistema de reconocimiento de voz se *adapte* al entorno en el que se va a usar mediante un laborioso proceso manual que exige gran cantidad de datos lo más adaptados posible a dicho entorno. Afortunadamente, se ha comprobado que a pesar de tener tasas de error de palabra elevadas (incluso del orden del 50%), los sistemas LVCSR siguen proporcionando información útil para la recuperación de información [13]. Esta información resulta también esencial para presentar el contenido del audio sin necesidad de reproducirlo, lo que permite una más rápida visualización y navegación por los resultados de una búsqueda. Por todos estos

motivos, un sistema LVCSR es prácticamente imprescindible, pero no suficiente, en un sistema de recuperación de información de audio.

3 Word spotting basado en modelos de relleno

Estos sistemas, a diferencia de los basados en LVCSR, contienen en su diccionario únicamente las palabras clave que se desean extraer del audio junto con unos modelos especiales de relleno. Es sabido que el proceso de decodificación acústica propone la/s secuencia/s más probable/s de palabras que aparecen en el audio. De esta forma, no sólo hay que tener en cuenta las palabras clave en el proceso, sino también cualquier tipo de palabra, efecto, etc que pueda aparecer en el audio. Esto último se consigue con los modelos de relleno, los cuales intentan absorber las palabras que aparecen en el audio que no se corresponden con ninguna de las palabras clave que se desea buscar. De esta forma, la salida de estos sistemas, está compuesta por el conjunto de palabras clave propuestas durante el proceso de decodificación, junto con los modelos de relleno que dicho proceso propone cuando considera que en cierta región de voz no existe ninguna palabra clave. Este tipo de sistemas fueron propuestos por Rose y Paul [14] en el año 1990. Como se ha podido ver en trabajos posteriores, el entrenamiento de los modelos de relleno, en cuanto a las unidades acústicas que se usan para su construcción, juega un papel fundamental a la hora de obtener un mejor rendimiento. Así, se han investigado modelos de relleno basados en fonemas, grafemas, sílabas, clases amplias, palabras, etc:

Rose y Paul [14] compararon el uso de palabras, fonemas (en su versión dependiente e independiente del contexto) como modelos de relleno y observaron que los modelos basados en fonemas dependientes del contexto obtuvieron los mejores resultados, seguido por los fonemas independientes del contexto y los modelos de palabra. Al usar únicamente 80 palabras para construir los modelos de relleno basados en palabras, éstos obtuvieron los peores resultados al no ser capaces de cubrir una gran parte de palabras que están presentes en la señal de voz. Sus modelos de relleno fueron entrenados a partir de aquellas regiones de la señal de voz que no se corresponden con ninguna palabra clave, mientras que los modelos de palabra clave fueron entrenados con toda la voz.

Manos y Zue [15] investigaron el uso de fonemas independientes del contexto y la agrupación (*clustering*) de éstos en clases amplias (oclusivas, nasales, fricativas, etc). Dentro de estas clases amplias, investigaron 18, 12 y 1 modelos de relleno. Todos estos modelos de relleno fueron entrenados a partir de la señal de voz que no contenía ninguna palabra clave. Como modelos de palabras clave, investigaron el uso de fonemas independientes del contexto entrenados a partir de toda la voz y entrenados a partir de la voz que únicamente contenía palabras clave. Demostraron que el uso de fonemas independientes del contexto como modelos de relleno y el uso de los modelos de fonema entrenados con las regiones de voz que contienen las palabras clave obtuvieron los mejores resultados.

Cuayahuitl y Serridge [16] compararon el uso de fonemas, sílabas y palabras como modelos de relleno para su sistema de word spotting desarrollado para

español. Para construir el modelo de relleno basado en palabras, usaron un conjunto de 30 palabras, además de las sílabas. Los modelos de las palabras clave se forman como concatenación de los modelos de fonema. En su trabajo, observaron que el modelo de relleno basado en sílabas era el que obtenía los mejores resultados, en tanto que aun no representando un modelo tan robusto como el de palabra, sí es más robusto que el modelo de fonema y a diferencia del modelo de palabra, es capaz de cubrir toda la variación del idioma.

Trabajos de word spotting para lenguajes orientales usaron la sílaba como unidad acústica con la que construir el modelo de las palabras clave, debido a las peculiaridades de los mismos. Así, Xin y Wang [17], en su sistema para el lenguaje chino mandarín, cada modelo de palabra era una concatenación de dos o tres sílabas. Como modelos de relleno, usaron un modelo acústico genérico entrenado a partir de toda la voz junto un modelo de antísilaba por cada sílaba, entrenado a partir de todas las sílabas menos para la cual se entrenaba el modelo.

A pesar de que el proceso de decodificación acústica de este tipo de sistemas obtiene un rendimiento bastante aceptable, medidas de confianza, con el objetivo de eliminar aquellas hipótesis (palabras clave) propuestas por dicho proceso que no tuviesen una puntuación (score) deseada han sido ampliamente investigadas. De esta forma el esquema general de estos sistemas tienen la arquitectura representada en la Fig. 3.

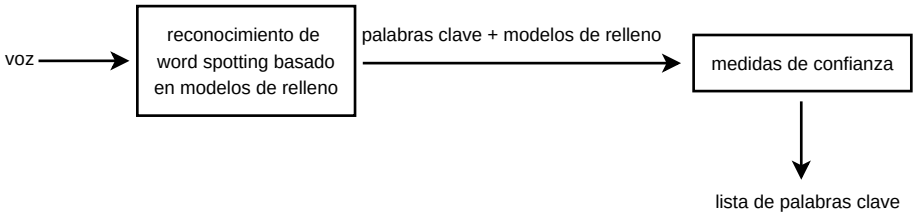


Fig. 3. *Esquema general de la arquitectura de word spotting basado en modelos de relleno.*

Numerosos trabajos como los de Cuayahuitl y Serridge [16], Xin y Wang [17], Tejedor et.al [18], o Szoke et.al [8] propusieron una medida de confianza a partir de la puntuación dada a la palabra clave durante el proceso de decodificación, de tal forma que las hipótesis que no tuviesen una puntuación por encima de un determinado umbral eran rechazadas de la salida final del sistema.

Ou et.al, [19] propusieron el uso de redes neuronales a partir de ciertas características de entrada obtenidas durante el proceso de decodificación (puntuación global de la palabra, puntuación de las N-mejores hipótesis, etc) de tal forma que aquellas hipótesis que la red neuronal clasifica como falsas alarmas eran rechazadas de la salida final del sistema. Máquinas de soporte vectorial (SVM) también han sido usadas para esta tarea de clasificación en trabajos presentados por Ben Ayed et.al [20, 21].

Tejedor et.al, [22] propusieron una medida de confianza basada en la semejanza que existe entre la salida de un reconocedor de fonemas en aquellas regiones de la señal de voz donde el reconocedor de word spotting basado en modelos de relleno proponía cada hipótesis y la secuencia real de fonemas de la hipótesis dada por éste último.

Estas medidas de confianza lograban aumentar el rendimiento del sistema de word spotting basado en modelos de relleno, en términos de aciertos y falsas alarmas.

4 Word spotting basado en reconocedores de voz de sub-unidades de palabra

Aunque los sistemas basados en sub-unidades de palabra han sido ampliamente usados desde hace tiempo para resolver el problema de word spotting, ha sido desde que el NIST llevara a cabo en el 2006 la primera evaluación de "Spoken Term Detection" (STD) [23] cuando este tipo de sistemas han proliferado como mecanismos de extracción de información en contenidos de audio. En estas evaluaciones, no sólo es importante la relación entre los aciertos y falsas alarmas que el sistema era capaz de generar, sino también la velocidad con la que el sistema era capaz de realizar la búsqueda en el audio. Por tanto, los sistemas suelen emplear una primera fase offline, donde se realiza el proceso de reconocimiento de voz y una fase online donde se realiza la búsqueda de los términos. Debido a que ésta debe ser lo más rápida posible, el hecho de tener que reprocesar todo el audio para buscar la lista de términos es inviable. Así, el proceso de reconocimiento de voz debe basarse en sub-unidades de palabra (fonemas, grafemas, sílabas, grafones, etc), que no cambian en un idioma. Este proceso genera un índice a partir del cual, en la fase de búsqueda, con el detector de palabras clave, el sistema puede devolver el resultado de acuerdo a la consulta del usuario. Medidas de confianza, con el objetivo de calcular de la mejor forma posible las puntuaciones para cada hipótesis del sistema y rechazar aquellas que no sobrepasen un cierto umbral, completan la arquitectura de estos sistemas desarrollados para STD, tal y como aparece en la Fig. 4.

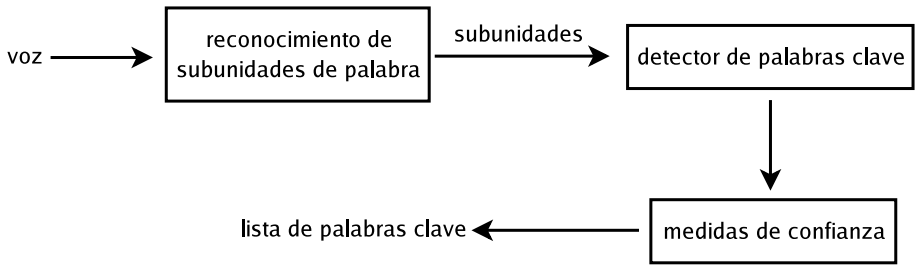


Fig. 4. *Esquema general de los sistemas de STD.*

Amir et.al [24], propusieron un sistema de reconocimiento de voz de gran vocabulario a partir del cual se obtenía la secuencia de fonemas correspondiente a la secuencia de palabras más probable (*1-best*) que proponía de dicho sistema. Así, dicha secuencia de fonemas era almacenada como índice y una posterior búsqueda de la secuencia de fonemas correspondiente a la consulta del usario en este índice, producía el resultado de la consulta. Debido a los posibles errores que pueden derivarse del proceso de reconocimiento, el sistema incorporó en la búsqueda una matriz de confusión donde se guardaban las diferentes probabilidades de inserción, borrado y substitución para cada fonema. En dicha búsqueda, programación dinámica es usada para, tomando la secuencia de fonemas de la consulta, la secuencia de fonemas resultante del proceso de reconocimiento de voz y la matriz de confusión, obtener las hipótesis finales del sistema.

Thambiratnam y Sridharan [25] y Szoke et.al [8], entre otros, propusieron la construcción del índice a partir de un reconocedor de fonemas, donde se guardaba para cada instante de tiempo más de una hipótesis (es decir, más de un fonema que podría ocurrir en la señal de voz) en forma de grafo (lattice) y un posterior algoritmo de programación dinámica proponía las hipótesis que, en la posterior fase de búsqueda, eran encontradas en dicho grafo. Dicho algoritmo también tenía en cuenta posibles errores de inserción, borrado y substitución de los fonemas almacenados en el *lattice*.

Aunque normalmente el tipo de sub-unidades de palabra usado para hacer el proceso de reconocimiento de voz son los fonemas, el uso de grafemas, sílabas, grafones han sido investigadas. Así, por ejemplo, Tejedor et.al [26], demostraron que el uso de los grafemas puede llegar a mejorar a los alófonos para español, debido principalmente a la relación que existen entre grafema-fonema en dicho lenguaje y al menor número de grafemas (28) que de alófonos (47) en español. También Wang et.al, [27] demostraron que para palabras OOV y el idioma inglés, debido a los errores de transcripción en los conversores grafema-fonema, lo cual conlleva a una cierta incertidumbre a la hora de realizar la predicción de la secuencia de fonemas para cada palabra, el uso de grafemas puede llevar a obtener valores cercanos al uso de fonemas. El uso de grafones (fragmentos de palabra que se entrenan a partir de la secuencia fonémica y grafémica de las palabras) también ha sido investigado y Akbacak et.al [28] demostraron su buen rendimiento a la hora de acceder a las palabras OOV. Un reconocedor silábico es usado en el sistema propuesto por Larsson et.al [29], que como en los trabajos citados anteriormente, consistía en una búsqueda en un grafo (en este caso de sílabas) de las palabras correspondiente a la consulta previa obtención de la transcripción silábica de la misma.

El uso de redes neuronales para calcular la puntuación final de cada hipótesis dada por el sistema de búsqueda también ha sido investigado en este tipo de sistemas, proporcionando un mejor rendimiento que los que presentan como puntuación final la calculada a partir de la puntuación acústica y del modelo lenguaje obtenidos durante el proceso de decodificación realizado de forma offline. Así, Wang et.al, [30] y [31] propusieron recalcular la puntuación de cada hipótesis con las probabilidades a nivel de fonema, calculadas durante el entrenamiento

de un perceptrón multicapa a partir de las características cepstrales de la señal de voz según la secuencia de fonemas correcta de cada hipótesis y a partir de la puntuación acústica y del modelo de lenguaje obtenido durante el proceso de reconocimiento de voz.

5 Combinación de sistemas de word spotting

Las ventajas y desventajas que poseen cada uno de los tipos de sistemas de word spotting comentados en este trabajo hace razonable pensar que la combinación de los mismos pueda ofrecer un mejor rendimiento del sistema final. De este modo, existen trabajos que combinan sistemas basados en reconocimiento de voz de palabra de gran vocabulario con reconocimiento de voz basado en fonemas durante el proceso de decodificación de la señal de voz, y otros que combinan las palabras clave proporcionadas por ambos sistemas cuando son ejecutados de forma independiente. Entre ellos destacan los siguientes:

Szoke et.al [32] y Vergyri et.al [33], en donde el uso de grafones como conjunto de sub-unidades de palabra proporciona los mejores resultados para la búsqueda de palabras OOV (a partir de la búsqueda en un *lattice* de grafones), mientras que las palabras que aparecen en el vocabulario del reconocedor de voz de palabra son accesibles a través del índice (en forma de *lattice* o grafo de palabras) construido durante su proceso de decodificación. Anteriormente, Yu y Seide [34] presentaron un reconocedor híbrido cuyo diccionario está compuesto por palabras y fragmentos de palabra (fonemas, sílabas, palabras completas) y cuyo proceso de reconocimiento producía un *lattice* compuesto por ambos tipos de unidades. Además, este trabajo, también presentó la combinación de dos reconocedores diferentes (reconocedor de palabra y reconocedor de fragmentos de palabra, extrayendo un *lattice* de palabras y de fragmentos de palabra respectivamente). De esta forma, la fase de búsqueda de las palabras clave se realiza sobre la salida de los dos reconocedores (las palabras OOV únicamente son accesibles a través del reconocedor de fragmentos de palabra). En su trabajo concluyeron que ambas formas de combinación proporcionaron un mejor resultado que el uso de cada una por separado.

Miller et.al [35] en donde a partir de la salida del reconocedor de voz de palabra, en forma de *lattices* de palabras, las palabras son convertidas a su representación fonética (a partir de la secuencia más probable de palabras o *1-best*) para acceder a las palabras OOV. De este modo, las palabras de la consulta que forman parte del reconocedor de voz de palabra son accedidas a través del índice, construido a partir del *lattice* de palabras, mientras que las palabras OOV son accedidas a través de la secuencia de fonemas derivada de la secuencia más probable de palabras dentro del lattice y un algoritmo de programación dinámica que encuentra las posibles hipótesis permitiendo errores de inserción, borrado y substitución en dicha secuencia.

Iwata et.al [36], también hacen uso de dos subsistemas de reconocimiento de voz en paralelo (uno basado en palabra, a partir de la secuencia más probable propuesta por el reconocedor y otro basado en fonemas), de tal forma que una

Sistema	ATWV
Reconocedor de palabra	72.5
Reconocedor de palabra + grafones	74.2

Tabla 2. Tasa de resultados a partir de las hipótesis dadas por el reconocedor de palabra y por el reconocedor híbrido de grafones y palabras. ATWV es la métrica definida por NIST para STD [23]. Cuanto mayor sea, mejor es el sistema.

Sistema	FOM
Reconocedor de palabra	49.3
Reconocedor de fonemas	64.0
Combinación	73.9

Tabla 3. Tasa de resultados a partir de las hipótesis dadas por el reconocedor de palabra, por el reconocedor de fonemas y por la combinación de las hipótesis propuestas por ambos. Debido a la palabras OOV, el sistema basado en un reconocimiento de fonemas ofrece mejores resultados que el basado en un reconocedor de palabra.

hipótesis es propuesta por el sistema final en caso de que aparezca en ambos subsistemas. Para realizar la búsqueda sobre el reconocedor de fonemas, se empleó un algoritmo de programación dinámica a partir de la secuencia más probable de fonemas propuesta por el reconocedor y el uso de una matriz de confusión para compensar los errores del reconocimiento de fonemas. La adición del reconocedor de voz basado en fonemas proporcionaba una mejora en el sistema comparado con el uso del reconocedor de voz basado en palabra.

Meng et.al [37], propusieron la fusión de las salidas de diferentes reconocedores de voz (palabras, grafemas y sílabas tonales y no tonales) para su sistema de extracción de información sobre el lenguaje chino mandarín. De este forma, los *lattices* generados a partir de cada reconocedor ejecutado de forma independiente son combinados para generar un único *lattice* el cual se almacena como índice para posteriormente ser usado en la búsqueda de las palabras clave. En su trabajo concluyeron que dicha combinación/fusión de los *lattices* mejoraba al índice que representa a cada uno de los sistemas por separado.

La Tabla 2 muestra los resultados presentados por Vergyri et.al [33], obtenidos por la incorporación de los grafones en la decodificación acústica para formar un reconocedor híbrido compuesto por grafones y palabras y su comparación con un reconocedor de palabra, donde se ve la ganancia de incoporar dicho conjunto de grafones para tratar las palabras OOV. Y la Tabla 3 muestra la mejoría en los resultados obtenidos por Yu and Seide [34] al combinar las hipótesis dadas por un reconocedor de habla continua de gran vocabulario y las dadas por el sistema construido a partir de un reconocedor de fonemas.

6 Conclusiones

En este trabajo se ha realizado un estudio del estado del arte de las diferentes aproximaciones que los sistemas de word spotting realizan a la hora de acceder

a la información contenida en repositorios de audio. Gracias a la gran cantidad de datos (tanto orales como escritos) que se dispone para multitud de lenguajes, los sistemas basados en reconocedores de habla continua de gran vocabulario ofrecen un óptimo rendimiento siempre y cuando la consulta del usuario tenga una total cobertura en el vocabulario de los mismos. Para tratar el problema de las palabras OOV que aparecen en las consultas se ha investigado el uso de reconocedores de voz basados en sub-unidades de palabra, lo cual hace que la combinación de ambos sistemas sea lo más robusto conocido en la actualidad aplicado a la extracción de información en contenidos de audio.

Referencias

1. Hauptmann, A.G., Wactlar, H.D.: Indexing and search of multimodal information. Proc. of ICASSP, 1, 195–198 (1997)
2. Koumpis, K., Renals, S.: Content-based access to spoken audio. IEEE Signal Process. Magazine, 22, 61–69 (2005)
3. Huerta, J.M., Chen, S., Stern, R.M.: The 1998 Carnegie Mellon university Sphinx-3 Spanish Broadcast News transcription system. Proc. of DARDA Workshop Broadcast news transcription and understanding, (1998)
4. Gauvain, J.L., Lamel, L., Adda, G.: The LIMSI Broadcast News transcription system. Speech Communication, 31(1–2), 89–108 (2002)
5. Hansen, J.H.L., Huang, R., Zhou, B., Seadle, M., Deller, J.R., Gurijala, A.R., Kurimo, M., Angkititrakul, P.: SpeechFind: Advances in Spoken Document Retrieval for a national gallery of the spoken word. IEEE Trans. on Speech and Audio Process., 13(5), 712–730 (2005)
6. Logan, B., Moreno, P., Van Thong, J.M., Whittaker, E.: An experimental study of an audio indexing system for the web. Proc. of ICASSP, 2, 676–679 (2000)
7. Watson, D.: Death sentence, The decay of public language. Book published by Knopf, Sydney (2003).
8. Szoke, I., Schwarz, P., Matejka, P., Burget, L., Karafiat, M., Fapso, M., Cernocky, J.: Comparison of keyword spotting approaches for informal continuous speech. Proc. of ICSLP, 633–636 (2005)
9. NIST.: NIST ASR History Web Page, Online at http://www.itl.nist.gov/iad/mig/publications/ASRhistory/index.html (2010)
10. Pallet, D.S.: A Look at NIST.s Benchmark ASR Test: Past, Present, and Future. Proc. of IEEE ASRU Workshop, 483–488 (2003)
11. Huang, X., Acero, A., Hon, H.: Spoken language processing: a guide to theory, algorithm, and system development. Prentice Hall PTR (2001)
12. Benesty, J., Shondi, M.M., Huang, Y.:, Springer Handbook of Speech Processing. Springer (2008)
13. Foote, J.: An overwiew of audio information retrieval. Multimedia Systems, 7, 2–10 (1999)
14. Rose, R.C., Paul, D.B.: A hidden markov model based keyword recognition system. Proc. of ICASSP, 129–132 (1990)
15. Manos, A.S., Zue, V.W.: A segment-based wordspotter using phonetic filler models. Proc. of ICASSP, 2, 899–902 (1997)
16. Cuayahuitl, H., Serridge, B.: Out-of-vocabulary word modelling and rejection for spanish keyword spotting systems. Proc. of MICAI, 155–165 (2002)

17. Xin, L., Wang, B.: Utterance verification for spontaneous mandarin speech keyword spotting. Proc. of ICH, 3, 397–401 (2001)
18. Tejedor, J., King, S., Frankel, J., Wang, D., Colás, J.: A novel two-level architecture plus confidence measures for a keyword spotting system. Proc. of V Jornadas en Tecnología del Habla, 247–250 (2008)
19. Ou, J., Chen, C., Li, Z.: Hybrid neural-network/hmm approach for out-of-vocabulary words rejection in mandarin place name recognition. Proc. of ICONIP, (2001)
20. Ben Ayed, Y., Fohr, D., Haton, J.P., Chollet, G.: Keyword spotting using support vector machines. Proc. of TSD, 285–292 (2002)
21. Ben Ayed, Y., Fohr, D., Haton, J.P., Chollet, G.: Confidence measures for keyword spotting using support vector machines. Proc. of ICASSP, 1, 588–591 (2003)
22. Tejedor, J., García, R., Fernández, M., López-Colino, F., Perdrix, F., Macías, J.A., Gil, R.M., Oliva, M., Moya, D., Colás, J., Castells, P.: Ontology-based retrieval of human speech. Proc. of DEXA, 485–489 (2007)
23. NIST.: The spoken term detection (STD) 2006 evaluation plan. Online at http://www.nist.gov/speech/tests/std (2006)
24. Amir, A., Efrat, A., Srinivassan, S.: Advances in phonetic word spotting. Proc. of CIKM, 580–582 (2001)
25. Thambiratnam, K., Sridharan, S.: Rapid yet accurate speech indexing using dynamic match lattice spotting. IEEE Trans. on Audio and Speech Process., 15(1), 346–357 (2007)
26. Tejedor, J., Wang, D., Frankel, J., King. S, Colás, J.: A comparison of grapheme and phoneme-based units for spanish spoken term detection. Speech Communication, 50(11–12), 980–991 (2008)
27. Wang, D., Frankel, J., Tejedor, J., Colás, J.: Comparison of phone and grapheme-based spoken term detection. Proc. of ICASSP, 4969–4972 (2008)
28. Akbacak, M., Vergyri, D., Stolcke, A.: Open-vocabulary spoken term detection using graphone-based hybrid recognition systems. Proc. of ICASSP, 5240–5243 (2008)
29. Larson, M., Eickeler, S., Kohler, J.: Supporting radio archive workflows with vocabulary independent spoken keyword search. Proc. of SSCS-SIGIR, (2007)
30. Wang, D., Tejedor, J., Frankel, J., King, S., Colás, J.: Posterior-based confidence measures for spoken term detection. Proc. of ICASSP, 4889-4892 (2009)
31. Wang, D., King, S., Frankel, J., Bell, P.: Term-dependent confidence for out-of-vocabulary term detection. Proc. of Interspeech, 2139–2142 (2009)
32. Szoke, I., Fapso, M., Burget, L., Cernocky, J.: Hybrid word-subword decoding for spoken term detection. Proc. of SSCS-SIGIR, (2008)
33. Vergyri, D., Shafran, I., Stolcke, A., Gadde, R.R., Akbacak, M., Roark, B., Wang, W.: The SRI/OGI 2006 spoken term detection system. Proc. of Interspeech, 2393–2396 (2007)
34. Yu, P., Seide, F.: A hybrid word/phoneme-based approach for improved vocabulary-independent search in spontaneous speech. Proc. of ICSLP, 635–643 (2004)
35. Miller, D.H.R., Kleber, M., Lin Kao, C., Kimball, O., Colthurst, T., Lowe, S.A., Schwartz, R.M., Gish, H.: Rapid and accurate spoken term detection. Proc. of Interspeech, 314–317 (2007)
36. Iwata, K., Shinoda, K., Furui, S.: Robust spoken term detection using combination of phone-based and word-based recognition. Proc. of Interspeech, 2195–2198 (2008)
37. Meng, S., Yu, P., Liu, J., Seide, F.: Fusing multiple systems into a compact lattice index for Chinese spoken term detection. Proc. of ICASSP, 4345–4348 (2008)

Recuperación de Información Geográfica basada en una Descripción Semántica del Espacio*

Nieves R. Brisaboa, Miguel R. Luaces, Diego Seco

Laboratorio de Bases de Datos, Universidade da Coruña
Campus de Elviña, 15071, A Coruña, España
{brisaboa,luaces,dseco}@udc.es

Resumen La recuperación de información geográfica constituye un novedoso campo de investigación surgido unos pocos años atrás para atender una incipiente demanda de los usuarios consistente en recuperar información relevante no sólo en cuanto a su contenido textual sino también en cuanto a su referente geográfico. La naturaleza espacial de los referentes geográficos motiva el tener en cuenta las características propias de este tipo de información que han sido muy estudiadas en el campo de los sistemas de información geográfica.
En este artículo presentamos las diferentes alternativas existentes en la literatura para realizar el proceso de recuperación de información geográfica. Además, realizamos un análisis en el que mostramos que existen distintos tipos de consultas que no pueden ser resueltas por estas alternativas. Finalmente, describimos una alternativa que incorpora información semántica al proceso y que permite recuperar documentos relevantes no sólo en cuanto a su contenido textual (empleando para ello un índice invertido clásico) sino también en cuanto a su referente geográfico (mediante una descripción semántica del espacio indexado). Esta alternativa abre un nuevo camino de investigación que incrementa de forma cualitativa las consultas que se pueden realizar.

Key words: Recuperación de información geográfica, índice espacio-textual, relevancia espacial.

1. Introducción

La enorme cantidad de repositorios de información digital disponibles en los últimos años (incluyendo la Web, bases de datos documentales, bibliotecas digitales, etc.) han convertido a la recuperación de información (IR) [1] en una de las áreas de investigación más activas dentro de la informática. Una de las demandas de los usuarios de estos repositorios que no se resuelve de manera adecuada dentro de la IR es la explotación de la información geográfica contenida en los mismos. A pesar de que la información más común en la mayoría de

* Este trabajo ha sido financiado parcialmente por el Ministerio de Educación y Ciencia (PGE y FEDER) ref. TIN2009-14560-C03-02 y por la Xunta de Galicia ref. 08SIN009CT.

los repositorios es de tipo textual, es posible encontrar con bastante frecuencia referencias geográficas en el texto de los documentos que permiten asignar a dichos documentos referentes geográficos (i.e. zonas del espacio en las cuales son relevantes). La naturaleza espacial de estos referentes geográficos les proporciona unas características especiales bien conocidas en el campo de los sistemas de información geográfica (GIS) [2]. El explotar estas características es el objetivo fundamental de un campo de investigación, bastante joven todavía, que se ha denominado recuperación de información geográfica (GIR) [3].

La recuperación de información geográfica consiste por tanto en la explotación de repositorios de información digital mediante consultas que pueden tener naturaleza textual, geográfica o una combinación de ambas (por ejemplo, *recuperar de la web documentos sobre conferencias de computación que se celebren en España*). El enfoque tradicional de los sistemas de recuperación de información sólo resuelve este problema de manera parcial ya que se basan en la aparición de los términos buscados en el documento (siguiendo con el ejemplo, si un documento contiene los términos *conferencia* y *Madrid* pero no *España* será sólo parcialmente relevante para la consulta). Otras técnicas como la expansión de consultas mejoran estos resultados (ya que pueden buscar términos relacionados con los indicados en la consulta) pero siguen sin ofrecer una solución elegante ya que no pueden resolver otras consultas donde la parte espacial se indique mediante una ventana o región de consulta (por ejemplo, *conferencias celebradas en un radio de 100 Km. centrado en Madrid*). Además de esta limitación para resolver determinados tipos de consulta, la información geográfica también requiere de otras metáforas de consulta y presentación de la información que han sido muy estudiadas en el campo de los GIS (por ejemplo, las interfaces de usuario en este tipo de sistemas proporcionan herramientas muy amigables para el dibujo de ventanas o regiones de consulta y el posterior marcado de los lugares relevantes).

En este artículo revisamos el estado del arte de este novedoso campo de investigación centrándonos en la parte de indexación, resolución de consultas y medidas de evaluación. Además, describimos nuestra principal aportación en el área, una estructura de indexación que refleja (y por ende permite explotar) tanto la naturaleza textual como la naturaleza geográfica de los documentos. Esta estructura de indexación está basada en el clásico índice invertido para la parte textual y en una descripción semántica del espacio donde se distribuyen los referentes geográficos de los documentos indexados para la parte geográfica. Esta alternativa permite solucionar de manera elegante la mayoría de los tipos de consulta de interés en sistemas GIR. Por último, describimos nuevas líneas de trabajo que abre esta estructura.

2. Trabajo relacionado

El proceso de recuperación de información geográfica, de manera similar al de la recuperación de información, se puede descomponer en tres etapas fundamentales: el análisis de los documentos, la indexación y la resolución de consultas.

Sin embargo dentro de cada una de estas etapas existen diferencias muy significativas. Por ejemplo, en GIR la etapa de análisis de documentos añade, además de todos los procesos en la IR clásica como lematización o borrado de *stopwords*, la localización de nombres de lugar (*geo-parsing*) y su traducción a un modelo geográfico del universo como pueden ser sus coordenadas en latitud/longitud (*geo-referenciation*). En esta revisión del estado del arte nos centraremos más en las etapas de indexación y resolución de consultas.

En la sección anterior introducimos las carencias que tiene un enfoque de IR clásico para los nuevos objetivos de la recuperación de información geográfica. Fundamentalmente estas carencias se deben a que los nombres de lugar (o los referentes geográficos) se tratan como términos de la misma naturaleza que cualquier otro término contenido en el documento. Sin embargo, estos nombres de lugar tienen una naturaleza geográfica implícita que les otorga unas características especiales bien estudiadas en el área de los GIS. De este modo, *A Coruña* y *Pontevedra* más allá de ser términos contenidos en un documento representan localizaciones reales que se pueden representar en un mapa y que además tienen ciertas características como las de ser provincias adyacentes, estar contenidas en una misma comunidad autónoma (y consecuentemente en el mismo país) o ser adyacentes a otra provincia como *Lugo*. Todas estas características son imposibles de capturar empleando únicamente el clásico índice invertido e incluso el empleo de técnicas más complicadas como la expansión de consultas no proporciona una solución elegante al problema.

Por otra parte, se podría pensar en crear sistemas GIR que empleen un índice invertido [4] (o cualquier otra estructura de indexación textual) para resolver la parte textual de las consultas y un R-tree [5] (o alguno de los muchos otros métodos de acceso espacial [6] existentes) para resolver la parte geográfica. Bajo esta idea surgieron las primeras aproximaciones para realizar recuperación de información geográfica.

Los primeros trabajos surgieron del proyecto SPIRIT [7] y se basan en la combinación de una estructura de *grid* [8] con un índice invertido. La estructura *grid* es uno de los índices espaciales más sencillos. Esta estructura divide el espacio en celdas y cada punto se asocia a la celda en la que está contenido. Los autores de este trabajo realizaron pruebas combinando el *grid* y el índice invertido en una única estructura y también manteniéndolos por separado. Su conclusión más importante es que manteniendo ambos índices separados se consigue un menor coste de almacenamiento aunque, por contra, puede implicar mayores tiempos de respuesta. Además, tener separados los índices tiene ventajas en cuanto a la modularidad, facilidad de implementación y facilidad de mantenimiento. Sus resultados también muestran que los métodos propuestos son capaces de competir en términos de velocidad y coste de almacenamiento con estructuras de indexación textual clásicas. Aunque la estructura propuesta es muy sencilla, y ya se han propuesto otras que la superan tanto en velocidad como en coste de almacenamiento, este trabajo es muy relevante ya que ha establecido una de las características distintivas de todas las propuestas posteriores. Dicha característica establece la distinción entre estructuras híbridas, que combinan los índices

textual y espacial en una única estructura, y estructuras de doble índice, que los mantienen por separado.

Trabajos más recientes, como [9,10], describen las dos estrategias base de la indexación en sistemas GIR teniendo en cuenta las propuestas del proyecto SPIRIT. Estas dos propuestas se nombran como *Text-First* y *Geo-First*. A nivel general, ambos algoritmos asumen la existencia de un índice espacial y de un índice textual, y emplean la misma estrategia para resolver las consultas: primero se emplea un índice para filtrar los documentos (el índice textual en el caso de *Text-First* y el índice espacial en el caso de *Geo-First*); el conjunto de documentos resultante se ordena por sus identificadores y posteriormente se filtra usando el otro índice (el índice espacial en el caso de *Text-First* y el índice textual en el caso de *Geo-First*). Estos nombres se pueden emplear también para estructuras híbridas. De tal modo que si la estructura emplea primero el índice textual es de tipo *Text-First* y si la estructura emplea primero el índice espacial es de tipo *Geo-First*. En la figura 1 mostramos las tres propuestas básicas de la indexación en recuperación de información geográfica. Las tres estructuras emplean un índice textual (sombreado en la figura) y un índice espacial (sin sombrear). La estructura de la izquierda es de doble índice, ya que mantiene por separado un índice textual y un índice espacial, mientras que las otras dos son estructuras híbridas. La estructura del centro pertenece a la categoría de *Geo-First*, ya que se accede primero al índice espacial, y la estructura de la derecha a la de *Text-First*, ya que se accede primero al índice textual.

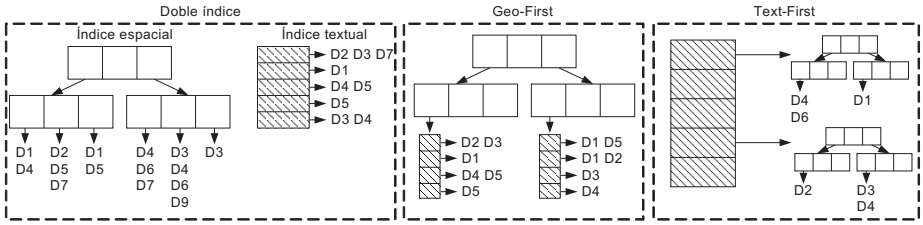


Figura 1. Estructuras de indexación básicas en sistemas GIR.

En [11] los autores proponen emplear un índice invertido y un R-tree (el ejemplo paradigmático y sin duda el método de acceso espacial más empleado). Realizan pruebas combinándolos de las tres formas que acabamos de describir y en sus experimentos concluyen que mantener por separado los índices es menos eficiente (esta misma conclusión había sido obtenida en [7]) y que sus estructuras son más eficientes que las que emplean la estructura de *grid*.

En [10], los autores comparan tres estructuras que combinan un índice invertido con la estructura *grid*, con el R-tree y con curvas de llenado del espacio [12,13]. Las curvas de llenado del espacio se basan en el almacenamiento de los objetos espaciales en un orden determinado por la forma de la curva de llenado. La conclusión de los autores es que la estructura que emplea estas curvas de llenado del espacio mejora el rendimiento de las otras aproximaciones.

Finalmente, en el proyecto STEWARD [14], los autores proponen emplear una estructura que mantenga por separado un índice invertido y un *Quad-tree*

[15]. El Quad-tree es una estructura similar al *grid* (ambas son dirigidas por el espacio y no por los objetos a indexar), que va dividiendo el espacio en cuadrantes hasta que los objetos se pueden almacenar en una página de disco. Además, en este trabajo se propone el empleo de un optimizador de consultas que decida si emplear primero el índice textual o el índice espacial en función de la previsión de cuál va a devolver menos resultados. Para poder emplear este optimizador de consultas el sistema debe almacenar estadísticas que permitan realizar la estimación del número de documentos resultante de una búsqueda de términos clave particular o de una ventana de consulta espacial determinada.

En cuanto al trabajo de otros grupos españoles, en [16] se describen los sistemas desarrollados por tres de los principales grupos del país y un mecanismo para fusionar sus resultados mejorando la eficacia individual de cada uno.

Un inconveniente de la mayoría de estas aproximaciones es que las estructuras empleadas en la parte espacial no tienen en consideración la jerarquía del espacio geográfico. Es decir, los nodos internos en las estructuras de indexación espacial no tienen significado en el mundo real (son significativos únicamente para la propia estructura). Por ejemplo, supongamos que queremos construir un índice para una colección de países, regiones y ciudades. Estos objetos se encuentran estructurados en una relación topológica de contenido; es decir, una ciudad se encuentra contenida en una región que a su vez está contenida en un país. Si construimos un R-tree (o cualquier otro método de acceso espacial) los nodos internos de la estructura no representan regiones ni países y, por tanto, la jerarquía del espacio no se mantiene en el índice. No es posible asociar ningún tipo de información con el nodo de una región y que las ciudades que pertenecen a esa región hereden automáticamente esa información ya que no hay relación de ningún tipo entre una región y sus ciudades en la estructura de indexación.

Una estructura que puede describir adecuadamente las características específicas del espacio geográfico es una ontología [17] (i.e. una especificación explícita y formal de una conceptualización compartida). Una ontología proporciona un vocabulario de clases y relaciones para describir un ámbito determinado. En [18], se propone un método para el mantenimiento efectivo de ontologías con muchos datos espaciales usando un índice espacial para mejorar la eficiencia de las consultas espaciales. Además, en [19,20] los autores describen cómo se emplean ontologías en tareas de expansión de los términos de las consultas, en la elaboración de rankings de relevancia y en la anotación de recursos web en el proyecto SPIRIT. Nuestra principal aportación al campo consiste en una estructura de indexación que para la parte espacial se basa en una descripción ontológica del espacio geográfico [21]. En la siguiente sección describimos en más detalle la ontología y la estructura de indexación basada en ella.

3. Estructura de indexación ontológica

Antes de desarrollar nuestra estructura hemos definido una ontología espacial, accesible en la URL http://lbd.udc.es/ontologies/spatialrelations, empleando la especie OWL-DL de OWL [22]. Las clases OWL se pueden interpre-

tar como conjuntos que contienen individuos (también conocidos como instancias). A su vez, estos individuos se pueden considerar como instancias de clases. Nuestra ontología describe ocho clases de interés: *SpatialThing*, *GeographicalThing*, *GeographicalRegion*, *GeopoliticalEntity*, *PopulatedPlace*, *Region*, *Country* y *Continent*. Además, existen relaciones jerárquicas entre *SpatialThing*, *GeographicalThing*, *GeographicalRegion* y *GeopoliticalEntity* ya que *GeopoliticalEntity* es subclase de *GeographicalRegion*, *GeographicalRegion* es subclase de *GeographicalThing* y *GeographicalThing* es subclase de *SpatialThing*.

Es decir, estas cuatro clases están organizadas en una jerarquía de especialización superclase – subclase, también conocida como *taxonomía*. Las subclases especializan (están *subsumidas por*) sus superclases *GeopoliticalEntity* tiene cuatro subclases: *PopulatedPlace*, *Country*, *Continent* y *Region*, y todos los individuos son miembros de esas subclases. Estas cuatro subclases tienen necesariamente una condición de aserción relativa a sus relaciones con cada una de las otras. Están conectadas por la propiedad *spatiallyContainedBy* que describe la existencia de una relación de contenido espacial entre ellas. Por ejemplo, todos los individuos de la clase *PoulatedPlace* están espacialmente contenidos (*spatiallyContainedBy*) en individuos de la clase *Region* (esto se describe en OWL como *PopulatedPlace spatiallyContainedBy only (AllValuesFrom) Region*).

La formalización de la ontología nos permite la definición de una estructura de indexación basada en ella. Por tanto, la estructura que proponemos es un árbol con cuatro niveles, cada uno de ellos correspondiente a una de las subclases de *GeopoliticalEntity*. El nivel superior del árbol contiene un nodo por cada una de las instancias de la clase *Continent*. A su vez, cada nodo en ese nivel referencia las instancias de la clase *Country* que están conectadas por medio de la relación de contenido espacial (*spatiallyContainedBy*). Los niveles correspondientes a *Region* y a *PopulatedPlace* los construimos empleando la misma estrategia. Es decir, la estructura del árbol sigue la taxonomía de la ontología. La figura 2 muestra la estructura de indexación espacial construida con las instancias de nuestra ontología.

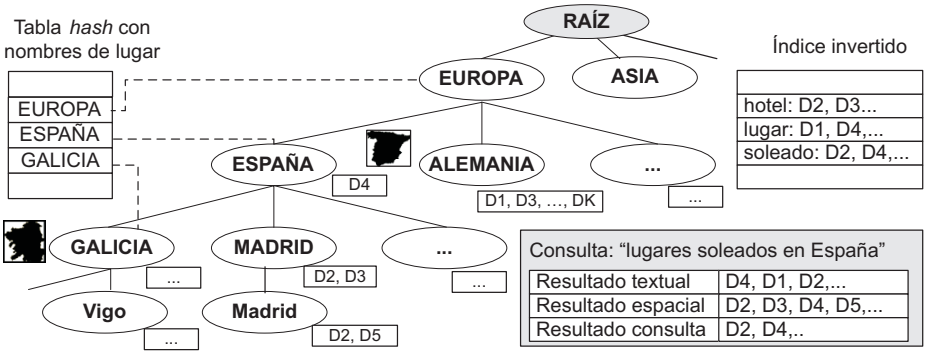


Figura 2. Ejemplo de la estructura de indexación.

Cada nodo contiene la siguiente información, además de la lista de nodos hijo que están contenidos espacialmente por él (*spatiallyContainedBy*): (i) la palabra clave (un nombre de lugar), (ii) el MBR de la geometría (i.e., el rectángulo más pequeño que la contiene) que representa ese lugar y (iii) una lista con los identificadores de los documentos que incluyen referencias geográficas a ese lugar. Además, cada nodo contiene un R-tree que mejora el rendimiento del acceso desde un nodo a sus hijos que satisfacen una consulta.

Al ser una estructura en forma de árbol y almacenar los MBRs en cada nodo, podemos usar la estructura para resolver consultas espaciales empleando el mismo algoritmo que se utiliza con las estructuras de indexación espacial clásicas. Dicho algoritmo consiste en descender a través de la estructura descartando aquellas ramas del árbol que no intersecan con la ventana de consulta. Los nodos del árbol que continúan tras el descenso constituyen el resultado de la consulta.

La ventaja principal de esta estructura de indexación espacial sobre otras alternativas es que los nodos intermedios de la estructura tienen significado en el espacio geográfico y pueden tener información adicional asociada. Por ejemplo, podemos asociar una lista de documentos que referencien a un país determinado y usar esa lista de documentos para resolver consultas que combinen una parte textual y una parte espacial. Además, dado que existe una relación superclase – subclase entre niveles, los niveles inferiores pueden heredar las propiedades asociadas con sus niveles superiores. En particular, los documentos asociados con un nodo de la estructura también se refieren a todos los nodos en el subárbol que parte de él. Esto permite que nuestra estructura de indexación puede realizar fácilmente expansión de los términos de consulta sobre referencias geográficas. Consideremos la consulta "*recuperar todos los documentos que se refieran a* España". El índice de nombres de lugar se emplea para localizar el nodo interno que representa el objeto geográfico correspondiente a *España*. Entonces, todos los documentos asociados con este nodo forman parte del resultado de la consulta. Sin embargo, todos los hijos de este nodo son objetos geográficos que están contenidos en *España* (por ejemplo, la ciudad de *Madrid*). De este modo, todos los documentos referenciados por el subárbol forman también parte del resultado de la consulta. La consecuencia es que la estructura de indexación se ha empleado para expandir la consulta porque el resultado contiene no sólo aquellos documentos que incluyen el término *España*, sino también aquellos documentos que incluyen el nombre de un objeto geográfico contenido en *España* (por ejemplo, todas las ciudades y regiones de *España*).

Otra ventaja es que la estructura es general en el sentido de que la ontología del espacio geográfico se puede adaptar para cada aplicación en particular. Por ejemplo, si una aplicación en concreto usa un área restringida del espacio geográfico donde las clases *Continent* y *Country* no son necesarias pero, en cambio, son necesarias las clases *Province*, *Municipality*, *City* y *Suburb*, podemos definir una ontología del espacio diferente y basar la estructura de indexación en ella ya que la relación *spatiallyContainedBy* continúa siendo válida entre las clases. Finalmente, podemos definir relaciones espaciales adicionales en la ontología, como

por ejemplo la de adyacencia (*spatiallyAdjacent*), y mantener esas relaciones en la estructura de indexación para mejorar las capacidades de consulta del sistema.

4. Consultas y relevancia

Una de las características más importantes de toda estructura de indexación es el tipo de consultas que permite resolver. En nuestra opinión los tipos de consulta más relevantes en un sistema de recuperación de información geográfica son los siguientes:

- *Consultas textuales puras.* Estas son consultas del tipo "*recuperar todos los documentos donde aparezcan las palabras* hotel *y* mar". Son las consultas típicas del campo de la recuperación de información. En nuestra estructura se resuelven empleando el índice invertido que forma parte de la misma y la relevancia se puede calcular empleando alguna de las múltiples variantes de la conocida fórmula $tf \times idf$.
- *Consultas espaciales puras.* Un ejemplo de este tipo de consultas es "*recuperar todos los documentos que se refieran a la siguiente área geográfica*". El área geográfica en la consulta puede ser un punto, una ventana de consulta o un objeto complejo como un polígono. Estas consultas, procedentes del campo de los GIS, adquieren un nuevo enfoque en los sistemas GIR al definir el concepto de relevancia espacial. En estos sistemas, los objetos en el resultado pertenecen al mismo con un valor de relevancia (es decir, no se hace simplemente la distinción entre objetos que pertenecen al resultado y objetos que no lo hacen). Dado que la estructura espacial que proponemos almacena en cada nodo el MBR del objeto geográfico correspondiente es posible resolver estas consultas empleando el clásico algoritmo de estructuras de indexación espacial descrito en la sección anterior. Además, podemos calcular la relevancia espacial de un documento teniendo en cuenta (mediante una función de agregación como el máximo) las relevancias individuales de cada uno de los referentes geográficos citados en el documento con respecto a la consulta. La ecuación 1 permite calcular la relevancia de un documento d para la consulta q debida al lugar l. Intuitivamente esta fórmula combina (ponderando mediante pesos) la relevancia debida al área de solape entre el MBR del lugar y la ventana de consulta, y la debida a la distancia al centro del foco de atención de la consulta. Además, toda la relevancia está ponderada por la importancia intrínseca del lugar (un parámetro precalculado que permite definir por ejemplo que *a priori* un documento que mencione *Londres* se refiere a la ciudad del *Reino Unido* y no a la de *Canadá*).

$$relevancia_{q,d,l} = \frac{w_{dc} * dc_{q,l} + w_{as} * as_{q,l}}{importancia} \quad (1)$$

- *Consultas textuales sobre un área geográfica.* En este caso se proporciona un área geográfica de interés junto con el conjunto de palabras. Un ejemplo de este tipo de consultas es "*recuperar todos los documentos con la palabra* hotel

que se refieren a la siguiente área geográfica". Al igual que en las consultas espaciales puras el área geográfica de la consulta puede ser un punto, una ventana de consulta o un objeto complejo. Dado que este tipo de consultas combina los dos anteriores el algoritmo más sencillo para resolverlas consiste en resolver la parte textual por un lado, la parte espacial por el otro y combinar los resultados de ambas subconsultas. En la bibliografía se pueden encontrar distintas fórmulas para combinar la relevancia textual de un documento con su relevancia espacial pero sin duda la suma ponderada de ambas es el método más sencillo y también el más empleado para dicho propósito.

- *Consultas textuales con nombres de lugar.* En este tipo de consultas algunos de los términos son nombres de lugar. Por ejemplo, "*recuperar todos los documentos con la palabra* hotel *referidos a España*". Estas consultas, que se realizan frecuentemente en sistemas de recuperación de información clásicos, alcanzan una nueva dimensión es sistemas GIR al poder aprovechar las características de la parte espacial de la consulta. El algoritmo para resolver estas consultas consiste en detectar los nombres de lugar mencionados en la consulta y traducirlos, mediante la tabla con nombres de lugar, a nodos internos de la estructura. Todos los documentos referenciados en dichos nodos son relevantes para la consulta pero también lo son todos los documentos referenciados en los subárboles que comienzan en ellos. Para calcular la relevancia espacial en este caso se debe tener en cuenta la profundidad a la que se encuentra cada nodo dentro del subárbol correspondiente ponderando siempre el valor por la importancia intrínseca del nodo.

Aunque estos son los tipos de consulta básicos y proporcionan una buena aproximación a cualquier otro tipo de consulta en sistemas GIR, es posible tener en cuenta otros tipos de consulta más específicos en la parte espacial. A continuación describimos algunos de ellos y esbozamos los algoritmos o las modificaciones que se deben realizar sobre la estructura para resolverlas.

- *Documentos geo-referenciados en los k-lugares más cercanos al de consulta.* Esta consulta, conocida como k-NN en métodos de acceso espacial (los k vecinos más próximos), se puede resolver en nuestra estructura mediante un procedimiento de refinamiento empezando a buscar con una ventana de consulta pequeña que se va expandiendo hasta llegar al cupo de los k vecinos. Sin embargo, dada la naturaleza semántica de nuestra estructura podemos reflejar más relaciones topológicas como puede ser la de adyacencia que nos ayuden en la resolución de la consulta. Otras relaciones como la de proximidad podrían funcionar mejor para este caso determinado aunque su definición es más subjetiva y aportan menos claridad al modelo topológico que define nuestra estructura (la relación topológica de adyacencia es mucho más común). Además, la relación de adyacencia nos permite resolver también el siguiente tipo de consultas bastante habitual en GIS cuando se consideran modelos topológicos.
- *Documentos geo-referenciados en lugares adyacentes al de consulta (o en el de consulta).* Un claro ejemplo de este tipo de consultas consiste en recuperar

todos los documentos sobre puntos de interés turístico que se encuentren en el ayuntamiento donde se aloja un turista o en los ayuntamientos adyacentes. Este tipo de consulta se pude resolver de manera sencilla en nuestra estructura si contemplamos la relación topológica de adyacencia. Es importante tener en cuenta que al añadir nuevas relaciones el árbol original se convierte en un grafo. En la siguiente sección veremos algunas consideraciones a tener en cuenta para la implementación eficiente de la estructura en ambos casos.

- *Documentos geo-referenciados en lugares que se encuentran en una cierta orientación (al norte, sur, noreste, etc.) respecto a la consulta.* Por ejemplo, cuando el mismo turista del ejemplo anterior quiere visitar a un familiar que vive al noreste de su hotel puede estar interesado en consultar todos los puntos de interés turístico que se encuentren al noreste de su hotel. Una aproximación para resolver este tipo de consultas consiste en emplear una matriz de transformación para las orientaciones más habituales y resolver una consulta de tipo región en la zona correspondiente a la orientación consultada. En el ejemplo, la matriz de transformación determinará que la esquina inferior izquierda de la región se debe posicionar en el lugar de consulta.

5. Implementación compacta

Realizar una implementación eficiente y compacta de esta estructura se puede descomponer en realizar una implementación eficiente y compacta de las dos partes fundamentales: el índice invertido y la estructura de indexación espacial. La compresión del índice invertido ha sido muy estudiada en los últimos años y es bien conocido que puede llegar a necesitar sólo un 20 % de espacio adicional sobre el texto [1]. Además, existen técnicas más novedosas de compresión que permiten representar de manera conjunta el texto y el índice invertido ocupando tan solo un 35 % del espacio necesario para el texto original [23,24].

Por otra parte, la estructura espacial consiste en un árbol donde en cada nodo sólo se necesitan las operaciones de acceder a los hijos y al padre. La implementación tradicional de este tipo de estructuras de árbol desperdicia mucho espacio al emplear punteros para permitir el acceso a los hijos (en un árbol de n nodos esto supone que cada puntero necesita $\log n$ bits). Sin embargo, existen técnicas más novedosas para representar estas estructuras de árbol empleando estructuras de datos compactas que necesitan tan sólo $2n + o(n)$ bits y permiten resolver las operaciones que necesitamos en tiempo constante. Las estructuras más conocidas son las de paréntesis balanceados (BP), LOUDS y DFUDS (en [25] puede encontrarse un buen resumen y una comparación de su rendimiento en la práctica). Por tanto, empleando dichas estructuras y almacenando la información satélite (el MBR del objeto geográfico, su topónimo, etc.) en estructuras auxiliares indexables por el identificador del nodo podemos representar la estructura completa de forma compacta. Además, podemos emplear técnicas similares a las de la compresión de las listas de ocurrencias para comprimir las listas de documentos geo-referenciados en cada nodo.

En la sección anterior describimos ciertas operaciones que se pueden resolver de manera más eficiente contemplando otras relaciones topológicas además de

la de contenido espacial. Contemplar todas estas relaciones de manera conjunta resulta en que el árbol original se convierte en un grafo y, por tanto, las representaciones que acabamos de nombrar no nos sirven. Sin embargo, las representaciones compactas de grafos han sido también muy estudiadas en los últimos años debido sobre todo a la importancia del grafo de la web o de los grafos que representan las redes sociales. Por ejemplo, el K2-tree [26], que permite representar la matriz de adyacencia de un grafo de forma comprimida, puede ser extendido fácilmente para representar todas la relaciones topológicas en nuestra estructura incluyendo además atributos en cada relación (por ejemplo una determinada orientación como *norte* o *sureste* en la relación de adyacencia).

6. Conclusiones y trabajo futuro

La recuperación de información geográfica se está consolidando como una prometedora extensión al campo de la recuperación de información debido fundamentalmente a la demanda de servicios donde poder consultar y representar información en mapas. En este artículo revisamos el estado del arte del área presentando nuestras aportaciones al mismo y describiendo los avances y las nuevas líneas de investigación que se han ido abriendo.

La principal carencia de la mayoría de las estructuras de indexación para sistemas GIR es el no tener en cuenta las características propias de la naturaleza espacial de los referentes geográficos. Este problema está presente en los índices invertidos, donde los referentes geográficos se consideran términos de la misma naturaleza que el resto de las palabras en el documento, pero también en las más recientes estructuras para GIR que contemplan sólo de manera parcial dichas características. Por ejemplo, no tienen en cuenta la naturaleza jerárquica del espacio ni otras relaciones topológicas existentes entre los referentes geográficos indexados. Nuestra principal aportación al área consiste en una estructura espacial que solventa estas carencias al estar basada en una ontología del espacio geográfico que permite representar todas las relaciones topológicas.

Esta estructura nos permite resolver nuevos tipos de consulta que no serían posibles (o las soluciones serían complicadas y poco elegantes) empleando métodos de acceso espacial clásicos que no contemplan las relaciones topológicas existentes entre los elementos indexados. Una línea de trabajo interesante consiste en el estudio de las posibles relaciones topológicas que se pueden representar en nuestra estructura y cómo afectan a los tipos de consulta existentes (o si permiten resolver nuevos tipos de consultas). Además, la implementación eficiente tanto en espacio como en tiempo de esta estructura constituye un interesante reto algorítmico. Finalmente, su integración en sistemas de recuperación de información existentes constituye el principal objetivo a más largo plazo.

Referencias

1. Baeza-Yates, R., Ribeiro-Neto, B.: Modern Information Retrieval. Addison Wesley (1999)

2. Worboys, M.F.: GIS: A Computing Perspective. CRC (2004) ISBN: 0415283752.
3. Jones, C.B., Purves, R.S.: Geographical information retrieval. In: Encyclopedia of Database Systems. (2009) 1227–1231
4. Zobel, J., Moffat, A.: Inverted files for text search engines. ACM Comput. Surv. **38**(2) (2006) 6
5. Manolopoulos, Y., Nanopoulos, A., Papadopoulos, A.N., Theodoridis, Y.: R-Trees: Theory and Applications. Springer-Verlag New York, Inc. (2005)
6. Gaede, V., Günther, O.: Multidimensional access methods. ACM Comput. Surv. **30**(2) (1998) 170–231
7. Vaid, S., Jones, C.B., Joho, H., Sanderson, M.: Spatio-Textual Indexing for Geographical Search on the Web. In: Proc. of SSTD. (2005) 218 – 235
8. Nievergelt, J., Hinterberger, H., Sevcik, K.C.: The grid file: An adaptable, symmetric multi-key file structure. In: Proc. of the ECI Conference. (1981) 236–251
9. Martins, B., Silva, M.J., Andrade, L.: Indexing and ranking in Geo-IR systems. In: Proc. of GIR, ACM Press (2005) 31–34
10. Chen, Y.Y., Suel, T., Markowetz, A.: Efficient query processing in geographic web search engines. In: Proc. of SIGMOD. (2006) 277–288
11. Zhou, Y., Xie, X., Wang, C., Gong, Y., Ma, W.Y.: Hybrid index structures for location-based web search. In: Proc. of CIKM, ACM (2005) 155–162
12. Morton, G.M.: A computer oriented geodetic data base and a new technique in file sequencing. Technical report, IBM Ltd. (1966)
13. Böhm, C., Klump, G., Kriegel, H.P.: Xz-ordering: A space-filling curve for objects with spatial extension. In: Proc. of the SSD Conference. (1999) 75–90
14. Lieberman, M.D., Samet, H., Sankaranarayanan, J., Sperling, J.: STEWARD: Architecture of a Spatio-Textual Search Engine. In: ACMGIS. (2007) 186 – 193
15. Nelson, R.C., Samet, H.: A consistent hierarchical representation for vector data. In: Proc. of the SIGGRAPH Conference. (1986) 197–206
16. Buscaldi, D., Perea-Ortega, J.M., Rosso, P., López, L.A.U., Ferrés, D., Rodríguez, H.: Geotextmess: Result fusion with fuzzy borda ranking in geographical information retrieval. In: Proc. CLEF. (2008) 867–874
17. Gruber, T.R.: A Translation Approach to Portable Ontology Specifications. Knowledge Acquisition **5**(2) (June 1993) 199 – 220
18. Dellis, E., Paliouras, G.: Management of Large Spatial Ontology Bases. In: Proc. of ODBIS. (September 2006)
19. Jones, C.B., Abdelmoty, A.I., Fu, G.: Maintaining ontologies for geographical information retrieval on the web. In: Proc. of ODBASE. (2003)
20. Fu, G., Jones, C.B., Abdelmoty, A.I.: Ontology-Based Spatial Query Expansion in Information Retrieval. In: Proc. of ODBASE. (2005) 1466 – 1482
21. Brisaboa, N.R., Luaces, M.R., Places, A.S., Seco, D.: Exploiting geographic references of documents in a geographical information retrieval system using an ontology-based index. GeoInformatica **14**(3) (2010) 307–331
22. World Wide Consortium: Owl web ontology language reference. (2008) Fecha de consulta: Marzo de 2008. Disponible en: http://www.w3.org/TR/owl-ref/.
23. Witten, I.H., Moffat, A., Bell, T.C.: Managing Gigabytes: Compressing and Indexing Documents and Images. Academic Press (1999)
24. Ziviani, N., de Moura, E.S., Navarro, G., Baeza-Yates, R.A.: Compression: A key for next-generation text retrieval systems. IEEE Computer **33**(11) (2000) 37–44
25. Arroyuelo, D., Cánovas, R., Navarro, G., Sadakane, K.: Succinct trees in practice. In: Proc. 11th ALENEX, SIAM Press (2010) 84–97
26. Brisaboa, N.R., Ladra, S., Navarro, G.: K2-trees for compact web graph representation. In: Proc. 16th SPIRE - LNCS 5721. Volume 5721. (2009) 18–30

Sobre la distribución de *scores* de documentos relevantes y no-relevantes: un problema abierto

Sergio López, Fernando Díez

Escuela Politécnica Superior, Universidad Autónoma de Madrid,
C/ Francisco Tomás y Valiente, 11, 28049, Madrid, España
{sergio.lopez, fernando.diez}@uam.es

Abstract: El trabajo que se presenta es parte de la investigación en curso sobre funciones de distribución asociadas a los *scores* de documentos relevantes y no-relevantes. Como parte del estudio, se hará una crítica del modelo Normal-Exponencial añadiendo algunas evidencias experimentales mediante contrastes de hipótesis a las ya conocidas limitaciones del modelo. El objetivo es tratar de inferir las curvas de precision-recall mejor que con el modelo Normal-Exponencial. El modelo que se propone para estimar estas distribuciones se basa en estimadores de densidad no paramétricos, concretamente, estimadores *kernel*. Se apuntan algunos resultados que hacen pensar que puede ser la dirección a seguir.

Keywords: Distribución de *scores*, Relevancia, Contraste de Hipótesis, Estimación no Paramétrica, *Precision*, *Recall*.

1 Introducción

El problema que describimos en el artículo ha sido estudiado en diferentes momentos a lo largo del último medio siglo. Desde los trabajos iniciales de Swets hasta los más recientes de Manmahta, Arampatzis, Robertson o Kanoulas, han sido numerosos los autores que han tratado de determinar cuáles son las distribuciones óptimas que mejor describen las densidades de documentos relevantes y no-relevantes en una colección.

El marco general es el de una colección de documentos (libros, páginas web, artículos, etc.) y un usuario con una necesidad de información concreta. Un sistema de recuperación de información tratará de obtener el subconjunto de documentos de la colección que mejor satisface esa consulta. El criterio para extraer ese subconjunto de la colección es establecer algún tipo de medida, denominada comúnmente *score*, que da una idea de cuánto de relevante es un documento para la consulta efectuada al motor de búsqueda. Se puede expresar como una función que toma un documento y una consulta y devuelve un número real (en la práctica, el número que devuelve comúnmente suele ser un real en R^+ o en [0, 1]).

Una vez que se dispone de los valores para todos los documentos de la colección para una consulta dada, es posible establecer un orden de mayor a menor, definiendo una suerte de clasificación o *ranking*. Son muy diversos los contextos de investigación que estudian la clasificación de los documentos con múltiples

propósitos. Una de las inquietudes más comunes es la de establecer el punto de corte que determina el umbral más allá del cual el usuario no va a mostrar interés por el resto de los documentos. Los documentos por encima del umbral eventualmente son leídos e inclusive accedidos por el usuario. Ése conjunto corresponde al conjunto de documentos que el usuario considera *relevantes*. El resto pasarían a formar el conjunto de los documentos *no relevantes* (la propiedad de relevancia es inherente a una consulta dada).

El conocimiento de la relevancia es un hecho importante intrínsecamente hablando, dado que posibilita un análisis diferente y más profundo de los datos. Entre otros aspectos, el conocimiento de la relevancia o no de un documento para un usuario permite efectuar algún tipo de aprendizaje que mejore el comportamiento del motor ante consultas futuras.

Hemos dividido el artículo en tres secciones. En la sección 2 presentamos un resumen de distintas propuestas, las más relevantes, efectuadas hasta la fecha. En la sección 3 describiremos los experimentos que hemos llevado a cabo sobre el modelo Normal-Exponencial junto con los resultados obtenidos. Finalmente expondremos las conclusiones y futuras líneas de trabajo.

2 Revisión de modelos hasta el presente

Swets [1] fue el primer autor en mencionar las distribuciones de scores en IR. El objetivo del artículo referido no era el de estudiar las distribuciones en sí mismas, sino que las empleaba para dar una medida de la efectividad de un sistema de recuperación de forma "global" en lugar de dar la efectividad como proporción de aciertos u otra proporción similar dentro de los elementos recuperados.

En su artículo Swets supone que la distribución de los documentos relevantes y no relevantes son sendas normales (con igual o distinta varianza, no parece que sea un detalle importante en su razonamiento) y utiliza como medida de efectividad la distancia entre las medias. Esto da una idea de cuánto de separadas están las distribuciones y puede dar una idea de la capacidad de clasificación del sistema: si la distancia es pequeña, las distribuciones serán parecidas y la probabilidad de "equivocarse" en la clasificación es alta. Por el contrario, si la distancia es grande las distribuciones están separadas y hay más certeza en la clasificación. Indicar que Swets no hace ninguna justificación en la elección de las distribuciones. De hecho afirma que no es un requisito estricto para su trabajo el que sean normales las distribuciones y deja como trabajo futuro el investigar más sobre la naturaleza de las mismas.

Con posterioridad este mismo autor [2], efectuó propuestas de nuevas distribuciones. En concreto optó por un modelo de dos Gaussianas de varianzas distintas y dos Exponenciales.

En la siguiente década Bookstein [3], hizo una revisión crítica de los trabajos anteriores, proponiendo como modelo dos distribuciones Poisson. Es de destacar el hecho de que esta propuesta está basada en una distribución discreta, no siendo habitual en el contexto el uso de esta clase de distribuciones.

Con el fin de siglo aparentemente se retoma el interés por el problema. Este hecho es debido, probablemente, a las capacidades de cómputo cada vez mayores y, sobre

todo, a la aparición de nuevos, y cada vez más eficientes, buscadores; así como a la posibilidad de combinar las salidas de los mismos haciendo combinación de rankings. Así, a finales de los noventa, Baumgarten [4], propone un nuevo modelo que se basa en el uso de sendas distribuciones Gamma, tanto para documentos relevantes como para los no relevantes. La propuesta deriva del principio de probabilidad de ranking de Robertson [5], a partir del cual se obtienen los valores de los scores.

Con el inicio del nuevo siglo el problema recobra interés. La conferencia SIGIR del 2001 pone de evidencia el interés existente. Buena muestra de ello son los trabajos de [6], Arampatzis [7] y de Zhang [8]. El modelo comúnmente más aceptado es el modelo Normal-Exponencial (N-E en adelante), propuesto por el mismo Manmahta.

Bennet [9], demuestra de forma empírica que el comportamiento de los documentos que se hayan entre las modas de las distribuciones de documentos relevantes y no relevantes es distinto. Según el autor se manifiesta un comportamiento asimétrico en las distribuciones que tratan de capturar las diferencias entre los documentos.

Robertson [10], estudia la relación *precision-recall* entre diferentes pares de distribuciones referidas anteriormente, comprobando que alguno de los pares comentados violan la relación inversa existente entre ambas medidas. Es decir, se hipotetiza sobre la propiedad de convexidad de las curvas ROC (*recall-fallout*) que debe presentar todo buen sistema de recuperación. En concreto, el modelo N-E carece de esta propiedad, lo cual supone, en cierto modo, una característica en contra de la bondad del modelo.

En [11] nuevamente se vuelve a poner en cuestión el modelo N-E. Kanoulas enfrenta el modelo clásico frente a una propuesta novedosa basada en una mixtura de K distribuciones Gaussianas para los documentos relevantes y una Gamma para los no relevantes (denominado modelo *GkG*). Una de las novedades de la propuesta está en la forma de estimar el número de componentes Gaussianas. El algoritmo clásico de EM (*Expectation Maximization*) necesita que *K* sea conocido para el ajuste óptimo de las distribuciones. En el caso de Kanoulas, para hacer la estimación de los parámetros en el caso de la Gamma emplean estimadores máximo verosímiles y en el caso de la combinación de Normales usan un enfoque Bayesiano Variacional, cuya descripción puede verse en [12]. La razón de usar este método es porque se tiene en cuenta el balance entre la complejidad del modelo y la bondad del ajuste, cosa que en el caso del algoritmo EM no se tiene en cuenta. Como resultado notable se establece que el número de componentes de la normal no depende ni del sistema de recuperación empleado ni de la consulta. Según demuestran los autores, el modelo *GkG* mejora la propuesta N-E a la hora de estimar las curvas de *precision-recall* usando como medidas de comparación tanto el RMSE (*Root Mean Squared Error*) como el MAE (*Mean Average Error*), medidas entre la curva real y cada una de las curvas inferidas con los modelos N-E y *GkG*.

Como podemos comprobar el problema aparentemente sigue abierto. A efectos prácticos no cabe duda que es necesario tomar partido por alguna de las propuestas. Como ya hemos indicado, la que en estos últimos años ha tenido mayor aceptación ha sido el modelo N-E de Manmahta, aunque los últimos resultados presentados por Kanoulas comprometen los de aquel y abren la puerta a nuevas propuestas.

Para concluir la sección no queremos dejar de mencionar algunos de los problemas que se mencionan en trabajos anteriores y que sin duda han de condicionar las

decisiones que se tomen al respecto de nuevos modelos o de la validación de los ya existentes. Por una parte, uno de los problemas habituales es el momento en que se trunca el *ranking*. Cuando se trunca muy pronto (digamos, por ejemplo, en los 1.000 primeros documentos) la distribución asociada puede tener una forma distinta a la correspondiente a un corte se produce más adelante (pongamos, por ejemplo, los 200.000 primeros documentos). Esta circunstancia debe hace pensar en el modelado de una distribución más flexible.

Por otra parte hemos observado que hay trabajos que explican la forma de la distribución de los *scores* cuando la consulta consta de un término. Cuando la consulta consta de varios términos, la contribución de cada término al *score* final del documento es distinta para cada uno. Haciendo un promedio de todos esos efectos es posible que se produzca un efecto 'colina' en la distribución de scores, quizá con diferentes pendientes a ambos lados (lo cual puede hacer pensar en una mixtura de exponenciales).

Finalmente, las funciones de *score* suelen descomponerse en sumas de *score* por términos, lo cual induce a pensar en algún tipo de comportamiento promedio.

3 Revisión del modelo Normal-Exponencial

Tal y como hemos explicado en el epígrafe anterior, en estos últimos años se han publicado resultados que van en contra, según distintos aspectos, del modelo N-E de Manmahta. Una constante en todos los artículos que hemos analizado está relacionada con la carencia de estudios estadísticos relativos tanto al modelo N-E como a los modelos propuestos, aunque hay varios autores que señalan algunas de las limitaciones teóricas y prácticas del modelo así como algunas correcciones [10] [13]. Únicamente Kanoulas efectúa un contraste de bondad de ajuste haciendo uso del contraste de Kolmogorov-Smirnov.

Para llevar a cabo la experimentación hemos empleado el entorno *Terrier*, el cual efectúa el proceso de búsqueda de documentos en una colección. En concreto hemos trabajado con las puntuaciones de los documentos asignadas por los motores de búsqueda *BM25*, *Hiemstra_LM*, *PL_2* y *TF_IDF*. El objetivo final es poder inferir las curvas de *precisión-recall* tanto con el modelo N-E clásico como con una nueva propuesta de modelo, el cual pensamos que es necesario efectuar a la vista de los resultados que hemos obtenido del análisis del modelo clásico y que vamos a describir en las siguientes secciones del presente epígrafe.

3.1 Ajuste no paramétrico de las distribuciones de documentos relevantes y no relevantes

En este epígrafe vamos a mostrar algunos ejemplos en los que parece razonable pensar que el modelo N-E no es la mejor opción para representar la distribución de los documentos. Para ello, haremos una estimación de la distribución mediante un estimador no paramétrico de la densidad mediante *kernels* gaussianos. También se podría hacer una estimación mediante, por ejemplo, un histograma. Sin embargo esta técnica tiene algunos problemas. La gráfica que se genera no es "suave". Además es

sensible a los puntos que marcan los límites de los subintervalos y al ancho de los intervalos en sí (ancho de ventana). Las técnicas de estimación no paramétrica comparten el problema del ancho del intervalo, pero no presentan las otras dos dificultades mencionadas. Para la determinación del ancho del intervalo en este caso hemos empleado el criterio de la "regla del pulgar", que tiene en cuenta la desviación estándar y el rango intercuartílico.

En la Fig. 1 se muestran las densidades no paramétricas obtenidas a partir de los scores de los documentos para 4 consultas concretas de los distintos motores (el número de la consulta figura al lado del motor). Se representa con una línea continua el ajuste realizado con la estimación no paramétrica y con una línea discontinua una Normal (o una Exponencial en la gráfica de la derecha) cuyos parámetros se han obtenido a partir de los scores de cada consulta. Se puede ver que ninguna de las distribuciones tiene la forma que uno espera de una distribución Normal o Exponencial (bi-modalidad, colas irregulares, no simétricas). En la parte inferior de cada gráfica se especifica el tamaño muestral y el ancho de ventana o *bandwidth*. Como se aprecia, en el caso de los documentos relevantes (gráfica izquierda) los tamaños muestrales no son muy grandes, lo cual dificulta en mayor medida hacer estimaciones precisas. En esta figura se muestran sólo algunas consultas a modo de ejemplo, pero estos efectos se producen en otras muchas, poniendo de manifiesto la dificultad de trabajar con modelos que sean válidos para la totalidad de las mismas.

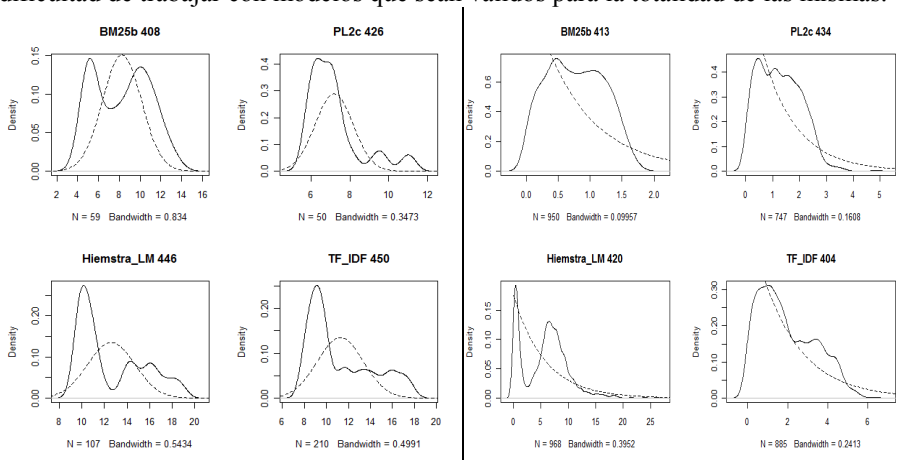


Fig. 1. Ejemplos de distribuciones para algunas consultas particulares

3.2 Contraste de hipótesis sobre las distribuciones N-E

En el epígrafe anterior hemos visto que los ajustes de los datos a las distribuciones Normal y Exponencial son muy deficientes para algunas consultas. Es obvio que cuando se trata de ajustar una determinada distribución de probabilidad a una muestra aleatoria, una de las medidas imprescindibles que es necesario efectuar, a nuestro juicio, es la del nivel de aceptación del ajuste. Por tanto y de cara a la revisión del modelo N-E planteamos dos tipos de contrastes de hipótesis distintos: los contrastes de Kolmogorov-Smirnov y de Shapiro-Wilk. La gran diferencia entre ambos contrastes radica, desde un punto de vista general, en la especificidad de los mismos.

En el caso del contraste de Kolmogorov-Smirnov (K-S), la hipótesis nula establece que la muestra proviene de una distribución de probabilidad cualquiera *F(x)*. En nuestro caso efectuamos los test de K-S sobre los documentos relevantes empleando la distribución Normal y sobre los documentos no relevantes contrastando la Exponencial.

Por su parte, el contraste de Shapiro-Wilk es un contraste específico de normalidad. Es decir, la hipótesis nula establece que los datos de la muestra han de proceder estrictamente hablando de una distribución Normal. Por tanto no es posible usar dicho contraste para el caso de la Exponencial en los documentos no relevantes.

Caso de los documentos relevantes

Para el caso de los documentos relevantes el contraste K-S se efectúa sobre una distribución Normal cuyos parámetros se han obtenido con los estimadores de máxima verosimilitud sobre los datos de la muestra. La Fig. 2 muestra el porcentaje de rechazos correspondientes a cada uno de los motores con dos niveles de confianza (alpha) 0.05 y 0.01.

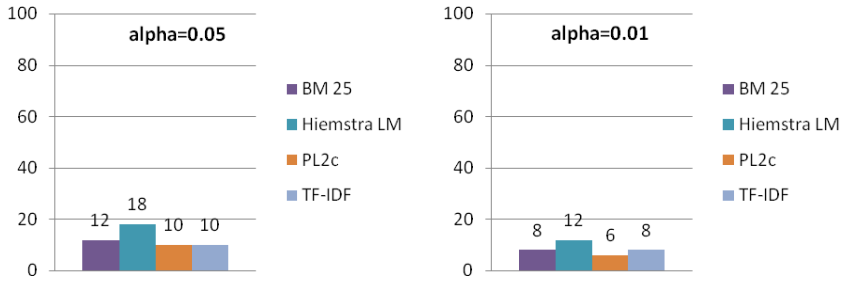


Fig. 2. Porcentaje de rechazos con K-S para relevantes

Se comprueba que, dependiendo del motor, y con el nivel de confianza 0.05, el porcentaje de rechazos del test no muestra una evidencia en contra de la hipótesis del modelo Normal para los documentos relevantes, siendo estos resultados compatibles con los presentados por Kanoulas. De todos modos nos planteamos efectuar alguna prueba adicional. Como hemos explicado al inicio del epígrafe 3.2, creemos necesario efectuar un contraste complementario al de K-S que sea específico sobre la normalidad de la muestra, decidimos usar el contraste de Shapiro-Wilks, que es un mejor test de cara a determinar si una distribución no es normal [14]. Así pues hemos efectuado el contraste de Shapiro-Wilk para normalidad obteniendo los resultados que se muestran en la Fig. 3.

Los resultados obtenidos son notablemente peores que para el contraste no específico. Se aprecia que, en el mejor de los casos, se alcanza un porcentaje de rechazo del 38% (para PL2c con alfa 0.01), pero resultan muy significativos los porcentajes por encima del 50% en todos los motores con alpha 0.05. Obviamente el modelo de normalidad está, a nuestro juicio, en cuestión y será parte de futuras pruebas que describiremos en la sección de conclusiones.

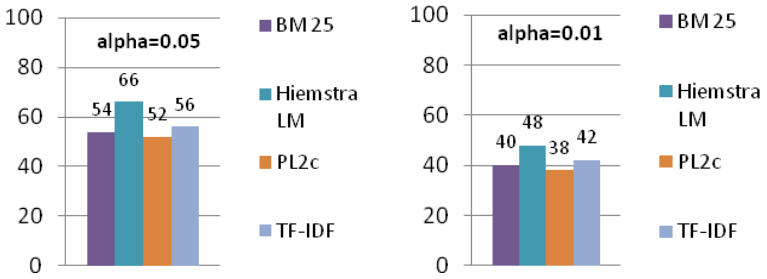


Fig. 3. Porcentaje de rechazos con S-W para relevantes

Caso de los documentos no relevantes

En el caso de los documentos no relevantes, ya hemos indicado más arriba que el contraste K-S se efectúa sobre una distribución Exponencial. En este caso los parámetros se han obtenido, como en el caso anterior, mediante los estimadores de máxima verosimilitud sobre los datos de la muestra. La Fig. 4 muestra el porcentaje de rechazos correspondientes a cada uno de los motores con los dos niveles de confianza habituales 0.05 y 0.01. Los resultados no muestran lugar a dudas salvo, quizás, el motor *PL2c*. Por tanto no parece una elección muy acertada considerar la distribución Exponencial como la distribución asociada a la muestra de documentos no relevantes, a la vista de los porcentajes tan elevados de rechazo.

Pensamos que cabe la posibilidad de efectuar nuevas propuestas de distribuciones que sean capaces de ajustar el modelo subyacente de forma más precisa.

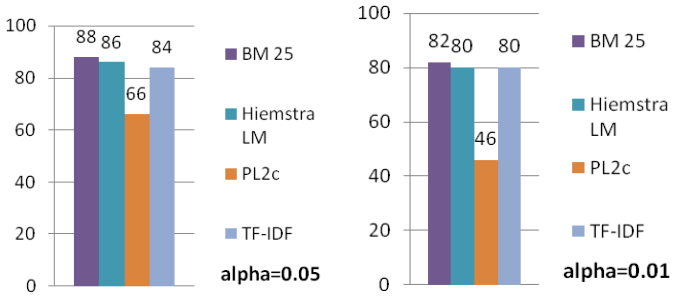


Fig. 4. Porcentaje de rechazos con K-S para no relevantes

Ya hemos visto en la revisión efectuada en la sección 2 que a lo largo de la historia se han efectuado propuestas muy distintas. En nuestro caso estamos realizando diferentes pruebas con distribuciones que hasta el presente no han sido consideradas en la literatura, aunque en el momento de efectuar la redacción de este artículo no podemos ofrecer aún una propuesta concreta que sea suficientemente satisfactoria y con una calidad contrastada.

3.3 Estimaciones de las curvas *P-R*

Tras el estudio de las distribuciones usamos, con el modelo no paramétrico, la metodología planteada por Kanoulas para inferir la curva de P-R mediante las distribuciones de *scores* de los documentos relevantes y no relevantes. Así pues, decidimos calcular la curva de P-R asociada a esas distribuciones con el objetivo de inferir la curva real de P-R. Al mismo tiempo, planteamos una prueba para ver el efecto de la normalización en los *scores*, una operación habitual en los motores de búsqueda, comparando las curvas obtenidas aplicando normalización y sin aplicarla.

Antes de mostrar los resultados, vamos a aclarar brevemente cuál es la metodología empleada para obtener las curvas de P-R a partir de las distribuciones de *scores* de los documentos relevantes y no relevantes.

Las curvas de P-R se obtienen contando el número de documentos relevantes que hay entre un cierto número de documentos recuperados. Variando el número de documentos recuperados (el *recall*) se obtienen diferentes valores de documentos relevantes (la *precision*). Esa información está contenida, de alguna forma, en las distribuciones de los *scores* si identificamos la probabilidad de encontrar un documento por encima de un cierto *score* como la proporción de documentos por encima de ese mismo *score*. De esta forma se establece una expresión para trazar la curva de P-R basada en las distribuciones. Se puede encontrar una explicación detallada en [10] sobre el procedimiento a seguir. En [11] se puede encontrar una versión simplificada del método.

En primer lugar, trabajaremos con las funciones de distribución "por la derecha". Se definen de esta forma para los documentos relevantes y no relevantes:

$$F_r(s) = \int_s^{\infty} f_r(x)dx$$

Siendo f_r la función de densidad de los documentos relevantes.

Fijamos un nivel de *recall r* con el que seleccionaremos la proporción de documentos relevantes y obtenemos el *score* asociado al último documento recuperado. Usaremos ese valor para conocer la proporción de documentos no relevantes que hay por encima de ese nivel de *score*, para ello usamos la función de distribución por la derecha de los documentos no relevantes:

$$n(r) = F_n(score(r)) \qquad score(r) = F_r^{-1}(r)$$

Hasta aquí tenemos las proporciones de documentos relevantes respecto al total de relevantes y la de no relevantes respecto al total de no relevantes. Para definir la precisión las transformamos a la misma escala, multiplicando por el factor *G*, cociente entre el número de documentos no relevantes y relevantes:

$$prec(r) = \frac{r}{r + n(r) \times G}$$

Una comprobación sencilla para cerciorarnos de que la fórmula es correcta consiste en multiplicar y dividir por el número total de documentos relevantes, llamémoslo *R*:

$$prec(r) = \frac{R \times r}{R \times r + R \times n(r) \times \frac{NR}{R}} = \frac{R(r)}{R(r) + NR(r)}$$

En la fórmula de la precisión hemos sustituido *G* por su valor, siendo *NR* el número de documentos no relevantes. Si simplificamos y operamos se obtiene:

Siendo *R(r)* el número de documentos relevantes a nivel de *recall r* y *NR(r)* el correspondiente para los no relevantes.

A partir de este desarrollo se obtienen las curvas P-R que se muestran en la Fig. 5. Se muestran las gráficas de las curvas obtenidas mediante el modelo N-E, mediante estimación no paramétrica de densidades así como la curva real proporcionada por *Terrier*, la cual nos sirve como *línea de base*.

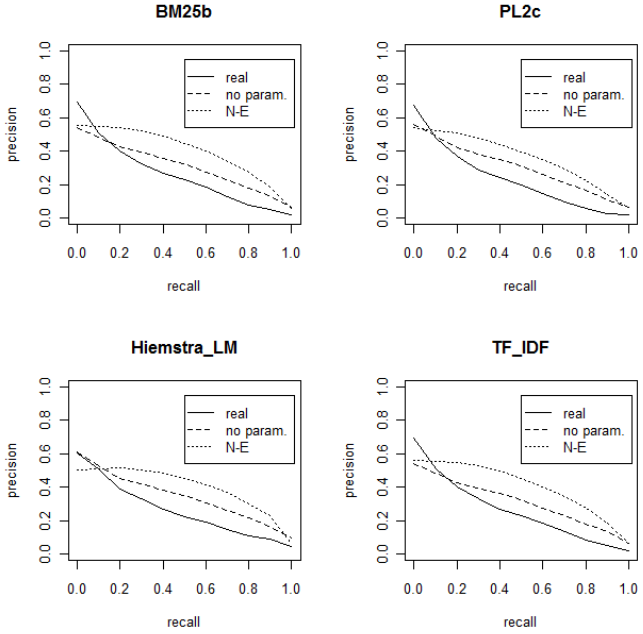


Fig. 5. Curvas de P-R. Real y estimadas

Por su parte, la Fig. 6 muestra el efecto de la normalización *minMax* sobre las curvas N-E y no paramétrica. Vemos que el efecto de la normalización no aparenta tener un gran impacto en las curvas. Este efecto, mostrado únicamente para BM25b, se conserva igualmente en el resto de los motores.

Nos parece importante destacar el hecho de que las curvas estimadas no paramétricamente están por debajo de las curvas del modelo N-E. Esta circunstancia es importante pues la interpretación que se debe hacer es diferente a la forma en que se leen las curvas de P-R determinadas por las salidas de un motor.

El objetivo que buscamos es inferir la curva de P-R real, que actúa como línea de base. Bajo este criterio las curvas obtenidas mediante la estimación no paramétrica mejoran a las obtenidas con el modelo N-E. A mayor distancia entre las curvas

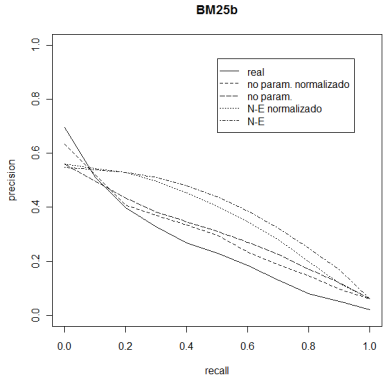


Fig. 6. Curvas de P-R con y sin normalización

estimadas y la real, más se evidencia la no adecuación de un modelo cualquiera al óptimo. En este caso el modelo no paramétrico mejora al N-E clásico.

Para medir el grado de separación entre las curvas procedemos a calcular el RMSE y MAE, entre la curva real y cada una de las curvas obtenidas con los dos modelos. Mostramos en la siguiente Tabla 1 los valores de RMSE y MAE obtenidos:

Tabla 1. Valores de MAE y RMSE para las curvas obtenidas con ambos modelos

	MAE		RMSE	
	No param.	N-E	No param.	N-E
BM25b	0.047	0.124	0.051	0.134
Hiemstra_LM	0.034	0.091	0.038	0.105
PL2c	0.061	0.118	0.064	0.128
TF-IDF	0.049	0.127	0.053	0.138

Se puede ver que en todos los casos el modelo no paramétrico se acerca más a la curva real que el modelo N-E.

4 Conclusiones

En el artículo presentado hemos efectuado una revisión crítica del modelo Normal Exponencial de Manmahta sobre las distribuciones de probabilidad asociadas a los *scores* de los documentos relevantes y no relevantes devueltos por un motor de búsqueda cualquiera.

Hemos comprobado que, si bien no existen evidencias notables en contra del modelo Normal para los documentos relevantes haciendo uso del contraste de Kolmogorov-Smirnov para la bondad de ajuste, si empleamos un contraste específico de normalidad, como es el caso del de Shapiro-Wilk, las evidencias en contra del modelo Normal son muy fuertes, alcanzando un rechazo superior al 50% para un nivel de confianza estándar de 0.05. De igual forma comprobamos que el rechazo en

el caso de la hipótesis de distribución Exponencial para el caso de los documentos no relevantes se sitúa por encima del 80% de los casos (salvo el caso particular de uno de los motores). Corroboran estos resultados las evidencias obtenidas mediante estimación no paramétrica de las densidades. Hemos empleado distintos modelos de estimación de ventana óptima (aunque en el artículo sólo hemos descrito uno) y en todos los casos se evidencian consultas que presentan aspectos muy alejados de los que cabría esperar para el caso del modelo N-E. Así, es común ver ajustes de distribuciones con múltiples modas, leptocúrticas o asimétricas (con asimetrías a ambos lados). Estas evidencias van en contra de la propuesta de un modelo como el N-E y, sospechamos, que en general pueden ser causantes de la dificultad de encontrar un único modelo que se comporte adecuadamente ante cualquier consulta y con cualquier motor de búsqueda.

Finalmente hemos comprobado de forma experimental que las curvas de P-R obtenidas mediante estimadores no paramétricos de la densidad mejoran a las correspondientes al modelo N-E clásico en el sentido de que aproximan mejor la curva de P-R real. En la Fig. 6 se puede ver que el efecto de la normalización no altera los resultados en el sentido de alterar cuál de los dos modelos es el que produce mayor error.

Actualmente estamos realizando experimentos para encontrar aplicaciones prácticas del modelo no paramétrico. Entre otras, estamos empleando las estimaciones no paramétricas de la densidad para construir un clasificador Bayesiano que permita transformar dichas densidades en probabilidades de pertenecer a una clase u otra de documentos (relevantes y no relevantes) y así poder comparar los resultados de los distintos modelos.

Otra de las tareas sería tratar de ajustar expresiones algebraicas para las densidades no paramétricas estimadas. Disponer de este tipo de expresiones nos proporcionaría determinadas ventajas tanto desde el punto de vista de la comprobación de propiedades teóricas que deben presentar las distribuciones asociadas como la posibilidad de establecer estimaciones de probabilidad de *scores* asociados a documentos con mucha mayor precisión.

5 Agradecimientos

El trabajo que se describe en este artículo ha sido subvencionado por el Ministerio de Ciencia e Innovación, Plan Nacional de I+D+i, como parte del proyecto número TIN2008-06566-C04-02. Los autores desean igualmente agradecer sus comentarios y aportaciones al profesor Antonio Cuevas del Departamento de Matemáticas de la Facultad de Ciencias de UAM.

Referencias

[1] Swets, J.A.: Information retrieval systems. Science, 141, no. 3577, 245--250, (1963)

[2] Swets, J. A.: Effectiveness of information retrieval methods. American Documentation, 20:72–89, (1969)

[3] Bookstein, A.: When the most "pertinent" document should not be retrieved—an analysis of the Swets model. Information Processing & Management, 13(6):377– 383, (1977)

[4] Baumgarten, C.: A probabilistic solution to the selection and fusion problem in distributed information retrieval. In SIGIR '99: Proceedings of the 22nd annual International ACM SIGIR conference on Research and development in information retrieval, 246–253, New York, USA, (1999).

[5] Robertson, S. E., K. Spärck Jones: Relevance weighting of search terms. Journal of the American Society for Information Science, 27(3):129–146, (1976)

[6] Manmatha, R., Rath, T., Feng, F.: Modeling score distributions for combining the outputs of search engines. In: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval, pp. 267-275. New York, USA. (2001)

[7] Arampatzis, A., van Hameran, A.: The score-distributional threshold optimization for adaptive binary classification tasks. In SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval, 285–293, New York, USA, (2001)

[8] Zhang Y., Callan, J.: Maximum likelihood estimation for filtering thresholds. In SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval, 294–302, New York, USA, (2001)

[9] Bennett, P.N.: Using asymmetric distributions to improve text classifier probability estimates. In SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval, pages 111–118, New York, USA, (2003)

[10] Robertson, S. On score distributions and relevance. In: G. Amati, C. Carpineto, & G. Romano (eds.) LNCS , vol. 4425, pp. 40--51. Springer, Heidelberg (2007).

[11] Kanoulas, E., Pavlu, V., Dai, K. & Aslam, J.: Modeling the score distributions of relevant and non-relevant documents. In Azzopardi, L. *et al.* (eds.) *ICTIR*, vol. 5766, pp. 152—163. Springer Berlin Heidelberg (2009)

[12] Bishop, C.M.: Pattern Recognition and Machine Learning, Information Science and Statistics. Springer, Heidelberg (2006)

[13] Arampatzis, A., Kamps, J. & Robertson, S.: Where to stop reading a ranked list?: threshold optimization using truncated score distributions. In SIGIR '09: Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval, 524-531. ACM, New York, USA (2009)

[14] Shapiro, S. S., Wilk, M. B. & Chen, H. J.: A comparative study of various tests for normality. Journal of the American Statistical Association 63, pp. 1343--1372 (1968)

Indexación y autoindexación comprimida de documentos como base de su procesado *

Nieves R. Brisaboa, Antonio Fariña, Susana Ladra, Ángeles S. Places, Eduardo Rodríguez

Laboratorio de Bases de Datos, Campus de Elviña s/n, A Coruña, España.
{brisaboa,fari,sladra,asplaces,erodriguezl}@udc.es

Resumen Tradicionalmente, las comunidades de investigación en Recuperación de Información (RI) y en Ingeniería Lingüística (IL) han desarrollado sus investigaciones utilizando habitualmente como base documentos no comprimidos indexados, en su caso, con índices clásicos como los índices invertidos. Por otro lado, la comunidad de investigación en estructuras de datos y algoritmos ha venido desarrollando compresores, índices y autoíndices de texto de gran potencia que, en general, son poco utilizados en las aplicaciones de Bibliotecas Digitales e Ingeniería Lingüística.

En este artículo tratamos de dar a conocer los más recientes avances en indexación y autoindexación comprimida de textos, señalando cómo su uso puede mejorar todas las tareas clásicas de RI e IL.

Además presentamos un modelo de encapsulación, que permitirá a las técnicas de RI e IL realizar el procesamiento de los documentos independientemente de su representación. La idea es que los documentos comprimidos y/o autoindexados, constituyan una capa base encapsulada con una interfaz bien definida, que permita invocar los algoritmos que, internamente, manejan las estructuras que almacenan los documentos.

1. Introducción y motivación

En los últimos años ha habido grandes avances en el ámbito de la compresión e indexación de textos en lenguaje natural. Las nuevas técnicas de compresión de textos [14,15,6] permiten, no sólo reducir su tamaño, sino que permiten procesar un texto comprimido mucho más rápido que la versión sin comprimir. En el caso de la indexación de textos, destaca la aparición de nuevas estructuras de indexación como los índices invertidos comprimidos [2,19] y la de los autoíndices, que dentro del propio índice mantienen una versión implícita del texto. Entre ellos, cabe señalar aquellos que indexan caracteres y permiten recuperar cualquier subcadena del texto indexado [16], o los orientados a palabras [4,7], más adecuados para el texto en lenguaje natural.

* Parcialmente financiado por: el "Ministerio de Ciencia e Innovación" (MICINN ref. TIN2009-14560-C03-02)

Sin embargo, estas técnicas no han sido todavía comúnmente adoptadas por otras disciplinas como la ingeniería lingüística (IL), donde generalmente se trabaja con colecciones de documentos sin comprimir, o la recuperación de información (RI) [19,22], centrada en la recuperación de documentos *relevantes* y que continúa apoyándose en estructuras de indexación tradicionales como los índices invertidos para realizar sus operaciones (cálculo de frecuencia inversa, ranking de documentos, etc.).

La motivación de este artículo es doble: por un lado, revisar el estado del arte en compresión e indexación; por otro lado, se propone un nuevo modelo que, a través de la definición de una serie de primitivas que podrán ser invocadas desde los procesos de IL o IR, permitirá aislar dichos procesos de la forma utilizada para representar la colección de documentos. La idea es permitir que dicha representación se comporte como una caja negra que permita, a través de una interfaz común, la invocación de una serie de primitivas (p. ej: obtener la frecuencia de una palabra, recuperar un trozo de la colección, etc.). De este modo, se pretende dar soporte para que las técnicas de IL y RI puedan aplicarse sobre esta nueva capa que oculta la forma en que la colección está almacenada. Es importante recalcar que pese a permitir realizar el procesamiento sobre la colección comprimida no se pretende simplemente reducir las necesidades de espacio, sino mejorar el rendimiento del sistema resultante.

En la Sección 2 se revisan técnicas de compresión adecuadas para la compresión de Bases de Datos textuales, típicamente junto a índices invertidos, que son descritos en la Sección 3. La siguiente sección se centra en los autoíndices orientados a palabras. La Sección 4 muestra el modelo de encapsulación propuesto, y finalmente presentamos nuestras conclusiones y trabajo futuro.

2. Compresión de textos orientada a Lenguaje Natural

En los últimos años se han desarrollado nuevas técnicas de compresión adecuadas para la compresión de Bases de Datos textuales, que no sólo permiten reducir su tamaño, sino también mejorar su rendimiento [15,6]. Estas técnicas deben permitir la realización de dos operaciones básicas: *i)* realizar búsquedas directas sobre el texto comprimido (buscando el patrón comprimido) que evitan la necesidad de descomprimir la colección para buscar un patrón; y *ii)* realizar descompresión aleatoria; esto es, acceder a cualquier posición del texto comprimido y comenzar la descompresión desde ese punto, sin necesidad de comenzar desde el principio.

Las técnicas de compresión clásicas como el Huffman clásico (orientado a caracteres) [12] cuyo ratio de compresión es muy pobre (60 %), o los de la familia Lempel-Ziv [20,21], que permiten buscar dentro del texto comprimido de forma eficiente, no son adecuadas para manejar colecciones de texto.

En 1989, Moffat [14] propone utilizar palabras, en lugar de caracteres como los símbolos a comprimir. Este hecho abre la puerta a la integración de indexación y compresión debido a que las palabras son los elementos básicos en ambos casos. Además, debido a que la distribución de frecuencias de las pal-

abras en un texto es mucho más sesgada que la de los caracteres, un compresor estadístico semi-estático como el Huffman basado en palabras y orientado a bits [18] es capaz de obtener ratios de compresión próximos al 25 % (frente al 60 % del Huffman clásico). Los compresores semi-estáticos realizan dos pasadas sobre el texto a comprimir: en la primera pasada, obtienen un modelo del texto (el vocabulario con las distintas palabras existentes y su frecuencia) que es utilizado para asignar códigos más cortos a las palabras más frecuentes. En la segunda pasada, sustituyen cada palabra por su código. Dado que cada palabra está representada siempre por el mismo código dentro del texto comprimido, estas técnicas semi-estáticas permiten realizar sobre el texto comprimido las mismas operaciones que pueden realizarse sobre texto plano.

En [15] se presentaron dos nuevas técnicas basadas en Huffman, pero orientadas a byte en lugar de a bit para agilizar la descompresión. La primera técnica *Plain Huffman* utiliza árboles Huffman 256-arios y obtiene ratios de compresión próximos al 30 %. La otra, *Tagged Huffman* (TH), reserva el primer bit de cada byte para marcar si un byte es el primer byte de un código o no, y aplica codificación Huffman sobre los restantes 7 bits de cada byte. La compresión obtenida está en torno al 33-34 %, pero los códigos generados de esta forma se vuelven *síncronos* permitiendo acceso aleatorio al texto comprimido, y la capacidad para realizar búsquedas sobre el texto comprimido utilizando algoritmos de la familia Boyer-Moore[3,11].

En [6] se presenta el *End-Tagged dense code* (ETDC), que mantiene el bit de marca del TH, pero en lugar de marcar el principio, marca el final de cada código. Esto permite utilizar una codificación totalmente secuencial (sin necesidad de utilizar codificación Huffman) y aprovechar todos los valores posibles de cada byte. En la práctica, el ETDC mantiene todas las propiedades del TH (sincronismo, descompresión aleatoria y búsquedas directas mediante Boyer-Moore) a la vez que obtiene ratios de compresión sólo 1 punto porcentual peores que el Plain Huffman (31 %). En [5,6], se presenta una mejora del ETDC, conocida como (s, c)-*Dense Code* (SCDC) que obtiene ratios de compresión sólo 0.2 puntos porcentuales peores que el Plain Huffman.

Los resultados mostrados en [15,6] muestran que estas técnicas son muy rápidas a la hora de comprimir (comprimen 1Gb de texto en poco más de 1 minuto en un pc convencional), siendo las más rápidas del estado del arte en cuanto a descompresión (descomprimiendo en torno a 60Mb/seg.), y permiten realizar búsquedas hasta 8 veces más rápido que sobre el texto sin comprimir.

3. Índices invertidos y compresión

Los índices invertidos son sin duda las estructuras de indexación más simples y utilizadas en recuperación de información (RI) para la indexación de texto en lenguaje natural [2,19], siendo también pieza clave en la mayoría de los motores de búsqueda actuales.

Un índice invertido es básicamente un vector de listas, donde para cada palabra o *término* del vocabulario[1], se almacena una lista que contiene las posiciones donde ésta aparece dentro de la colección de documentos indexada. Existen dos variantes principales [1,22].

La primera variante [19,22], se enfoca hacia la recuperación de documentos *relevantes* ante una consulta dada. En este caso, para cada documento se mantiene un vector, cuyas dimensiones son los términos del vocabulario, y en el que sus valores representan la relevancia de un término en dicho documento. De este modo las listas invertidas almacenan para cada término, los documentos en los que aparece, y el peso en dicho documento. Una consulta normalmente implica mezclar las listas de los términos de la consulta de modo que se recuperen los documentos que se consideren relevantes, lo que requiere combinar los pesos de dichos términos en cada documento.

Por otro lado, los índices invertidos orientados a la *recuperación de texto completo*, se centran en la recuperación de documentos en los que un término aparece. Las listas invertidas almacenan los documentos donde aparece cada término, típicamente ordenadas por orden de aparición. Las consultas que se refieren a una única palabra son resueltas con la simple obtención de la lista invertida de dicha palabra; consultas tipo *OR* implican mezclar listas; y consultas conjuntivas requieren la realización de intersección de listas [2]. Cuando se buscan frases, el índice invertido permite filtrar los documentos en los que aparecen todos los términos de la consulta (mediante intersección de sus listas), siendo necesario un posterior escaneado de dichos documentos para verificar si dicha frase se encuentra en el documento. La búsqueda de frases puede realizarse más eficientemente en el propio índice invertido si éste almacena también la posición en cada documento donde aparece cada palabra. Otras variantes como los índices invertidos con direccionamiento por bloques particionan la colección en bloques, y para cada término apuntan en qué bloques aparece. De esta forma obtienen diferentes relaciones espacio/tiempo en función del tamaño de bloque elegido.

El desarrollo de las técnicas de compresión semi-estáticas orientadas a palabras abrió el camino para que los índices invertidos pudiesen indexar texto comprimido, agilizando su funcionamiento especialmente en aquellos casos en los que las listas invertidas se mantienen en disco, o cuando el índice no almacena información posicional completa. Además, las listas invertidas, que son secuencias de números crecientes, también son altamente compresibles mediante alguna técnica de compresión de enteros (golomb-rice codes, bytecodes, etc.).

El tamaño total de un índice invertido comprimido (incluyendo el del texto comprimido) está en torno al 50-60 % del tamaño original de la colección. En índices invertidos que apunten a documentos o bloques, dicho tamaño puede reducirse hasta un 35-30 %, manteniendo la posibilidad de realizar las mismas operaciones que los índices invertidos sin comprimir. Por ejemplo, en un índice invertido que apunte a bloques, comprima las listas invertidas con golomb-codes,

[1] Hablaremos de vocabulario como el conjunto de palabras diferentes del texto. Sin embargo, también podríamos estar interesados en que el vocabulario no almacenase *stopwords*, o contuviese sólo las *raíces* o *lemas* obtenidos tras un proceso *lematización*.

y donde el texto (1Gb) esté comprimido con ETDC (tamaño (índice + texto) = 37 % del texto original), el tiempo medio para localizar una aparición de 1 palabra está en torno a $10\mu s$, y el de una frase formada por 2 palabras es próximo a $15\mu s$ [7].

4. Autoindexación

En escenarios donde el concepto de palabra no existe (p.ej. secuencias biológicas), y donde se hace necesario ser capaz de buscar cualquier subcadena del texto, los índices invertidos no resultan interesantes. Por este motivo aparecieron estructuras como los *arrays de sufijos (SA)* [13]. Los *SA* son capaces de encontrar cualquier subcadena del texto muy eficientemente (en tiempo $O(\log_2(|texto|)))$, pero pueden ocupar hasta cuatro veces más que el propio texto indexado.

Los *autoíndices* [17,16] surgen para reducir el tamaño de los *SA*. Los autoíndices son estructuras comprimidas que permiten indexar un texto al mismo tiempo que mantienen una representación implícita del mismo, evitando así la necesidad de almacenarlo aparte. Frente a los autoíndices originales, en 2008 surgen dos nuevos autoíndices que indexan palabras y están enfocados a la indexación de texto en lenguaje natural llamados *Byte Oriented Codes Wavelet-Tree (BOCWT)* [4] y *Word Compressed Suffix Array (WCSA)* [7].

Actualmente, la autoindexación constituye un campo de investigación dinámico y prometedor, donde numerosas propuestas[2] han surgido en la última década. En cualquier caso, los nuevos autoíndices siguen sin ser prácticamente utilizados en operaciones de IR [9] o de IL debido posiblemente a que han evolucionado "demasiado" rápido y no han sido desarrollados pensando en proveer ciertas funciones que estas disciplinas necesitan. El modelo propuesto en la Sección 4 pretende permitir dichas operaciones sobre cualquier estructura de indexación, entre ellas sobre los autoíndices *BOCWT* y *WCSA*.

4.1. Autoíndices para texto en lenguaje natural

Los autoíndices que indexan palabras en lugar de caracteres se benefician de tener que indexar un menor número de símbolos (en inglés la longitud media de una palabra es aproximadamente 4-5 caracteres). Este factor los hace más compactos (el mismo autoíndice orientado a caracteres puede ocupar más del doble [7]) y rápidos. A cambio, sólo pueden buscar palabras o frases, no cualquier subcadena del texto.

Los autoíndices siguen el API en `http://pizzachili.dcc.uchile.cl/api.html`. Entre otras, han de solucionar las siguientes operaciones básicas: *i) count(P)*, contar el número de apariciones de un patrón; *ii)) locate(P)*, para encontrar en qué posiciones aparece el patrón buscado y *iii) extract(ini, end)*, que recupera cualquier parte del texto original desde la posición *ini* hasta la *end*, esto es, descomprime el texto contenido por el autoíndice.

[2] Artículos relacionados y códigos fuente disponible en `http://pizzachili.dcc.uchile.cl/`

Word-Compressed Suffix Array (WCSA)

El WCSA [7] es la evolución del *word-based SA (WSA)* presentado en [8]. Dado un texto $T_{1..n}$, se obtiene el vocabulario de palabras, y se le asigna a cada palabra un identificador entero *id*. A continuación se obtiene un array T_{id} sustituyendo cada palabra de T por su *id*. De esta forma T puede ser reemplazado por T_{id} y el vocabulario de palabras (ordenado alfanuméricamente). A continuación se ordenan todos los sufijos de T_{id}, donde un sufijo i representa a la secuencia de enteros $T_{id}[i..n]$, de forma que $T_{id}[SA[i]..n] \preceq T_{id}[SA[i+1]..n]$. En la Figura 1 se muestra un SA que indexa palabras. Se puede observar que $T_{id}[SA[5]..n] = \langle 3, 2, ...\rangle$ es menor que $T_{id}[SA[6]..n] = \langle 3, 6, ...\rangle$.

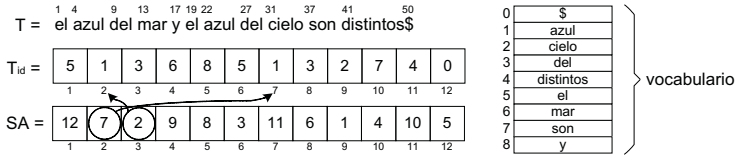


Figura 1. Estructura de un SA orientado a palabras.

La ventaja de un array de sufijos radica en la posibilidad de buscar cualquier patrón usando una simple búsqueda binaria, debido precisamente a que todos los sufijos apuntados desde SA están ordenados. Dado un patrón P, tal que P es la secuencia de ids de las palabras de una frase, el WSA puede resolver *count(P)* en $O(|P| \log_2 n)$ y *locate(P)* en tiempo $O(|P| \log_2 n + occ)$, siendo *occ* el número de veces que aparece el patrón P. Para buscar la palabra "`azul`" en la Figura 1, simplemente habría que buscar dónde aparece el $id = 1$ en T_{id}. Esas posiciones son apuntadas desde $SA[2..3]$. Esto nos indica que sólo hay 2 apariciones $(2 = 3 - 2 + 1)$, y sus localizaciones en T_{id}; esto es, las posiciones $SA[2] = 1$ y $SA[3] = 7$ de T_{id}. De nuevo, el problema del WSA es que su tamaño es similar al tamaño del texto indexado.

El *WCSA* presentado en [7] junta las ideas del WSA y aquellas mostradas en [17], para obtener un autoíndice comprimido que mantiene la funcionalidad del WSA, ocupando sólo un 35-40 % de tamaño del texto original. Junto al *WCSA*, en [7] también se presentó el *Flexible WCSA (FWCSA)*. El *FWCSA* es el primer autoíndice basado en arrays de sufijos que permite incluir operaciones típicas de la indexación de lenguaje natural como: lematización, eliminación de *stopwords*, *case folding*, etc.

Byte Oriented Codes Wavelet-Tree (BOCWT)

El Byte-Oriented Codes Wavelet Tree (BOC-WT) [4] es un autoíndice que se construye a partir de los bytes de un texto reorganizándolos en una estructura con forma de árbol. Puede aplicarse sobre el texto comprimido obtenido mediante

cualquier técnica de compresión libre de prefijo que sea estadística, semiestática, orientada a byte y a palabras. Mantiene el mismo ratio de compresión y tiempos similares de compresión y descompresión que la técnica de compresión utilizada, pero mejora drásticamente las búsquedas debido a que se consigue autoindexar el texto de forma implícita mediante la estructura de árbol.

De forma resumida, este método recoloca los bytes del texto comprimido siguiendo una estrategia de wavelet tree [10], situando los primeros bytes de cada código en el primer nivel del árbol, los segundos bytes de cada código en el segundo nivel, en la rama correspondiente en función del primer byte del código, y así sucesivamente hasta llegar al último nivel del árbol, que tendrá una profundidad igual a la longitud del código de la palabra más larga. De esta forma, se puede asociar un nodo hoja del árbol con cada palabra de forma que las palabras se pueden buscar de forma eficiente, independientemente del tamaño del texto. Así, esta técnica adquiere propiedades de índice, obteniendo resultados interesantes para contar, localizar y extraer los snippets de las apariciones de cualquier palabra del texto.

5. Modelo de encapsulación

En esta sección presentamos un modelo que encapsula los documentos de la colección representándolos eficientemente mediante alguna de las técnicas descritas en las secciones anteriores. Ya sea una representación comprimida o autoindexada de la colección, el usuario podrá manipular estas estructuras de datos mediante una interfaz común que determinará completamente su comportamiento externo, quedando oculto el funcionamiento interno de la representación.

Así, la representación compacta y eficiente de la colección de documentos se vuelve una "caja negra", siendo solamente necesario definir una interfaz para poder interactuar con ella. Los detalles de implementación se mantienen ocultos, de forma que no es necesario ningún conocimiento exhaustivo de las estructuras de datos complejas que se usan para llevar a cabo las operaciones sobre la colección, ni de los algoritmos que las soportan.

Actualmente están disponibles las implementaciones de las estructuras de indexación descritas en las secciones anteriores y los algoritmos necesarios para realizar las búsquedas sobre ellas. Próximamente se podrá acceder a la interfaz común que a continuación describimos, de forma que facilite su uso para cualquier usuario no experto en el campo de las estructuras de datos y algoritmos complejos y además que permita intercambiar el tipo de representación dependiendo de la funcionalidad deseada.

5.1. Descripción de las primitivas

A continuación describimos de forma detallada la interfaz mediante la que el usuario interactúa con la representación de la colección de documentos, de forma que se opera como si se tratase de una caja negra, ocultando los detalles internos de su funcionamiento.

Aunque se enumeran las primitivas más básicas, esta interfaz es extensible a cualquier tipo de operación que sea necesario realizar sobre la colección de textos. Si bien alguna operación compleja puede verse como una agregado de alguna de las operaciones simples que se proponen, es posible que la operación compleja pueda realizarse de forma más eficiente en algún tipo de índice determinado que la suma de las operaciones individuales. Por ello, aunque no se detallen a continuación, es posible ampliar el conjunto de primitivas de la interfaz.

Representación de la colección

En esta sección detallamos las primitivas más básicas que permiten, por un lado, construir la representación eficiente de la colección de documentos, y por el otro lado, recuperar cualquier documento o la colección entera a partir de la representación.

- `CrearRepresentacion[tipo]<coleccion>`: permite crear la representación eficiente de la colección de documentos. El usuario debe decidir qué tipo de representación es el más adecuado para representar la colección, decisión que dependerá del uso que realizará sobre los documentos. Para ello se detallan todas las posibles configuraciones internas del índice, con sus características, ventajas e inconvenientes. De esta forma el usuario tendrá toda la información necesaria para elegir la representación más adecuada al uso posterior, y la decisión dependerá de la frecuencia en la que requiera el uso de cada una de las operaciones permitidas. Para crear el índice que represente la colección de documentos se utiliza la llamada `CrearRepresentacion`, en la que se debe especificar el *tipo* de índice que se creará para la representación (índice invertido, array de sufijos comprimido, representación con wavelet tree, etc.), y que recibe como argumento la *coleccion* de documentos que se quieren representar de forma eficiente.
  Para utilizar una configuración óptima, el usuario debe elegir la representación que mejor se adapte a sus necesidades. Para asistirle en esta decisión, en la Tabla 1 se compara la funcionalidad de los índices y su eficiencia para resolver las diferentes tareas. Por ejemplo, si lo que se desea normalmente es localizar apariciones de frases, debería utilizar un WCSA.
  Igualmente, si se diese el caso de una decisión de tipología del índice no adecuada por parte del usuario, la representación de la colección podría ser reconstruida con otro parámetro y reemplazar la representación anterior sin ninguna modificación en el resto del código, ya que todos los índices creados siguen la misma interfaz.
- `Añadir<Documento>`: permite añadir un nuevo documento a la colección. Con determinados tipos de representación de la colección es posible añadir nuevos documentos de forma eficiente, sin necesidad de reconstruir totalmente la representación de toda la colección. Con otros tipos de índice, esta operación no es posible, ya que no soporta dinamismo. Independientemente del tipo de representación utilizada, el usuario puede añadir un nuevo documento a la colección y es la propia representación quien se encarga de realizar el proceso

Tarea	II comprimido	WCSA	BOC-WT
Calcular frecuencias de términos	Eficiente sólo si almacena frec.	Bueno	Óptimo
Localizar apariciones de términos	Depende del tamaño de bloque	Bueno	Muy bueno
Localizar apariciones de frases	Ineficiente	Óptimo	Muy bueno
Extraer snippets	Bueno con tamaño de snippet grande	Muy bueno	Bueno
Extraer todo el documento	Óptimo	Muy bueno	Bueno
Búsqueda de lemas	Es posible	Es posible	No disponible actualmente
Consideración de stopwords	Es posible	Es posible	No disponible actualmente
Añadir documentos a la colección	Eficiente	No eficiente (reconstruir)	Eficiente

Tabla 1. Comparativa entre los distintos tipos de índice.

de adición del texto a la representación de la colección, actuando como se ha dicho previamente como una caja negra. Para ello se pasa como parámetro el nuevo *Documento* a la primitiva `Añadir`.

- `ObtenerColeccion`: permite recuperar la colección de documentos original. La primitiva `CrearRepresentacion` permite obtener la representación eficiente de la colección de documentos. Esta representación sustituye al conjunto de textos original a la hora de procesarlos y trabajar sobre ellos. Sin embargo, es posible que el usuario necesite recuperar la colección de documentos, por lo que es necesaria esta primitiva en la interfaz del modelo de encapsulación.
- `ObtenerDocumento<IdDocumento>`: permite recuperar el texto completo original de un documento específico de la colección de forma eficiente a partir de la representación. Simplemente es necesario invocar a la primitiva pasándole como argumento el número identificativo del texto que se quiere recuperar.

Procesamiento de la colección

La representación del documento debe permitir un procesamiento del texto básico, que pueda devolver la palabra que está en determinada posición de un documento de la colección, devolver la siguiente palabra, un fragmento del documento, etc.

- `Vocabulario`: permite obtener el conjunto de todos los términos distintos que contiene la colección de documentos.
- `Termino<IdDocumento><pos>`: permite obtener la palabra que está en una posición determinada de un documento. De esta forma, se podría procesar el texto palabra por palabra utilizando esta primitiva de forma recurrente.
- `ObtenerDocumento<IdDocumento><posInicial><posFinal>`: permite obtener una parte de un documento. En este caso, restringimos la obtención de un documento al fragmento entre dos posiciones del texto dadas.

Recuperación de datos simples sobre la colección

A continuación detallamos una serie de primitivas que permiten recuperar datos simples de la colección de documentos. Estas primitivas son básicas, por ejemplo, para realizar cálculos como la medida de la frecuencia inversa de documento, imprescindibles para el área de Recuperación de Información.

- `NumeroDocumentos`: es una operación básica que permite obtener el número total de documentos que conforman la colección.
- `NumeroTerminos`: devuelve el número total de palabras distintas que contienen todos los documentos de la colección.
- `NumeroTerminos<IdDocumento>`: de forma análoga a la anterior, devuelve el número de términos distintos en un documento, que se pasa como argumento mediante su identificador de documento.
- `FrecuenciaTermino<termino>`: calcula el número de veces que aparece en todos los documentos de la colección un término en concreto, que se pasa mediante argumento a la primitiva. Es decir, calcula la frecuencia de un término en toda la colección.
- `FrecuenciaTermino<termino><IdDocumento>`: es análoga a la anterior, pero restringida a un documento en concreto que se pasa como argumento mediante su identificador de documento. Es decir, calcula la frecuencia de un término en un documento.
- `NumeroDocumentos<termino>`: permite calcular el número de documentos de la colección que contienen un término. Éste se pasa como argumento a la primitiva.

Búsqueda de términos en la colección

En esta sección se describen las primitivas que permiten realizar búsqueda de términos dentro de los documentos de la colección.

- `Documentos<termino>`: obtiene los documentos de la colección que contienen un término. Este término se pasa como argumento a la primitiva.
- `Posiciones<termino><IdDocumento>`: permite obtener todas las posiciones en las que aparece un determinado término en un documento concreto. Tanto el término como el documento se pasan como argumentos.
- `SiguientePosicion<termino><IdDocumento><posPartida>`: obtiene la posición de la siguiente aparición de un término en un documento concreto. Para ello, se establece una posición de partida en el texto y se busca la aparición más cercana a esa posición del documento.
- `Snippets[longitud]<termino><IdDocumento>`: permite obtener todos los snippets de un término en un documento. Se requiere un parámetro para determinar la longitud de los snippets devueltos, es decir, de los trozos de texto que rodean al término en cada aparición del mismo en el documento.
- `SiguienteSnippet[longitud]<termino><IdDocumento><posPartida>`: de forma análoga a la primitiva `SiguientePosicion`, podemos obtener el snippet de la siguiente aparición del término a una posición dada.

Otras funcionalidades

Dependiendo del tipo de representación que se esté utilizando, es posible extender el conjunto de operaciones permitidas. Por ejemplo, si se está utilizando un índice que mantenga información de lemas y variantes de términos, es posible utilizarlo para obtener las diferentes variantes de un término dado en un documento.

Asimismo, esta interfaz está abierta a cualquier demanda de funcionalidad proveniente de cualquier comunidad investigadora que vea interesante el uso de esta representación, que ofrece un acceso, manipulación y almacenamiento eficientes de una colección de documentos, permitiendo la realización de las tareas propias de sus campos de investigación sin necesidad de entender los detalles complejos de las estructuras de datos ni de sus algoritmos.

6. Conclusiones y trabajo futuro

En este trabajo se han revisado los últimos avances en compresión e indexación para texto en lenguaje natural. Hemos puesto especial atención a las técnicas de compresión semi-estáticas orientadas a palabras, que pueden ser integradas junto a estructuras de indexación como los índices invertidos, mejorando su rendimiento en todos los aspectos. En cuanto a técnicas de indexación nos hemos centrado en los índices invertidos comprimidos y en los autoíndices de palabras. Estas estructuras permiten representar una colección de textos de forma compacta al mismo tiempo que permiten manejarla de forma eficiente.

En la investigación en Recuperación de Información e Ingeniería Lingüística no suele ser prioritaria la eficiencia de las técnicas de representación de las colecciones de texto con las que trabaja, centrándose fundamentalmente en aspectos relacionados con la eficacia. Sin embargo, la utilización de estructuras avanzadas de compresión e indexación como base de sus sistemas mejoraría notablemente la eficiencia de los mismos. Por ello, proponemos un modelo que encapsula la representación interna de las colecciones y ofrece al usuario/investigador una interfaz con la funcionalidad necesaria para realizar sus tareas.

Estamos actualmente terminando la implementación de la encapsulación de las estructuras de datos y algoritmos de los distintos métodos citados en este trabajo: un índice invertido comprimido y los autoíndices WCSA y BOC-WT. El siguiente paso será hacerlo público para que cualquier usuario pueda descargarlo y utilizarlo.

Referencias

1. R. Baeza-Yates, A. Moffat, and G. Navarro. *Searching Large Text Collections*, pages 195–244. Kluwer Academic, 2002.
2. R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley Longman, 1999.
3. R. S. Boyer and J. S. Moore. A fast string searching algorithm. *Communications of the ACM*, 20(10):762–772, 1977.

4. N. Brisaboa, A. Fariña, S. Ladra, and G. Navarro. Reorganizing compressed text. In *Proc. 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 139–146. ACM Press, 2008.
5. N. Brisaboa, A. Fariña, G. Navarro, and M. Esteller. (s,c)-dense coding: An optimized compression code for natural language text databases. In *Proc. 10th International Symposium on String Processing and Information Retrieval (SPIRE)*, LNCS 2857, pages 122–136. Springer, 2003.
6. N. Brisaboa, A. Fariña, G. Navarro, and J. Paramá. Lightweight natural language text compression. *Information Retrieval*, 10(1):1–33, 2007.
7. N. Brisaboa, A. Fariña, G. Navarro, A. Places, and E. Rodríguez. Self-indexing natural language. In *Proc. 15th International Symposium on String Processing and Information Retrieval (SPIRE)*, LNCS 5280, pages 121–132. Springer, 2008.
8. Paolo Ferragina and Johannes Fischer. Suffix arrays on words. In *In Proceedings of the 18th Annual Symposium on Combinatorial Pattern Matching, volume 4580 of LNCS*, pages 328–339. Springer, 2007.
9. Travis Gagie, Simon J. Puglisi, and Andrew Turpin. Range quantile queries: Another virtue of wavelet trees. In *SPIRE '09: Proceedings of the 16th International Symposium on String Processing and Information Retrieval*, pages 1–6, Berlin, Heidelberg, 2009. Springer-Verlag.
10. R. Grossi, A. Gupta, and J.S. Vitter. High-order entropy-compressed text indexes. In *Proceedings of 14th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 03)*, pages 841–850, 2003.
11. R. N. Horspool. Practical fast searching in strings. *Software Practice and Experience*, 10(6):501–506, 1980.
12. D. A. Huffman. A method for the construction of minimum-redundancy codes. *Proceedings of the IRE*, 40(9):1098–1101, 1952.
13. U. Manber and G. Myers. Suffix arrays: a new method for on-line string searches. *SIAM Journal on Computing*, 22(5):935–948, 1993.
14. A. Moffat. Word-based text compression. *Softw. Pract. Exper.*, 19(2):185–198, 1989.
15. E. Moura, G. Navarro, N. Z., and R. Baeza-Yates. Fast and flexible word searching on compressed text. *ACM TOIS*, 18(2):113–139, 2000.
16. G. Navarro and V. Mäkinen. Compressed full-text indexes. *ACM Computing Surveys*, 39(1):article 2, 2007.
17. K. Sadakane. New text indexing functionalities of the compressed suffix arrays. *Journal of Algorithms*, 48(2):294–313, 2003.
18. A. Turpin and A. Moffat. Fast file search using text compression. In *Proc. 20th Australian Computer Science Conference (ACSC)*, pages 1–8, 1997.
19. I.H. Witten, A. Moffat, and T.C. Bell. *Managing Gigabytes: Compressing and Indexing Documents and Images*. Morgan Kaufmann Publishers, USA, 1999.
20. J. Ziv and A. Lempel. A universal algorithm for sequential data compression. *IEEE Transactions on Information Theory*, 23(3):337–343, 1977.
21. J. Ziv and A. Lempel. Compression of individual sequences via variable-rate coding. *IEEE Transactions on Information Theory*, 24(5):530–536, 1978.
22. J. Zobel and A. Moffat. Inverted files for text search engines. *ACM Comp. Surv.*, 38(2):article 6, 2006.

Sistema interactivo de búsqueda de respuestas con sugerencia de términos

Angel Zazo, José Luis Alonso Berrocal, Carlos G. Figuerola, y Raquel Gómez Díaz

Grupo de investigación REINA - Universidad de Salamanca
C/ Francisco de Vitoria, 6-16, 37008 – Salamanca
{zazo, berrocal, figue, rgomez}@usal.es

Resumen Este trabajo analiza el comportamiento del usuario en un sistema interactivo de búsqueda de respuestas que incluye mecanismos de sugerencia de términos. El análisis se lleva a cabo utilizando metodologías de evaluación aplicadas en experimentos de este tipo. Se organizó un grupo de 16 usuarios, cada uno de ellos debía encontrar la respuesta a ocho preguntas, más dos de entrenamiento; sólo en la mitad se aplicaba el mecanismo de sugerencia de términos. Los resultados muestran que la posibilidad de añadir términos relacionados con los de la pregunta fue bien valorada por los usuarios, que la consideraron útil, y con la que se obtuvieron respuestas más precisas. Hasta el momento hemos utilizado relaciones de coocurrencia de términos para crear un tesauro de términos relacionados con los de la pregunta, que se muestran al usuario mediante una capa flotante para, si lo desea, añadirlos a su pregunta, pero estamos trabajando también en técnicas de extracción de información.

1. Introducción

Una de las tareas más avanzadas en Recuperación de Información es aquella que busca respuestas precisas a necesidades de información concretas. Esto es lo que se conoce como búsqueda de respuestas (BR), *question answering* en inglés. Los sistemas de BR deben encontrar un fragmento de texto mínimo que responda a la necesidad informativa concreta. Por eso, algunas aproximaciones a este problema están basadas en la recuperación de pasajes o fragmentos de texto [12, 13]. La respuesta puede ser extraída o generada después mediante algún proceso automático o, si el sistema no es capaz de proporcionar una respuesta válida de una manera totalmente autónoma, debe disponerse de algún proceso de interacción con el usuario para que este pueda satisfacer su necesidad informativa.

Muchos usuarios utilizan sistemas estándar de recuperación de información, como los motores de búsqueda de Internet, para los procesos de búsqueda de respuestas. Es un escenario bastante frecuente. La gran cantidad de documentos y la habilidad del usuario para refinar sus búsquedas permite a estos reconocer la respuesta buscada dentro de los documentos recuperados. Dada la complejidad de este tipo de búsquedas, el nivel de participación del usuario, en términos de cantidad y de calidad, es mayor que para otros sistemas de recuperación. La

definición de una acertada estrategia de búsqueda y su formulación requieren un nivel de experiencia mayor. Sin ningún tipo de asistencia, debe ser el propio usuario el que, en su proceso de refinamiento, dé solución a todos los problemas que va encontrando para plasmar adecuadamente su necesidad informativa.

El objetivo de este trabajo ha sido aplicar tecnologías de análisis de textos y mecanismos de presentación de información, como elementos de ayuda para que los usuarios sean capaces de plasmar de forma más precisa su necesidad informativa. Partiendo de los términos de búsqueda, o al menos de los caracteres que va tecleando el usuario, han debido resolverse varios aspectos, como qué mecanismos de sugerencia de términos pueden aplicarse, cuáles pueden ser los términos sugeridos, cómo presentar al usuario la información que pueda ayudarle en el planteamiento de su consulta, o cómo realizar la presentación de resultados.

Uno de los objetivos concretos ha sido determinar si la sugerencia de términos ayuda al usuario a encontrar mejores respuestas a las preguntas formuladas, en cuanto a precisión y rapidez. Ahora bien, la experimentación con usuarios reales entraña algunos inconvenientes. Por ejemplo, una misma pregunta no puede ser formulada al mismo usuario más que una sola vez, por el efecto de aprendizaje que supone haberla contestado previamente. Para evitar efectos adversos lo normal es utilizar un orden pseudo-aleatorio de preguntas y sistemas a evaluar. En la sección 3 volveremos sobre este aspecto.

En la Sección 2 se describen los módulos de sugerencia de términos. En la Sección 3 se describe el diseño de nuestros experimentos. En la Sección 4 analizamos los resultados, y en la 5 se presentan las conclusiones y comentarios.

2. Sugerencia de términos

Existen varios mecanismos para asistir al usuario en el planteamiento de sus necesidades informativas, en general mediante la sugerencia o exploración de términos [9], unas veces de construcción manual, como los tesauros de los distribuidores comerciales de bases de datos, otras de construcción automática, como la utilización de jerarquías de términos extraídas de agrupación documental, relaciones de co-ocurrencia entre términos, etc. En los actuales buscadores de Internet el usuario puede verse ayudado principalmente de dos maneras: mediante el *autocompletado* de términos o frases según se van introduciendo caracteres en el campo de búsqueda (Google, Yahoo!), o mediante la *sugerencia de términos relacionados* con los que ha introducido en su búsqueda (lo ofrecen Google y Yahoo! en botones especiales). Esos han sido también los módulos que hemos desarrollado en nuestro trabajo.

Nuestro *módulo de autocompletado* se lanzaba automáticamente al ir tecleando el usuario los términos de búsqueda. Se implementó una capa flotante debajo del campo de búsqueda en la que aparecían términos que comenzaban por los caracteres tecleados hasta ese momento. Se permitía incluso el autocompletado de varios términos. Sin embargo, una vez desarrollado dicho módulo, decidimos prescindir de él, pues nos dimos cuenta de que en realidad no suponía ninguna ventaja para los usuarios: las preguntas estaban formuladas de antemano, con

lo que el usuario ya sabía los términos que tenía que buscar. Por otra parte, entre los términos sugeridos aparecían con demasiada frecuencia términos que claramente eran errores de tecleo, y eso podía llegar a confundir al usuario en vez de ayudarlo. Este fue otro motivo para no incluirlo en la interfaz final.

El *módulo de sugerencia de términos relacionados* con los de la consulta se presentaba igualmente como una capa flotante debajo del campo de búsqueda, y contenía términos que podían ser añadidos a la búsqueda. Decidimos que no apareciese de manera automática, sino que se lanzase cuando el usuario tuviese la necesidad de que el sistema le sugiriese términos. Utilizamos para ello el carácter dos puntos (:) en vez de botones especiales. Para el desarrollo de este módulo utilizamos relaciones de co-ocurrencia: dos términos están relacionados si aparecen en los mismos contextos. Ello nos permite crear un tesauro de términos relacionados con los de la consulta. Para medir el grado de relación entre dos términos existen varias funciones [10]; hemos utilizado la de Dice porque en trabajos previos hemos comprobado su buen comportamiento [15]. Como unidad de co-ocurrencia empleamos pasajes de texto: justamente los términos sugeridos proceden de los primeros pasajes recuperados teniendo en cuenta los términos introducidos por el usuario antes de escribir el carácter dos puntos.

La ecuación (1) muestra la función de asociación de Dice entre un término t_q de la consulta y un término t_k con el que co-ocurre en uno o más pasajes; n_q y n_k son el número de pasajes en los que aparecen t_q y t_k, respectivamente, y n_{qk} el número de pasajes en los que co-ocurren.

$$\mathrm{Dice}(t_q, t_k) = \frac{2 \cdot n_{qk}}{n_q + n_k} \tag{1}$$

Para cada término de los pasajes se calculó el valor de asociación con toda la consulta sumando los valores individuales de asociación con cada uno de los términos de la misma. Finalmente, los términos con valores más altos de asociación se mostraban al usuario (véase la Fig. 1). Comprobamos, no obstante, que los términos más frecuentes en los pasajes introducían un efecto negativo en la sugerencia de términos. Los términos más frecuentes son también los candidatos a ser los mejor relacionados con el resto de términos, y por tanto debieran ocupar posiciones destacadas al ser sugeridos al usuario, pero el usuario los encontraría tan generales que le servirían de poco para refinar su consulta. En la ecuación (2) se evita el efecto de valores altos de IDF [11], siendo N el número de pasajes considerados a la hora de obtener las relaciones de co-ocurrencia; su valor se fijó en 30, igual al número de pasajes que se mostraban al usuario en la interfaz de búsqueda.

$$\mathrm{Dice}^{\mathrm{IDF}}(t_q, t_k) = \frac{2 \cdot n_{qk}}{n_q + n_k} \cdot \log \frac{N}{n_k} \tag{2}$$

3. Experimentos

En nuestros experimento hemos seguido la metodología utilizada en varias tareas de las conferencias TREC y CLEF para la evaluación de sistemas interactivos

de búsqueda de respuestas [3,5]. Hemos utilizado la colección de documentos en español procedente de la Agencia EFE de noticias correspondiente a los años 1994 y 1995, utilizada con fines de investigación bajo licencia de ELRA y utilizada extensamente en las conferencias CLEF [1]. Se trata de 454.045 documentos etiquetados bajo SGML con varios campos, como número de documento, título, texto de la noticia, fecha, categoría, etc. En nuestros experimentos solamente hemos utilizado los campos de título, texto y fecha. Tal como se ha indicado en la introducción, la búsqueda de respuestas muy a menudo utiliza pasajes de texto como unidades de búsqueda. Es el enfoque que nosotros hemos aplicado, entre otras cosas porque para un usuario es más sencillo localizar la respuesta a una pregunta en un pasaje de texto que tener que leer el documento completo.

En varios experimentos interactivos se ha comprobado que la posibilidad de ver el documento completo del que deriva el pasaje debiera ser una de las características deseables del sistema, sin duda, porque al dividir el documento en pasajes se pierde un contexto fundamental que sirve para verificar que la respuesta es correcta [2,7]. En nuestro caso hemos decidido intencionadamente no disponer de esa posibilidad, y centrarnos exclusivamente en la sugerencia de términos.

Cada pasaje de texto estaba formado por al menos un párrafo. Después de varias pruebas se decidió que cada pasaje debía tener al menos 60 palabras. Si al ir dividiendo el documento en párrafos, alguno de los pasajes tenía menor número de palabras, se le iba concatenando otro párrafo, de manera que el pasaje tuviera al menos 60 palabras. El número total de pasajes fue de casi dos millones, con una media de 77,8 palabras por pasaje.

En cuanto a las preguntas del experimento, en la Tabla 1 pueden verse las que se utilizaron. Procedían de las tareas interactivas de las conferencias CLEF de los años 2004 y 2005, aunque aquí el entorno ha sido monolingüe. En esas tareas el número de preguntas fue justamente el doble, convirtiéndose en un experimento largo y pesado para los usuarios [14], por eso seleccionamos un número de preguntas menor. Las preguntas 9 y 10 se emplearon como preguntas de entrenamiento para que los usuarios se familiarizasen con la interfaz de consulta. Todas las preguntas tenían respuesta en la colección documental.

Para poder dar por correcta una respuesta el usuario debía indicar el pasaje de texto en el que la había encontrado. Si no indicaba el pasaje, o este no respondía verdaderamente a la pregunta, se consideró que la respuesta no era válida en sentido estricto, salvo que el usuario ya conociese la respuesta de antemano. Para asegurar este punto, se pasó previamente un cuestionario en el que el usuario debía indicar su conocimiento sobre cada pregunta.

3.1. Sistema de recuperación

No hemos utilizado un sistema de búsqueda de respuestas en nuestros experimentos, sino que hemos adaptado un sistema clásico de recuperación de información para esta tarea. Varios experimentos como el nuestro se han desarrollado con sistemas genéricos de búsqueda [6], de hecho, es muy frecuente que los usuarios de Internet utilicen buscadores generales para la búsqueda de respuestas. Es de

Tabla 1. Preguntas del experimento.

Nº	**Pregunta**
1	¿Qué institución inició la campaña europea de la juventud contra el racismo?
2	¿Qué iglesia ordenó mujeres sacerdote en marzo de 1994?
3	¿Quién era primer ministro de Noruega cuando se celebró el referéndum sobre su posible incorporación a la UE?
4	¿Cuándo se estima que ocurrió el Big Bang?
5	¿Cuánto costó el Túnel del Canal?
6	¿Quién es el director gerente del Fondo Monetario Internacional?
7	Fecha de la muerte de Lenin
8	¿Quién ganó el Premio Nobel de Literatura en 1994?
9	¿Cuándo abdicó Eduardo VIII?
10	¿Con el nombre de qué enfermedad se corresponde el acrónimo BSE?

señalar que en algunos casos la utilización de filtros propios de la búsqueda de respuestas para seleccionar el tipo de respuesta buscada (nombres, referencias temporales o cantidades) no han conseguido mejorar los resultados en procesos interactivos [7].

En cuanto al proceso de indización, se eliminaron acentos, se redujeron las palabras a minúsculas, y se consideraron vacías las palabras que aparecían en más del 5 % de los documentos; no se aplicó lematización, aunque se podía utilizar el asterisco como operador de truncamiento por la derecha, pero fue poco utilizado.

3.2. El experimento interactivo

La interacción con el usuario presenta aspectos importantes que deben ser destacados en nuestros experimentos. Por un lado, se trata de un experimento de búsqueda de respuestas, y en este caso, el conocimiento previo de su temática o de su posible respuesta condiciona la manera en la que los usuarios plantean sus búsquedas, de ahí el cuestionario previo. Por otro lado, para dar solución a cada una de las preguntas hay que proporcionar un tiempo suficiente, pero finito; de otra manera, el usuario puede dedicar demasiado tiempo a las primeras preguntas y llegar muy cansado a las últimas. Para resolver cada una de las preguntas se contó con un tiempo máximo de cinco minutos.

La mayoría de experimentos interactivos presentan dos sistemas para ser evaluados; ambos son idénticos salvo en alguna de las funcionalidades que uno de ellos presenta y el otro no. Esa funcionalidad es la que se desea evaluar. En nuestro caso ha sido la posibilidad de disponer de sugerencia de términos relacionados con los que introduce el usuario. Hemos denominado **sistema A** al sistema que no permite la sugerencia de términos, y **sistema B** al que sí la permite. Cada usuario debía resolver la mitad de las preguntas en un sistema y la otra mitad en el otro. Todos los usuarios eran instruidos previamente, disponiéndose de dos preguntas de entrenamiento, una en cada sistema. Para evitar el efecto que el orden de las preguntas o del sistema pudiera tener en el desarrollo del experimento, se utilizó un orden pseudo-aleatorio de actuación para cada uno de los usuarios,

Tabla 2. Esquema de actuación de cada usuario.

Nº	**Entren.**	**Experimento**	**Nº**	**Entren.**	**Experimento**
u1	B:10 A:9	B:4-7-5-8 A:1-3-2-6	u9	A:10 B:9	A:4-7-5-8 B:1-3-2-6
u2	A:9 B:10	A:3-5-7-1 B:8-4-6-2	u10	B:9 A:10	B:3-5-7-1 A:8-4-6-2
u3	A:10 B:9	A:1-3-4-6 B:2-8-7-5	u11	B:10 A:9	B:1-3-4-6 A:2-8-7-5
u4	A:9 B:10	A:5-2-6-3 B:4-7-1-8	u12	B:9 A:10	B:5-2-6-3 A:4-7-1-8
u5	B:10 A:9	B:7-6-2-4 A:3-5-8-1	u13	A:10 B:9	A:7-6-2-4 B:3-5-8-1
u6	B:9 A:10	B:8-4-3-2 A:6-1-5-7	u14	A:9 B:10	A:8-4-3-2 B:6-1-5-7
u7	A:10 B:9	A:6-1-8-7 B:5-2-4-3	u15	B:10 A:9	B:6-1-8-7 A:5-2-4-3
u8	B:9 A:10	B:2-8-1-5 A:7-6-3-4	u16	A:9 B:10	A:2-8-1-5 B:7-6-3-4

como puede verse en la Tabla 2. Este modelo de actuación ha sido aplicado en experimentos similares en tareas interactivas de TREC y CLEF [3, 5].

El **sistema A** consistió en presentar al usuario 30 pasajes de texto en los que posiblemente se encontrase la respuesta a la pregunta. Se eligió este número para que fuese mayor que dos pantallas de resultados de motores de Internet (20 documentos), número máximo de pantallas que los usuarios suelen ver. El **sistema B** era idéntico al *sistema A*, pero incluía el módulo de sugerencia de términos: el usuario escribía los términos de la consulta que le parecían convenientes y luego podía, si así lo deseaba, escribir el carácter dos puntos para que el sistema ofreciese términos relacionados (véase la Fig. 1).

En experimentos interactivos es importante registrar las impresiones de los usuarios, y para ello hemos utilizado tres cuestionarios: uno previo al experimento, otro al finalizar la sesión de preguntas con cada sistema, y uno más al final. El primero tenía el objetivo de determinar el grado de experiencia de cada usuario en sistemas de búsqueda de información y en el conocimiento previo de las preguntas. En el cuestionario posterior a cada sistema se realizaban preguntas sobre dicho sistema, haciendo énfasis en la funcionalidad que se quería resaltar. El cuestionario final tenía que ver con todo el experimento, una comparativa de ambos sistemas y comentarios de cada uno de ellos. Más adelante se describen los resultados de dichos cuestionarios.

Para nuestros experimentos formamos un grupo de 16 usuarios, con una edad media de 21,9 años, todos ellos con amplia experiencia en sistemas de búsquedas en OPAC y en Internet, es decir, sistemas que utilizan los términos de búsqueda como términos índice. Se trataba de alumnos del *Grado en Información y Documentación* y de la *Licenciatura en Documentación* de la Universidad de Salamanca, lo cual ha dotado a este estudio de un valor añadido. En la Tabla 3 se pueden ver los resultados del cuestionario previo al experimento. Se indican valores medios en una escala de 1 a 5.

El bagaje cultural de los usuarios puede llevar a distorsiones en los resultados de un experimento interactivo de búsqueda de respuestas, debido a la propia temática de las preguntas. En este sentido, el cuestionario previo sobre la familiaridad del usuario con las preguntas mostró que para el 65 % de las mismas los usuarios no tenían idea de cuáles podrían ser sus respuestas. Para el 34 % de las preguntas los usuarios indicaron que el tema les sonaba, pero que no conocían

Tabla 3. Resultados del cuestionario previo al experimento.

Número de años que realiza búsquedas en línea	8,2
Experiencia de búsquedas en OPAC	4,0
Experiencia en buscadores de Internet	4,7
Frecuencia de búsqueda	4,8
Satisfacción en la búsqueda de información	4,0

la respuesta. Solamente dos usuarios dijeron saber la respuesta de una pregunta (la pregunta 2). En resumen, salvo en estos dos casos, para el resto de preguntas no era conocida su respuesta exacta.

3.3. La interfaz de búsqueda

Para el diseño de la interfaz utilizamos los típicos formularios web, un entorno muy conocido por los usuarios. La Figura 1 muestra la interfaz interactiva de búsqueda para el *sistema B* con la pregunta de entrenamiento número 10 justamente después de que el usuario haya añadido a sus términos de búsqueda "enfermedad" y "BSE" los dos primeros términos sugeridos, "locas" y "vacas". La interfaz permitía modificar la búsqueda en cualquier momento dentro del tiempo restante, mostrado en la parte superior derecha. En la parte central aparecían

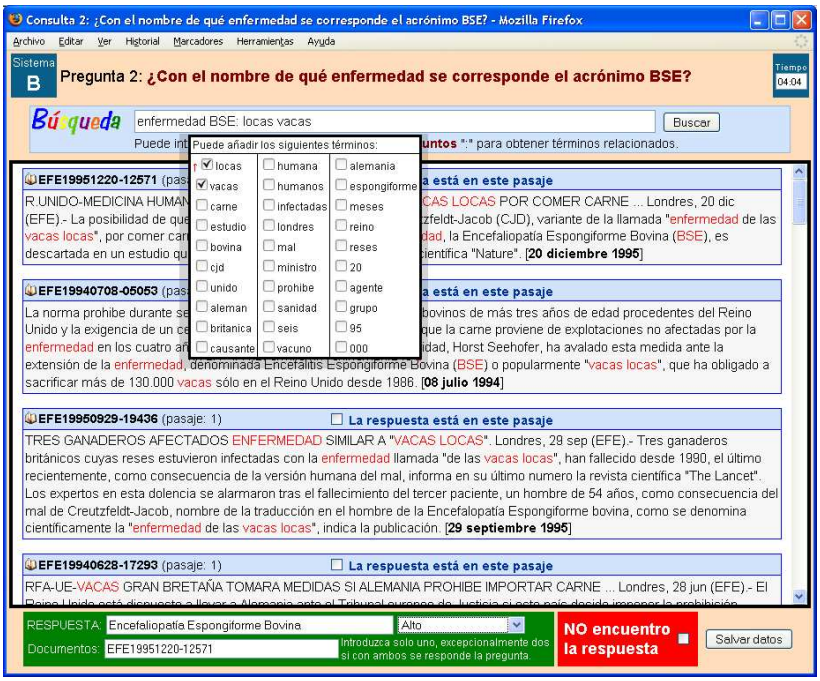


Figura 1. Interfaz de consulta.

los pasajes devueltos por el sistema de recuperación. Un aspecto destacado de la interfaz de usuario era que mostraba resaltados en color rojo aquellos términos no vacíos de la pregunta que aparecían en los pasajes de texto. De esta manera el usuario podía fácilmente localizar la respuesta a las preguntas. La interfaz de búsqueda del *sistema A* era idéntica a la del *sistema B*, solamente que no se permitía la sugerencia de términos.

Cuando el usuario encontraba la respuesta a la pregunta debía marcar el pasaje correspondiente y escribir la respuesta en la parte inferior de la ventana, o en caso de darse por vencido, marcar que no encontró la respuesta, y seguir con una nueva pregunta. De cualquier modo, una vez superados los cinco minutos el sistema automáticamente mostraba una última posibilidad de introducir una respuesta o marcar que no se había encontrado.

3.4. Juicios

Decidimos utilizar los mismos juicios de valoración de las respuestas que los utilizados en la búsqueda de respuestas [8]: una respuesta es correcta (R) si es exacta y el documento indicado la respalda; una respuesta es inexacta (X) si el documento contiene la respuesta, pero ésta no es correcta por incompleta o por contener más texto del necesario; una respuesta sin respaldo (U) es aquella que es correcta pero sin respaldo del documento. El resto son incorrectas.

Considerando esta división se tienen dos tipos de medidas de precisión: estricta y relajada. La precisión en sentido estricto considera únicamente como correctas las respuestas exactas (R), mientras que en sentido relajado también considera correctas las que no tienen respaldo (R+U). Por el diseño de nuestra interfaz sólo se dieron respuestas sin respaldo en dos ocasiones, pues se obligaba al usuario a incluir el pasaje al dar una respuesta. Por este motivo, en los resultados utilizaremos la precisión en sentido relajado.

4. Resultados

4.1. Precisión

En la Tabla 4 se aprecia que el *sistema B* es superior en un 23 % al *sistema A*. Ahora bien, no todos los usuarios utilizaron la sugerencia de términos en el *sistema B*. El número de preguntas para las que se utilizó la sugerencia de términos fue de 41, de un total de 64, y fueron correctas en sentido relajado 36, el 87,8 %. De las 87 preguntas restantes que no utilizaron la sugerencia de términos

Tabla 4. Precisión en sentido relajado y tiempo medio por consulta.

Sistema	Precisión	Tiempo
A	0,61	65
B	0,75	72

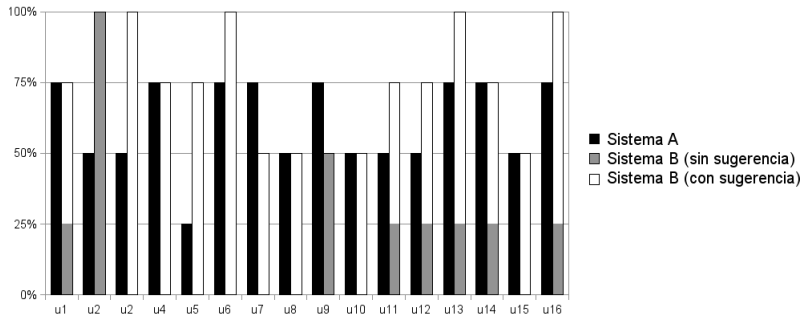


Figura 2. Precisión en sentido relajado para cada usuario.

para ser resueltas, fueron correctas 51, es decir, el 58,6 %. Esto significa que la sugerencia de términos produce más éxitos. Este resultado no es igual para todos los usuarios: en la Fig. 2 pueden verse los resultados de manera individualizada.

En cuanto al tiempo medio empleado para resolver las preguntas correctas (en sentido relajado), fue de 65 segundos para el *sistema A* y 72 para el *sistema B*. Este resultado no debe engañarnos, pues se trata de valores medios para todo el *sistema B*. Aquellas preguntas que utilizaron la sugerencia de términos se resolvieron en realidad más rápidamente, con una media de 64 segundos. Habíamos esperado que la sugerencia de términos agilizase el proceso de búsqueda, no obstante, hay que tener en cuenta que el usuario requiere cierto tiempo para que pueda ver los términos sugeridos y seleccionar aquellos que considera más pertinentes para añadir a la búsqueda y resolver la pregunta.

4.2. Dificultad de las preguntas. Refinamiento de búsquedas

No todas las preguntas tuvieron la misma dificultad para ser resueltas. La Figura 3 muestra la precisión en sentido relajado de cada pregunta, independientemente del sistema utilizado. La pregunta 5, *¿Cuánto costó el Túnel del Canal?*, fue mal contestada en la mayoría de los casos porque en uno de los pasajes se

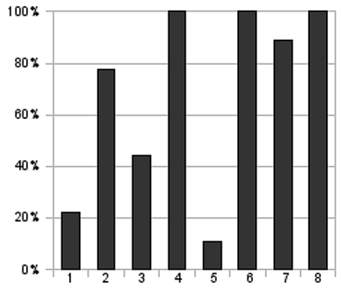


Figura 3. Precisión de las respuestas.

citaba una parte del coste de dicho canal, y no el coste completo. La pregunta 1, *¿Qué institución inició la campaña europea de la juventud contra el racismo?*, resultó difícil porque, a pesar de que la respuesta aparecía con poca dificultad, los usuarios desconocían la organización política de la UE en los años 1994 y 1995. La pregunta 3, *¿Quién era primer ministro de Noruega cuando se celebró el referéndum sobre su posible incorporación a la UE?*, resultó también complicada porque se trataba de una mujer, en vez de un hombre (recordemos que el sistema no aplicaba lematización en el proceso de indización).

Es importante analizar el refinamiento de las consultas. El número medio de consultas por pregunta fue de 3,6 para el *sistema A* y 3,4 para el *sistema B*, es decir, ligeramente inferior. Ahora bien, debemos tener en cuenta que los usuarios utilizaron la sugerencia de términos aproximadamente una vez por pregunta, y eso significa que de manera invisible para el usuario el motor de sugerencia tuvo que realizar una consulta. Hay que señalar también que en la mayoría de los casos la sugerencia de términos fue lanzada al comienzo de la búsqueda de las respuestas, un 67 % de veces frente a un 33 % que se lanzo en mitad de una búsqueda. Hemos comprobado, al igual que en otro estudio [4], que varios usuarios utilizaban la reformulación como mecanismo de verificación de respuestas, al incluir los términos de las mismas en la nueva reformulación, y eso puede dar lugar a errores en la interpretación.

4.3. Cuestionarios

Cada usuario respondió un cuestionario después de las sesiones en cada uno de los sistemas. El resultado se sintetiza en la Figura 4. En todos los aspectos, el *sistema B* fue considerado ligeramente mejor que el *sistema A*. En el cuestionario último que comparaba ambos sistemas y recogía comentarios de los usuarios, estos se decantaron claramente por el *sistema B* como mejor sistema. Un alto porcentaje de usuarios manifestaron que la sugerencia de términos les había presentado términos que de otro modo no habrían utilizado en sus consultas.

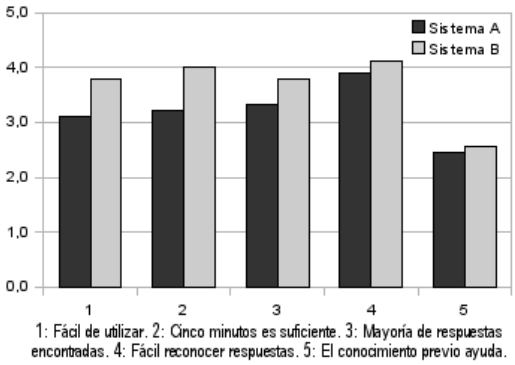


Figura 4. Resultados de los cuestionarios post-sistema.

5. Comentarios y conclusiones

Hemos explorado la actuación del usuario en un sistema interactivo de búsqueda de respuestas con sugerencia de términos. Para ello se ha utilizado un sistema convencional de recuperación de información basado en el modelo del espacio vectorial, al que se le han añadido ciertas características adicionales para la tarea interactiva: división de los documentos en pasajes de texto, sugerencia de términos para la expansión de la pregunta y una interfaz de usuario adecuada.

Uno de los primeros aspectos que deseamos resaltar es el hecho de haber obtenido una precisión del 68 % en la resolución de todas las preguntas, un valor muy alto. Esto indica que cuando la interacción es posible, un sistema simple proporciona mejores resultados que un sistema completamente automático [3].

En relación con el método interactivo de ayuda de la sugerencia de términos (obtenidos de acuerdo a la construcción automática de un tesauro de asociación), los resultados muestran que se trata de una opción apreciada por los usuarios, al permitir ver y utilizar términos que a priori el usuario no hubiese utilizado, obteniéndose mejores resultados.

Uno de los aspectos que los usuarios indicaron como muy útiles para la localización de la respuesta fue el resaltado de los términos de búsqueda en los pasajes de texto, pues rápidamente les permitía situarse en las zonas más prometedoras para encontrar la solución a las preguntas.

En relación con el sistema de recuperación en sí, con o sin sugerencia de términos, la mayoría de usuarios manifestó que era sencillo de utilizar, al parecerse mucho la interfaz de consulta a la de los motores de búsqueda en Internet. Alguno incluso llegó a indicar que la posibilidad de utilizar truncamiento, como en nuestro caso, era una opción muy acertada, no olvidemos que los usuarios procedían del campo documental.

Un aspecto que debemos comentar es si el experimento está bien planteado. Por ejemplo, los resultados con usuarios reales son difíciles de repetir, sobre todo si el número de usuarios es pequeño, o si cambia cualquiera de las condiciones de formación de los mismos (alumnos de otras titulaciones, por ejemplo). En este sentido es muy difícil incorporar a la evaluación el conocimiento previo que los usuarios pudieran tener en lo referente a sistemas de búsqueda de información, planteamiento de estrategias de búsqueda, conocimiento del propio sistema de recuperación, el bagaje cultural sobre las propias preguntas, o cómo son los procesos de inferencia que las personas utilizamos para resolver preguntas. El orden pseudo-aleatorio que se ha seguido ha pretendido reducir efectos no deseados en el orden de sistemas y preguntas, pero es seguro que no los ha eliminado por completo. También es difícil justificar por qué aparentemente algunas preguntas fáciles de responder por la mayoría de usuarios, a unos pocos les resultan especialmente complicadas.

Sería deseable repetir los experimentos con muchos más grupos de usuarios y de preguntas, pero los experimentos interactivos como el que hemos realizado suponen un duro trabajo de planificación, de selección de usuarios y en el desarrollo de los mismos, y muchas veces no es posible realizarlos. No obstante, es nuestra intención seguir con esta línea de investigación, y ya estamos analizando

técnicas de extracción de información y resumen automático que nos permita sugerir algo parecido a *conceptos* en vez de términos.

Agradecimientos

Este trabajo ha sido financiado por los proyectos KA3Ñ-2009 de la Universidad de Salamanca y FS/9-2008 de la Fundación Memoria Samuel Solórzano Barruso.

Referencias

1. Braschler, M., Peters, C.: CLEF 2003 methodology and metrics. In: CLEF 2003, LNCS. vol. 3237, pp. 7–20 (2004)
2. Figuerola, C.G., Zazo, A., Alonso Berrocal, J.L., Rodríguez, E.: Interactive and bilingual question answering using term suggestion and passage retrieval. In: CLEF 2004, LNCS. vol. 3491, pp. 363–370 (2005)
3. Gonzalo, J., Oard, D.W.: Overview of the CLEF 2004 interactive track. In: CLEF 2004, LNCS. vol. 3491, pp. 310–322 (2005)
4. He, D., Wang, J., Luo, J., Oard, D.W.: iCLEF 2004 at Maryland: Summarization design for interactive cross-language question answering. In: Working Notes for the CLEF 2004 Workshop, 15-17 September, Bath, UK (2004)
5. Hersh, W., Over, P.: The TREC-9 interactive track report. In: The Ninth Text REtrieval Conference (TREC 9). pp. 41–49 (2000)
6. Hersh, W., Over, P.: TREC-2001 interactive track report. In: The Tenth Text REtrieval Conference (TREC 2001). pp. 41–49 (2001)
7. López-Ostenero, F., Peinado, V., Gonzalo, J., Verdejo, F.: Interactive question answering: Is cross-language harder than monolingual searching? Information Processing & Management 44(1), 66 – 81 (2008)
8. Magnini, B., Romagnoli, S., Vallin, A., Herrera, J., Peñas, A., Peinado, V., Verdejo, F., de Rijke, M.: The multiple language Question Answering track at CLEF 2003. In: CLEF 2003, LNCS. vol. 3237, pp. 471–486 (2004)
9. Peñas, A., Gonzalo, J., Verdejo, F.: Acceso a la información mediante exploración de sintagmas. In: Canós, J., García, P. (eds.) III Jornada de Bibliotecas Digitales JBIDI'02. 18 y 19 de Noviembre. El Escorial (Madrid). pp. 57–66 (2002)
10. Salton, G., McGill, M.J.: Introduction to Modern Information Retrieval. McGraw-Hill, New-York (1983)
11. Spärck Jones, K.: A statistical interpretation of term specificity and its applications in retrieval. Journal of Documentation 28(1), 11–21 (1972)
12. Vicedo González, J.L.: Recuperación de Información de Alta Precisión: Los Sistemas de Búsqueda de Respuestas. Sociedad Española para el Procesamiento del Lenguaje Natural, Alicante (2003)
13. Woods, W.A., Green, S., Martin, P., Houston, A.: Halfway to question answering. In: The Ninth Text REtrieval Conference (TREC 9). pp. 489–500 (2000)
14. Zazo, A., Figuerola, C.G., Alonso Berrocal, J.L., Fernández Marcial, V.: Use of free on-line machine translation for interactive cross-language question answering. In: CLEF2005, LNCS. vol. 4022, pp. 263–272 (2006)
15. Zazo, A., Figuerola, C.G., Berrocal, J.L.A., Rodríguez, E., Gómez, R.: Experiments in term expansion using thesauri in Spanish. In: CLEF 2002, LNCS. vol. 2785, pp. 301–310 (2003)

Evaluación de los sistemas QA de dominio abierto frente a los de dominio especializado en el ámbito biomédico

María-Dolores Olvera-Lobo[1,2], Juncal Gutiérrez-Artacho[3]

[1] CSIC, Unidad Asociada Grupo SCImago, Madrid, España

[2] Departamento de Biblioteconomía y Documentación,
Universidad de Granada, Granada, España

[3] Departamento de Traducción e Interpretación,
Universidad de Granada, Granada, España

{molvera, juncalgutierrez}@ugr.es

Resumen. Los sistemas QA se presentan como una alternativa a los sistemas tradicionales de recuperación de información tratando de ofrecer respuestas precisas a preguntas factuales. Hemos realizado un estudio para evaluar la eficiencia de estos sistemas como fuentes terminológicas para los especialistas y para usuarios en general. Con este fin, se ha evaluado el funcionamiento de cuatro sistemas QA, dos especializados en el dominio biomédico (MedQA y HONqa) y dos de dominio general (START y QuALiM). El estudio ha utilizado una colección de 150 preguntas biomédicas definicionales (*What is…?)*, obtenidas del sitio web médico WebMD. Para determinar el funcionamiento, se han evaluado las respuestas ofrecidas utilizando una serie de medidas específicas (precisión, MRR, TRR, FHS).
El estudio permite confirmar que los cuatro sistemas son útiles para la recuperación de información definicional en este ámbito, ya que han proporcionado respuestas coherentes y precisas con un grado de aceptabilidad adecuado.

Palabras clave: Sistemas QA de dominio abierto, Sistemas QA de dominio especializado, Evaluación del funcionamiento, Información biomédica

1 Introducción

En el entorno de la Web la sobrecarga de información se deja sentir aún más que en otros contextos. De esta forma, en demasiadas ocasiones, al plantear una determinada consulta en las herramientas de búsqueda de información web (buscadores, directorios o metabuscadores) el número de páginas web recuperadas resulta excesivo y no todas ellas son relevantes ni útiles para los objetivos del usuario. Por ello, los profesionales de diversos ámbitos comienzan a reconocer la utilidad de otros tipos de sistemas, como los

sistemas de búsqueda de respuestas (en inglés *question-answering systems*, en adelante sistemas QA), como método para la obtención de información especializada de forma rápida y efectiva [1-3].

La Recuperación de Información (en adelante, RI) se ha entendido como el proceso, totalmente automático, en el que dada una consulta (que, supuestamente, expresa la necesidad de información del usuario) y una colección de documentos, el sistema devuelve una lista ordenada de documentos potencialmente relevantes para esa consulta [4]. Un sistema de RI con funcionamiento óptimo recuperaría todos los documentos relevantes (lo que implica una exhaustividad completa) y sólo aquellos documentos que son relevantes (precisión perfecta). El modelo tradicional de RI lleva consigo muchas restricciones implícitas tales como: a) la suposición de que los usuarios del sistema buscan documentos (textos completos), no respuestas, y que son los documentos, como tales, los que responden y satisfacen una consulta; b) que el proceso debe ser directo y unidireccional en lugar de interactivo; y c) que la consulta y el documento están escritos en la misma lengua.

Un paso en la evolución hacia la mejora de la RI son los sistemas QA. Se presentan como una alternativa a los tradicionales sistemas de RI tratando de ofrecer respuestas precisas y comprensibles a preguntas factuales, en lugar de presentar al usuario una lista de documentos relacionados con la búsqueda [5], de modo que el usuario no ha de leer documentos completos para obtener la información requerida. El desarrollo de los sistemas QA toma un importante impulso en el seno de la conferencia sobre recuperación de información TREC (*Text REtrieval Conference)* −principalmente a partir de TREC-8 [6]− la cual, desde 1992, constituye un foro internacional para aunar e incentivar la investigación en diferentes ámbitos de la recuperación de información.

Según un estudio de Ely y otros [7], los especialistas médicos tardan más de dos minutos de promedio en buscar información relativa a las preguntas que les surgen y, a pesar del tiempo empleado, muchas de ellas no consiguen obtener la respuesta adecuada. En este sentido, varios trabajos han demostrado la confianza de los especialistas médicos en el uso los sistemas QA como método de búsqueda y recuperación de información especializada [2,8], así como que los pacientes también han aumentado sus consultas en estos sistemas antes y después de ver al médico para obtener información sobre la naturaleza de la enfermedad, las indicaciones y contraindicaciones de los tratamientos, entre otros [9].

El funcionamiento de los sistemas QA se basa en los modelos de respuestas cortas [10] puesto que ofrece la respuesta potencialmente correcta en forma de un número, un sustantivo, una frase corta o un fragmento breve de texto. Existen diferentes patrones a la hora de plantear las preguntas en los sistemas QA, la mayoría se caracterizan por aceptar preguntas expresadas a través de partículas interrogativas (qué, cómo, quién, por qué, cuándo, dónde), o de forma imperativa. Los sistemas QA proceden a la construcción de respuestas coherentes expresadas en lenguaje natural [11]. Planteada la pregunta en el motor de búsqueda del sistema, éste procede a analizar la pregunta separando la palabra o palabras claves y determinando el tipo de respuesta esperada, luego se localiza y extrae una respuesta a partir de diferentes fuentes –dependiendo de la cobertura temática del sistema se utilizarán unas u otras fuentes de información [12]–, y finalmente, se evalúa y elimina aquella información redundante o que no responde correctamente a la pregunta planteada para, posteriormente, elaborar y presentar una o

varias respuestas concretas que supuestamente satisfacen la necesidad del usuario [13,14]. Una de las dimensiones más importantes de estos sistemas QA es el proceso de la evaluación de las respuestas, ya que además de evaluarlas, las compara y ordena [15]. Algunos de estos sistemas utilizan algún módulo destinado a la comparación de las preguntas planteadas por el usuario con el pasaje del documento o documentos seleccionados que, potencialmente, recogen la respuesta adecuada [16,17].

Los sistemas QA suelen tener una sencilla interfaz con un motor de búsqueda en el que los usuarios plantean su pregunta, algunos de ellos facilitan la lista de las últimas cuestiones introducidas para facilitar al usuario la comprensión acerca del funcionamiento del mismo. Para el tratamiento y gestión de las preguntas, los sistemas QA aplican algoritmos y métodos de análisis lingüístico y de procesamiento del lenguaje natural con el fin de identificar sus componentes y determinar el tipo de respuesta esperada [9]. Este análisis consiste normalmente en utilizar una variedad de tipos de preguntas estándar en los que se reemplazan ciertas palabras por las etiquetas aceptadas por el sistema [18].

Los sistemas QA pueden ser de dominio general –si puede atender consultas de temas muy diversos, como START [19] o QuALiM [20] – o de dominio específico, si se centran en un ámbito determinado, como MedQA [21] o HONqa [22]. Otro de los aspectos clave de estos sistemas es que el establecer una relación sistema-usuario no unidireccional y una interacción en el proceso de búsqueda ayudaría al sistema a encontrar mejores respuestas, y al usuario a encontrar la respuesta más rápidamente. No obstante, todavía es necesario profundizar en el diseño de estos sistemas interactivos que hagan posible la existencia de un verdadero *feedback* entre preguntas y respuestas, y que el usuario se comunique a nivel conversacional con el sistema.

A pesar del avance que supone el poder contar con herramientas de búsqueda de información de este tipo, los sistemas QA presentan algunas restricciones como que muchos de los sistemas han sido desarrollados únicamente como prototipos, o *demos*, y sólo en casos muy poco frecuentes se han comercializado.

Si bien en los últimos años se han analizado diversos aspectos de los sistemas QA, aún son escasos los estudios que evalúan el funcionamiento de estas herramientas. Estos sistemas deben generar frases con definiciones dinámicas y coherentes que contengan y resuman la información más descriptiva que posean en su colección de documentos sobre el término o foco de la pregunta del usuario [23, 13].

En este trabajo hemos llevado a cabo un estudio para evaluar la eficiencia de los sistemas de búsqueda de respuestas como fuentes terminológicas para los especialistas y para usuarios en general. Con este fin, se ha analizado y evaluado el funcionamiento de cuatro sistemas de búsqueda de respuestas, dos especializados en el dominio biomédico (MedQA y HONqa) y dos de dominio general (START y QuALiM). Para ello se ha utilizado una colección de 150 preguntas biomédicas definicionales y se han evaluado las respuestas ofrecidas utilizando medidas específicas aplicables a este tipo de sistemas considerando además las fuentes utilizadas por éstos para extraer tales respuestas. A continuación se describe con detalle la metodología utilizada y los principales resultados obtenidos.

2 Metodología

En nuestro análisis se utilizaron 150 preguntas de definición sobre diversos temas médicos. La colección de preguntas utilizadas se ha obtenido tras plantear la expresión "*What is*" en el motor interno de búsqueda del sitio web WebMD [24], un portal estadounidense creado por especialistas médicos para dar respuesta a las incertidumbres de los pacientes y en el que se ofrece información sobre una amplia lista de temas médicos de diferente grado de especialización. El sitio web proporcionó más de 6000 preguntas. Finalmente, seleccionamos para nuestro estudio las 150 preguntas que obtuvieron respuesta en los cuatro sistemas analizados, los cuales se eligieron debido a que sus bases de datos presentan una extensa cobertura y están actualizadas. El conjunto de preguntas utilizadas superó el test de validez interna con un alfa de Cronbach de 0,997.

Los sistemas QA utilizados fueron: START, QuALiM, MedQA y HONqa. START, desarrollado por el *Massachusets Institute of Technology* es un sistema que permite a los usuarios plantear preguntas sobre temas diversos por lo que también debe ser capaz de responder preguntas especializadas de dominio médico [25]. Las fuentes de información de las que extraen las respuestas son muy variadas, entre las que se encuentran página o sitios web generales como *Wikipedia*, diccionarios de uso general, *Internet Public Library*, *WorldBook*, *The World Factbook 2008*, entre otros, así como otras páginas o sitios especializados en una determinada área como diccionarios y enciclopedias especializados, etc. Por su parte, el segundo sistema QA de dominio abierto analizado fue QuALiM. Este sistema, financiado por *Microsoft*, recupera tanto información textual −para lo que utiliza únicamente la enciclopedia *Wikipedia*− como gráfica −extraída del buscador de imágenes de Google− [26].

Por su parte, MedQA, un sistema QA de dominio especializado desarrollado en la Universidad de Columbia se trata de un sistema especializado que analiza miles de documentos para generar un párrafo que responda correctamente a preguntas especializadas en temas médicos, por lo que sus fuentes de información difieren parcialmente de los anteriores [8]. Utiliza una amplia gama de fuentes de información, entre las que se encuentran *Wikipedia*, *Medline*, *Medline Plus*, un par de ellas más. Por último, HONqa está desarrollado por la *Health On the Net Foundation*, una organización suiza sin ánimo de lucro cuyo objetivo es promover la creación de información médica de calidad y veraz. Es el único sistema multilingüe ya que se puede recuperar información en inglés, italiano y francés [27]. Las fuentes de información que utiliza son muy variadas y suelen ser portales médicos generales o especializados en una determinada enfermedad.

Tras plantear las preguntas en los cuatro sistemas, varios profesionales médicos valoraron las respuestas como incorrectas, inexactas o correctas. Se consideró una respuesta correcta aquélla que respondía adecuadamente a la pregunta planteada, no utilizaba más de 100 palabras para generar la respuesta y no contenía información que no fuera relevante a la pregunta. Todas las respuestas que, a pesar de contestar correctamente a la pregunta no se ajustaban al resto de criterios, fueron valoradas como inexactas. La valoración realizada para cada respuesta sirvió de base para la aplicación de las medidas de evaluación del funcionamiento de los sistemas [28], las cuales se describen a continuación.

Mean Reciprocal Rank (MRR) asigna el valor inverso de la posición en la que la respuesta correcta fue encontrada (1 si es la primera, ½ si es la segunda, ¼ si es la cuarta, y así sucesivamente), o cero si la respuesta correcta no fue encontrada. Esta medida únicamente considera la primera respuesta correcta encontrada en la lista de resultados ofrecidos por el sistema, y el valor final es el promedio de los valores obtenidos para cada pregunta. MRR asigna un valor alto a las respuestas que están en las posiciones más altas de la clasificación.

$$MRR = \frac{1}{q}\sum_{i=1}^{q}\frac{1}{far_i}$$

Total Reciprocal Rank (TRR) es útil para evaluar la existencia de varias respuestas correctas ofrecidas por un sistema ante una misma pregunta. En estos casos no es suficiente considerar únicamente la primera respuesta correcta en las evaluaciones, por lo que TRR las tiene todas en cuenta y asigna un peso a cada respuesta de acuerdo con su posición en la lista de resultados recuperados. Así, si la segunda y la cuarta respuesta de una lista de resultados son correctas para una pregunta el valor de TRR será ½ + ¼ = ¾.

First Hit Success (FHS) asigna valor 1 si la primera respuesta ofrecida es correcta, y valor 0 si no lo es (por lo que sólo considera la respuesta que aparece en primer lugar en la lista de resultados).

Además, se utilizó una medida clásica en la evaluación de la recuperación de información, la precisión. Ésta se define como la capacidad del sistema para recuperar documentos (o respuestas, en el caso de los sistemas QA) que sean relevantes a las consultas (o preguntas) planteadas y que estén bien ordenados (en el caso de que los sistemas establezcan un ranking de resultados).

$$Precisión = \frac{Número\ de\ documentos\ relevantes\ recuperados}{Número\ total\ de\ documentos\ recuperados}$$

3 Resultados y Discusión

Tras plantear las 150 preguntas en los cuatro sistemas QA se analizaron las cinco primeras respuestas de cada uno de estos sistemas, al ser el promedio de respuestas recuperadas por la mayoría de ellos y debido a que los usuarios, pretendidamente, utilizarían este tipo de sistemas para la recuperación rápida de información, centrando su atención en las primeras respuestas. Para ciertas preguntas sin embargo, algunos sistemas QA ofrecieron un número superior y otros no llegaron a ofrecer las cinco respuestas.

El volumen de respuestas recuperadas por los sistemas de dominio abierto es inferior a los de dominio restringido, siendo el menor el de START (con 1,6 respuestas como media) seguido de cerca de QuALiM (con 3 respuestas). En los sistemas QA de dominio especializado los resultados aumentan sustancialmente, sobre todo en el caso de HONqa (con 44,23 respuestas), mientras que en MedQA es ligeramente superior a la de los sistemas QA de dominio abierto (5,34 respuestas a cada pregunta de promedio).

Las respuestas correctas están presentes en mayor medida en START (70,08%), en los dos sistemas QA de dominio especializado este promedio desciende,−MedQA (46,67%) y HONqa (47,25%)− y QuALiM se presenta como el más deficiente con el 40,89% de respuestas correctas. La figura 1 muestra el número de respuestas correctas, inexactas e incorrectas de los cuatro sistemas QA analizados.

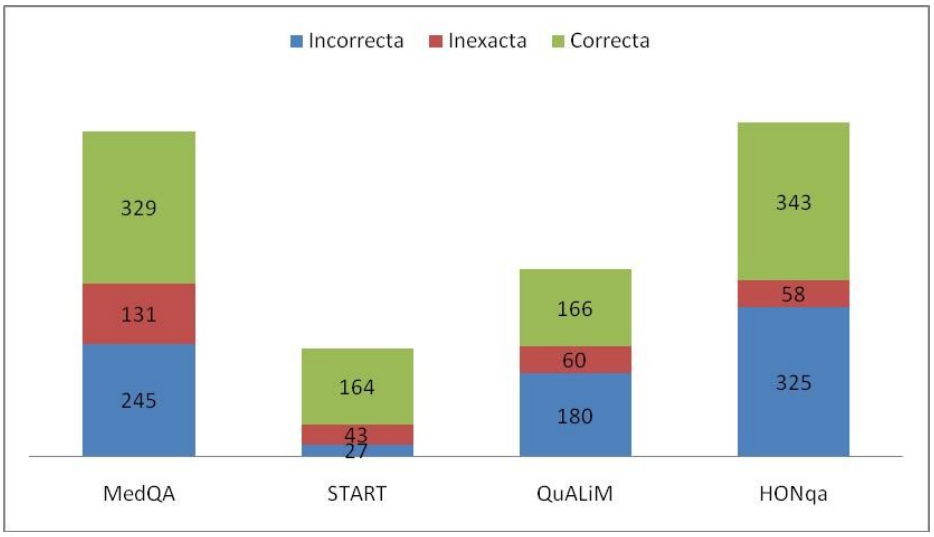


Figura 1. Respuestas incorrectas, inexactas y correctas

En relación a las respuestas inexactas, sistema QA con el promedio inferior es HONqa (7,97%), ofreciendo el resto de los sistemas QA un promedio similar (MedQA, 18,58%; START, 18,38%; QuALiM, 14,78%).

Las respuestas incorrectas en HONqa (44,78%) y QuALiM (44,33%) alcanzan una cifra similar. En MedQA es algo inferior (34,75%) aunque continúa siendo elevada siendo START (11,54%) el sistema que presenta menos respuestas inexactas.

El valor obtenido al aplicar la medida de evaluación MRR y FHS indica que MedQA ordena mejor las respuestas, de manera que la primera respuesta correcta aparece en los primeros lugares de la lista de resultados. Esto es significativo puesto que no emplea ningún algoritmo de ranking para llevar a cabo este proceso sino que siempre recurre a la misma ordenación de las respuestas, según de la fuente de la que provengan.

FHS resulta una medida muy relevante puesto que los usuarios, en muchas ocasiones, tienden a centrarse en la primera respuesta recuperada obviando el resto.

Tabla 1. Medidas de evaluación de los sistemas QA

	MRR	**TRR**	**FHS**	**Precisión**
MedQA	0,87	1,29	0,76	0,54
START	0,67	0,81	0,64	0,76
QuALiM	0,65	0,77	0,59	0,56
HONqa	0,75	1,15	0,55	0,46

En la situación contraria nos encontramos con el otro sistema QA de dominio especializado, HONqa, ya que ofrece el FHS con valor más bajo, aún cuando se trata del segundo sistema con un promedio superior en MRR. El comportamiento de los dos sistemas QA de dominio abierto es muy similar y no hay una gran diferencia entre ambas medidas, lo que se explica si se tiene en cuenta que, para cada pregunta, las respuestas recuperadas suelen oscilar entre 1 y 3.

No obstante, la medida TRR es superior en los sistemas de dominio especializado puesto que pondera el valor de cada respuesta correcta en función del lugar que ocupa en la lista de resultados. Necesariamente, este valor aumente, al ofrecer un mayor número de resultados.

El valor de la precisión en los sistemas QA de dominio abierto −principalmente en START− ha sido superior al de los sistemas QA de dominio especializado ya que éstos últimos presentan una alta tasa de ruido documental.

Como se puede observar, ninguna de las medidas aplicadas presenta valores muy altos. Esta circunstancia se ha visto claramente influida por el hecho de que las condiciones requeridas para evaluar una respuesta como correcta tenían un alto nivel de exigencia. En muchas ocasiones, por ejemplo en MedQA y en START se encontraron respuestas correctas con más de 100 caracteres, lo que provocó que fueran consideradas como inexactas, como se indicó en el apartado Metodología. En Honqa, por ejemplo, se encontraron respuestas correctas que, al no responder a la pregunta de forma totalmente concreta y precisa, fueron consideradas también inexactas. La siguiente tabla muestra la correlación existente entre las métricas utilizadas en este estudio.

Tabla 2. Correlación entre las medidas

		MRR	**TRR**	**FHS**
MRR	Pearson Correlation	1,000	,959*	,700

	Sig. (2-tailed)		,041	,300
	N	4,000	4	4
	Pearson Correlation	,959*	1,000	,472
	Sig. (2-tailed)	,041		,528
	N	4	4,000	4
FHS	Pearson Correlation	,700	,472	1,000
	Sig. (2-tailed)	,300	,528	
	N	4	4	4,000
Precisión	Pearson Correlation	-,450	-,606	,201
	Sig. (2-tailed)	,550	,394	,799
	N	4	4	4

* La correlación es significativa al nivel 0,05.

La tabla muestra que las medidas que tienen alta correlación son MRR y FHS y por otro lado TRR y precisión, de manera inversa, pero tan sólo MRR y TRR tienen una correlación significativa ($p<0,05$).

4 Conclusiones

El análisis de los resultados obtenidos al plantear las 150 preguntas en los sistemas QA, MedQA, START, QuALiM y HONqa, ha permitido evaluar su funcionamiento aplicando métricas específicas. A pesar de las restricciones que muestran estos sistemas, ya que no son accesibles para todos y no se encuentran siempre desarrollados completamente, se ha comprobado que los cuatro sistemas QA son válidos y útiles para la recuperación de información definicional médica, puesto que ofrecieron respuestas coherentes y precisas.

Otro aspecto interesante se refiere a las fuentes de información utilizadas por cada uno de estos sistemas QA. Si comparamos las fuentes utilizadas por los sistemas QA de dominio especializado con aquellas utilizadas por los sistemas QA de dominio general, vemos grandes diferencias en la tipología y especialización, como era de esperar. Al comparar las fuentes utilizadas por START y QuALiM comprobamos que no existe gran diferencia, ya que ambos utilizan Wikipedia como fuente principal, aunque START se nutre de más fuentes para su recuperación. Sin embargo, en las fuentes de los dos sistemas QA de dominio especializado vemos grandes diferencias en las fuentes. Mientras que MedQA utiliza diccionarios, enciclopedias y bases de datos especializados

en el dominio biomédico, HONqa se decanta por sitios web especializados en este dominio.

Los resultados son esperanzadores al mostrar este tipo de herramientas como una nueva posibilidad en el ámbito de la recuperación de información precisa, fiable y concreta en un periodo breve de tiempo. En este sentido, algunos autores [29,22] han explorado varias posibilidades de mejora, tales como el uso de ontologías, las cuales contribuyen a elevar el nivel de calidad de las respuestas obtenidas puesto que formaliza la información relevante del dominio en cuestión.

5 Bibliografía

1. Crouch, D., Saurí, R., Fowler., A.: AQUAINT Pilot Knowledge-Based Evaluation: Annotation Guidelines. Tech. rep., Palo Alto Research Center (2005)

2. Lee, M., Cimino, J.; Zhu; H.R., Sable; C., Shanker; V., Ely, J., Yu., H.: Beyond Information Retrieval – Medical Question Answering. En: AMIA (2006)

3. Yu, H., Lee, M., Kaufman, D., Ely;, J., Osheroff, J.A., Hripcsak, G., Cimino, J.: Development, implementation, and a cognitive evaluation of a definitional question answering system for physicians. Journal of Biomedicine Informatics. 4, pp. 236–251 (2007)

4. Baeza-Yates, R.; Ribeiro-Nieto, B.: Modern Information Retrieval. Nueva York: ACM Press; Addison-Wesley (1999)

5. Pérez-Coutiño, M., Solorio, T., Montes-y-Gómez, M., López-López, A., Villaseñor-Pineda, L.: Toward a Document Model for Question Answering Systems. En Proceedings of the Second International Atlantic Web Intelligence Conference. AWIC 2004. Cancun, México (2004)

6. Voorhees, E.M.: The TREC 8 Question Answering Track Report. En Voorhees, E.M. y D.K. Harman (eds.), Proceedings of the 8th Text REtrieval Conference, vol. 500–246, pp. 107–130. NIST, Gaithersburg, Maryland (1999)

7. Ely, J.W., Osheroff, J., Gorman, P.N., Ebell, M.H., Chambliss, M.L., Pifer, E.A., Stavri., P.Z.: A taxonomy of generic clinical questions: classification study. BMJ, vol. 321, pp. 429–432 (2000)

8. Yu, H., Kaufman., D.: A cognitive evaluation of four online search engines for answering definitional questions posed by physicians. Pacific Symposium on Biocomputing, vol. 12, pp. 328–339 (2007)

Evaluación de técnicas de Aprendizaje Activo para codificación CIE-9-MC de informes de alta hospitalaria

David Lojo[1], David E. Losada[2], Álvaro Barreiro[3]

[1] Servicio de Informática. Complexo Hospitalario Universitario de Santiago
Santiago de Compostela, Spain
[2] Grupo de Sistemas Inteligentes, Dep. de Electrónica y Computación
Universidade de Santiago de Compostela, Spain
[3] IRLab. Dep. de Computación
Universidade da Coruña, Spain

Abstract.
El Aprendizaje Activo es una técnica según la cual, a partir de un conjunto de documentos sin etiquetar, se ordenan y seleccionan los documentos para ser etiquetados de modo que el nuevo conjunto de entrenamiento mejore el clasificador construido. En los hospitales se genera un gran volumen de información, pero sólo se codifica una pequeña parte de los informes producidos. Es por tanto un escenario donde se necesita elegir bien lo que se etiqueta para que las herramientas automatizadas de clasificación puedan surtirse de buenos conjuntos de entrenamiento. En nuestro trabajo, vamos a utilizar técnicas de Aprendizaje Activo para elegir los informes de alta hospitalaria que se deben etiquetar con códigos CIE-9-MC y, a continuación, evaluaremos la calidad de ese proceso de selección. Los documentos se representan utilizando técnicas populares en Recuperación de Información y la calidad de los conjuntos de entrenamiento se evalúa utilizando clasificación con Máquinas de Soporte Vectorial. El dominio clínico donde trabajamos es muy complejo, con un gran número de clases, y existe desbalanceo y poca independencia entre las clases. Los resultados de experimentación demuestran que nuestra estrategia es prometedora para mejorar este tipo de sistemas.

1 Introducción

En los hospitales se genera un gran volumen de información con considerable complejidad. La capacidad de clasificación manual es limitada por lo que es imposible que todos los documentos producidos sean etiquetados. Una de las tareas de clasificación que se realizan es la codificación de los diagnósticos de los informes de alta. La codificación es un proceso que consiste en analizar la documentación del alta, y asignar los códigos de los diagnósticos de ese episodio clínico. Este proceso se realiza de forma manual por un médico codificador, con un gran coste por la complejidad del tipo de clasificación, ya que consiste en la asignación de algunos de

los más de 21.000 códigos que tiene el CIE-9-MC [1] de la Organización Mundial de la Salud (OMS).

Se trata de un problema de clasificación con múltiples clases. Un documento puede pertenecer a varias clases, y el número de clases es variable para distintos documentos. En los hospitales, los episodios que se codifican habitualmente son los ingresos hospitalarios. Otros episodios clínicos no son codificados (por ejemplo los que corresponden a episodios de consultas externas, urgencias, pruebas funcionales, etc.). Actualmente el porcentaje de episodios clínicos que se codifican con respecto al total es mínimo. Si quisiésemos codificar todos los episodios clínicos que se generan en un centro hospitalario, tendríamos que aumentar de forma considerable los recursos humanos de médicos codificadores, lo que implicaría un elevado coste económico.

Debido a estas limitaciones los episodios clínicos pasan usualmente por una clasificación generalista, simplemente para generar una contabilidad básica, sin considerar la patología tratada para cada paciente. En cambio, con la codificación CIE-9-MC completa de estos episodios podríamos medir, comparar y mejorar la calidad asistencial, agrupando a los pacientes de acuerdo a requerimientos y características comunes.

En la literatura existen propuestas de clasificación automática para dar soporte a la codificación de informes [2,3,4]. Para poder aplicar técnicas de clasificación automática a los episodios clínicos, debemos elegir bien los episodios que actúan como entrenamiento para el clasificador. Para ello proponemos aplicar técnicas de Aprendizaje Activo (AA), para seleccionar aquellos documentos con los cuales poder entrenar un clasificador con resultados eficaces. El aprendizaje activo [5,6] es una técnica empleada en el entrenamiento de clasificadores que a partir de un conjunto de documentos sin etiquetar, escoge los documentos más informativos para la construcción del clasificador, obteniendo un conjunto etiquetado con una alta capacidad discriminante.

En este artículo evaluamos AA sobre una colección real del dominio clínico que presenta alto desbalanceo entre clases y en la que el problema de clasificación es difícil. Nuestros experimentos demuestran que AA se puede utilizar en este escenario con resultados razonables.

El resto del documento está organizado de la siguiente forma. Las estrategias de aprendizaje activo para la clasificación de textos multietiqueta se describe en la sección 2. En la sección 3 se concreta el método utilizado, para terminar con los experimentos en la sección 4 y las conclusiones en la sección 5.

2 Aprendizaje activo para la clasificación de textos multietiqueta

El Aprendizaje Activo (Active Learning), consiste en seleccionar los ejemplos más informativos entre los no etiquetados con la finalidad de etiquetarlos y agregarlos al conjunto de entrenamiento. Con esta técnica intentamos reducir el coste a la hora de obtener una colección etiquetada mediante la selección de los mejores documentos para el sistema de aprendizaje.

En muchas ocasiones en un esquema de aprendizaje supervisado la colección de entrenamiento es pequeña, y es muy costoso obtener nuevos documentos etiquetados. Sin embargo, se suele disponer de una gran cantidad de documentos sin etiquetar. Esta es la situación que nos encontramos en los hospitales en donde para ciertas áreas clínicas no disponemos de una colección etiquetada de sus episodios. Además la etiquetación manual es costosa y ha de ser realizada por expertos.

Para conseguir la colección de entrenamiento aplicamos AA en un entorno multietiqueta. Dado un conjunto de clases predefinidas $C = \{C_1, \ldots, C_m\}$, y una colección de documentos a clasificar *(D)*, el objetivo es encontrar una función con la forma $\varphi : D \times C \rightarrow \{-1,+1\}$, que denominamos clasificador (-1 y +1 representan la pertenencia o no a una clase). Un documento puede no pertenecer a ninguna clase, a una clase o a varias clases. En una clasificación de textos multietiqueta usualmente se generan *m* clasificadores binarios, uno por cada clase c_j. En nuestro trabajo nos centramos en clasificadores que tienen la forma $\varphi^* : D \times C \rightarrow [-1,+1]$. Esta función permite estimar la clase a la que el clasificador cree que el documento pertenece (signo de $\varphi^*(d_i,c_j)$) y, además proporciona una estimación de la confianza del clasificador en la decisión tomada, $|\varphi^*(d_i,c_j)|$.

Las técnicas de AA para entornos con una única clase consisten básicamente en escoger primero aquellos ejemplos no etiquetados sobre los que el clasificador automático tiene menos confianza. El codificador humano de esta forma etiquetará manualmente solo aquellos documentos más informativos para el aprendizaje. En situaciones como la nuestra, con múltiples clases, cada documento tiene un valor de confianza para cada clase por lo que es necesario combinar esas puntuaciones para estimar qué documento no etiquetado es globalmente más informativo para el proceso de clasificación multietiqueta.

En la literatura [7], nos encontramos dos opciones para seleccionar los documentos que vamos a incorporar al conjunto de entrenamiento. Por un lado, generar *m* rankings independientes de documentos (cada uno de ellos asociado a una clase). El médico tendría que codificar los documentos que figuran más arriba en el ranking para cada clase. Esta opción se denomina *etiquetado local*, ya que se realiza localmente en cada clase. La otra opción, consiste en generar un único ranking de documentos en base a la combinación de los *m* valores de confianza asociados a un mismo documento, y se denomina *etiquetado global*. En colecciones con un número elevado de clases, como ocurre con la colección utilizada en esta investigación, el etiquetado local obliga al médico codificador a trabajar con muchos rankings diferentes. Esto supone un gran esfuerzo para la etiquetación puesto que un documento puede aparecer en múltiples rankings y el codificador debe revisarlo cada vez para asignar, o no, la etiqueta de la correspondiente clase. En cambio, la opción de etiquetado global se adapta mejor a los requisitos del trabajo a desarrollar, pues el médico codificador revisa exclusivamente un único ranking y está garantizado que un documento se lee como máximo una vez.

En este trabajo vamos a comparar algunas estrategias propuestas en [7], para la obtención de un ranking de documentos en orden decreciente en cuanto a su capacidad informativa, dentro de una estrategia de etiquetado global. Las distintas variantes se definen a través de tres dimensiones: dimensión “evidencia”, dimensión “clase” y dimensión “peso”. Cada estrategia que apliquemos va a ser una

combinación de decisiones en estas tres dimensiones, y las representaremos con una secuencia de tres letras, en donde cada letra representa una elección en cada una de las dimensiones.

Antes de entrar en detalles sobre las dimensiones, conviene aclarar terminología: dado el clasificador $\varphi^*: D \times C \rightarrow [-1,+1]$, el valor $\varphi^*(d_i,c_j)$ lo denominaremos *puntuación* de la clase c_j para el documento d_i y el valor $|\varphi^*(d_i,c_j)|$ la *confianza* de la clase c_j para el documento d_i y $sgn(\varphi^*(d_i,c_j))$ será el signo de la categoría c_j para el documento d_i.

2.1 Dimensión "evidencia"

Esta dimensión hace referencia al tipo de evidencia que utilizamos como base para la estimación de lo informativo que es un documento para una clase. Una posibilidad es utilizar el valor $|\varphi^*(d_i,c_j)|$, que representa la ***Confianza*** (**C**) del clasificador de la clase c_j con respecto al documento d_i. La intuición es que cuanto menor sea el valor de la confianza con el clasificador actual, el documento aportará más información al clasificador tras ser etiquetado (esto es, tenderemos a etiquetar los documentos sobre los que haya más duda).

La otra alternativa es usar directamente la puntuación $\varphi^*(d_i,c_j)$ como evidencia. Aquí la intuición es otra, se trata de promover documentos que son claramente ejemplos positivos de la clase porque en muchas situaciones los casos positivos son los que ayudan más en escenarios supervisados. A esta opción le denominaremos ***Puntuación*** (en inglés *Score*, ***S***).

Tenemos pues dos estrategias, *confianza* y *puntuación*, a la hora de estimar lo informativo que es un documento para una clase. La siguiente dimensión, de clase, hace referencia a cómo se combinan los *m* valores de evidencia que tienen un documento en un único valor.

2.2 Dimensión "clase"

En la dimensión "clase" lo que se pretende es generar una valoración global para cada documento independiente de la clase. Una opción consiste en maximizar la capacidad informativa esperada calculada sobre todas las clases. Esto significa que si la dimensión de evidencia es Confianza (C), para la dimensión de "clase" se tomaría $\min_{c_j \in C} |\varphi^*(d_i,c_j)|$. Si la dimensión de evidencia es Puntuación (S), para la dimensión de clase escogeríamos $\max_{c_j \in C} \varphi^*(d_i,c_j)$. Intuitivamente pretendemos que el médico codificador se concentre en documentos que se consideran de gran valor por lo menos para una clase. Esta elección se denomina ***Min / Max (M)***.

Otra opción consiste en utilizar el promedio de todos los valores obtenidos para un documento en todas las clases. De este modo seleccionaremos los documentos útiles globalmente para el conjunto de clases. Esta aproximación se llama ***Promedio*** (***Avg, A***).

Una última opción, ***Round Robin (R)***, consiste en seleccionar los documentos mejores de cada clase de la siguiente forma: se toma el mejor documento para cada clase (según dimensión de evidencia) y se crea un ranking de documentos (como mucho de tamaño *m*, pues se eliminan repetidos) ordenados decrecientemente según evidencia; a continuación, se toman los segundos documentos de cada clase, se incorporan al final del ranking (ordenados entre sí por evidencia), y así sucesivamente.

2.3 Dimensión "Peso"

La dimensión "Peso", ***Weight (W)*** tiene la función de no tratar a todas las clases por igual. Uno de los objetivos es dar más peso a las clases en donde el clasificador obtiene peores resultados. Para ello utilizamos una función de evaluación $f(\phi_j)$ que tenga un valor entre [0,1] y que nos indica qué rendimiento tiene el clasificador automático para la clase c_j. Cuando estemos trabajando con Confianza (C), multiplicaremos el valor de la confianza $|\varphi^*(d_i,c_j)|$ por la función de evaluación $f(\phi_j)$, que indica la efectividad del clasificador en la clase. Para Puntuación (S) calcularíamos el producto de $\varphi^*(d_i,c_j)$ por $(1-f(\phi_j))$. Estos ajustes sobre los valores de evidencia consiguen promover aquellas clases que no dan buen rendimiento ($f(\phi_j)$ bajo)[1]. En nuestros experimentos, al igual que se hizo en [7], utilizaremos la conocida medida F_1 para definir $f(\phi_j)$, aplicando además un suavizado de Laplace ($\varepsilon=0.05$) para evitar multiplicaciones por 0. La alternativa de no utilizar pesos para las clases la denominaremos ***No Weight (N)***.

3 Metodología para evaluar Aprendizaje Activo

Para evaluar las estrategias de AA mantendremos una colección separada de documentos de test (*TestSet*) y crearemos incrementalmente una colección de entrenamiento cuya calidad iremos contrastando mediante la construcción de un clasificador automático a partir del conjunto de entrenamiento y su evaluación contra el *TestSet*.

En el proceso de clasificación vamos a utilizar una representación vectorial de los documentos en un espacio de características. En este trabajo hemos seleccionado una representación vectorial basada en el popular esquema de pesado *tf/idf*. Para clasificar utilizaremos la metodología aplicada en [3] sobre Máquinas de Soporte Vectorial (SVM). La implementación de SVM utilizada fue SVM^{Light} [8]. Para cada clase el clasificador SVM utilizado nos devuelve un valor que representa

[1] Para confianza se multiplica por $f(\phi_j)$ porque la elección de documentos se hace en orden creciente de evidencia. Para una clase *j* con pobre rendimiento ($f(\phi_j)$ bajo), $f(\phi_j)$ multiplicado por su confianza será bajo con lo que se favorecerá la elección de los documentos. Análogamente sucede con Puntuación al multiplicarle por $(1- f(\phi_j))$ y escoger los documentos en orden decreciente.

$\varphi^*(d_i,c_j)$. La distancia al hiperplano lo interpretamos como la puntuación y su valor absoluto como la confianza.

Sea D_{Todo} el conjunto de documentos etiquetados con el que vamos a surtir a la colección de entrenamiento. El algoritmo es como sigue:

1. Seleccionamos aleatoriamente 100 documentos de D_{Todo} que denominamos D_{Train}, en donde $|D_{Train}| = 100$ y $D_{Todo} = D_{Todo} - D_{Train}$.
2. Construimos *m* clasificadores SVM, un clasificador para cada clase, utilizando D_{Train} y los evaluamos con *TestSet.*
3. Aplicamos *AA* para seleccionar 50 nuevos documentos para incorporar a D_{Train}. Para ello, según la variante utilizada:

   a. *Evidencia de Confianza* y *Clase Min (CM)*

   $\forall d_i \in D_{Todo}$ calculamos $d_{score}=\min |\varphi^*(d_i,c_j)| \; \forall c_j \in \{C_1,\ldots, C_m\}$

   Realizamos un ranking creciente por d_{score} y seleccionamos el *Top 50*.

   b. *Evidencia de Puntuación* y *Clase Max (SM)*

   $\forall d_i \in D_{Todo}$ calculamos $d_{score}=\max \varphi^*(d_i,c_j) \; \forall c_j \in \{C_1,\ldots, C_m\}$

   Realizamos un ranking decreciente por d_{score} y seleccionamos el *Top 50*.

   c. *Evidencia de Confianza* y *Clase Promedio (CA)*

   $\forall d_i \in D_{Todo}$ calculamos $d_{score}=\text{avg} |\varphi^*(d_i,c_j)| \; \forall c_j \in \{C_1,\ldots, C_m\}$

   Realizamos un ranking creciente por d_{score} y seleccionamos el *Top 50*.

   d. *Evidencia de Puntuación* y *Clase Promedio (SA)*

   $\forall d_i \in D_{Todo}$ calculamos $d_{score}=\text{avg}\, \varphi^*(d_i,c_j) \; \forall c_j \in \{C_1,\ldots, C_m\}$

   Realizamos un ranking decreciente por d_{score} y seleccionamos el *Top 50*.

   e. *Evidencia de Confianza* y *Clase Round Robin (CR)*

   $\forall c_j \in \{C_1,\ldots, C_m\}$ *calculamos* $d_{Cj}= \min |\varphi^*(d_i,c_j)| \; \forall d_i \in D_{Todo}$

   Con el documento d_{Cj} seleccionado para cada clase realizamos un ranking de documentos por orden creciente de confianza, eliminamos los duplicados y seleccionamos el *Top 50*. Si no hay 50 documentos en el ranking se tomarían los siguientes documentos seleccionados por cada clase y así sucesivamente.

   f. *Evidencia de Puntuación* y *Clase Round Robin (SR)*

$\forall c_j \in \{C_1, ..., C_m\}$ calculamos $d_{Cj} = \max \varphi^*(d_i, c_j) \ \forall d_i \in D_{Todo}$

Con el documento d_{Cj} seleccionado para cada clase realizamos un ranking de documentos por orden decreciente de confianza, eliminamos los duplicados y seleccionamos el *Top 50*. Si no hay 50 documentos en el ranking se tomarían los siguientes documentos seleccionados por cada clase y así sucesivamente.

4. Los 50 documentos obtenidos los incorporamos a D_{Train}: $D_{Train} = D_{Train} \cup$ Top50 y $D_{Todo} = D_{Todo}$ - Top50.
5. Volver al paso 2 si $|D_{Train}| < 1000$.

Si además se quiere aplicar la dimensión de Peso, lo primero es calcular F_1 para cada clase, F_1^j. Estos valores se calculan con los clasificadores construidos en el punto 2 y evaluados con *TestSet.* Y este valor lo obtenemos de la siguiente forma:

$$F_1^j = \frac{2TP_j}{2TP_j + FP_j + FN_j} \quad \forall j \in \{1, ..., m\}$$

Si $TP_j = FP_j = FN_j = 0$ entonces $F_1^j = 1$. Donde TP, FP Y FN son el número de positivos verdaderos, positivos falsos y negativos falsos, respectivamente, obtenidos por el clasificador.Al cálculo de F_1^j le aplicamos el suavizado de Laplace con un valor $\varepsilon=0.05$. Como se explicó antes estos valores se utilizan para ajustar en el proceso anterior las evidencias de los documentos en las clases.

Con este proceso incremental de creación de una colección de entrenamiento podemos elegir iterativamente los 50 mejores documentos para etiquetar. En nuestra evaluación comparamos estas estrategias de AA entre sí y, además, incluimos una alternativa aleatoria en la que se toman siempre 50 documentos al azar.

4 Experimentos

Para construir la colección realizamos un estudio de los servicios que trabajan con informes de alta mediante documento informatizado. Desde este análisis elegimos el servicio de Medicina Interna del Hospital de Conxo, que forma parte del *Complejo Hospitalario Universitario de Santiago*, España. Las razones para esta selección son el alto número de documentos, el gran tamaño de los documentos, la utilización de división de párrafos homogénea y la complejidad de los diagnósticos utilizados.
La asignación de códigos CIE-9-MC a un episodio clínico tiene los siguientes elementos importantes:

- El diagnóstico principal (DxP) es la enfermedad que tras su estudio y en el momento del alta, el médico que atendió al paciente establece como causa del ingreso.

- Los diagnósticos secundarios (DxS) se consideran aquellas enfermedades que coexisten con el DxP en el momento del ingreso o que se han desarrollado durante la estancia hospitalaria y que han influido en la duración del ingreso.

La asignación de códigos CIE-9-MC es un problema multietiqueta, pero SVM se diseñó para realizar clasificación binaria. Es posible resolver los problemas de entornos SVM multiclase, basándose en la combinación de clasificadores binarios. Estas técnicas descomponen el problema multiclase en múltiples problemas binarios. Las dos principales alternativas que se aplican en la literatura para utilizar SVM cuando el número de clases, c, es superior a dos, son *1-vs-todos* y *1-vs-1*.

- *1-vs-todos* descompone un problema multiclase con *c* clases en otros tantos problemas binarios, en donde cada una de las clases se enfrentan al resto, generando *c* clasificadores. Un nuevo documento a clasificar se ordena por aquellas clases sobre las que el clasificador maximice el margen.
- *1-vs-1* descompone el problema d*e c* clases en (c*(c-1))/2 problemas binarios, donde se crean todos los clasificadores binarios posibles de los enfrentamientos uno a uno entre las clases. Cada documento a clasificar se somete a todos estos clasificadores, y se añade un voto a la clase ganadora, resultando un ranking de clases propuesta ordenada por el número de votos.

Si añadimos el factor de trabajar con una colección con un elevado número de clases, lo más recomendable es utilizar *1-vs-todos*.

La colección está compuesta de 1823 documentos. Mediante un reparto aleatorio obtenemos 1501 documentos para la colección de entrenamiento y 322 para la colección de test. La colección de entrenamiento tiene 1238 clases diferentes y la colección de test 544 clases. En la tabla 1 detallamos las características básicas de la colección.

Tabla 1. Propiedades de la colección

	Entrenamiento	Test
# de documentos	1501	322
Tamaño	5963 Kb	1255 Kb
Media de # códigos por documento	7.06	7.05
Máximo # de códigos por documento	23	19
Media de términos por documento	519.5	508.1
Min–Max # términos en documento	64 - 1386	109 - 1419

Las métricas de evaluación requieren la existencia de un *gold standard*. En nuestro caso, conocemos los códigos correctos por cada documento de test, porque cada documento de entrenamiento o test tiene una lista de clases asignada por el

médico codificador. Por supuesto, la lista de los códigos asociados a los documentos de test sólo se usa para fines de evaluación. Adoptamos las siguientes métricas, que se han utilizado en el pasado para la evaluación de los clasificadores de documentación clínica [2,3,4]:

- *Top candidato:* proporción de casos en los que el código principal es el principal candidato (*top 1*) propuesto por el sistema automático.
- *Top 10:* proporción de casos en los que el código principal está en los primeros 10 candidatos producidos por el sistema.
- *Recall 15 y recall 20:* Nivel de recall en los primeros 15 o 20 candidatos.

Utilizamos una cadena de tres letras para cada uno de las variantes de AA. Por ejemplo, CAN es la combinación de elegir *Confianza (C)* para la dimensión "evidencia", *Avg(A)* para la dimensión "clase" y (*N)* indica que no se utiliza la dimensión "peso". Con los distintos tipos de dimensiones posibles obtenemos 12 combinaciones posibles, que generan otros tantos experimentos.

Tabla 2. Resultados *Top candidato*

#Docs.	CMN	SMN	CAN	SAN	CRN	SRN	CMW	SMW	CAW	SAW	CRW	SRW	Aleatorio
100	6.52	6.52	6.52	6.52	6.52	6.52	6.52	6.52	6.52	6.52	6.52	6.52	6.52
150	8.07	8.07	6.52	7.76	8.07	8.07	8.07	**8.39**	6.83	7.14	8.07	**8.39**	7.76
200	9.94	9.32	7.45	8.70	9.94	9.32	8.07	**10.25**	8.07	8.39	8.07	**10.25**	6.83
250	**11.49**	9.94	9.32	9.94	**11.49**	9.94	9.94	9.32	7.45	10.25	9.94	9.32	10.87
300	11.18	**11.80**	11.49	12.42	11.18	**11.80**	10.87	**11.80**	11.49	**11.80**	10.87	**11.80**	10.87
350	12.11	14.29	12.42	13.04	12.11	14.29	11.18	13.98	**14.60**	13.66	11.18	13.98	12.11
400	13.04	**15.22**	13.04	13.04	13.04	**15.22**	13.04	14.29	13.98	13.98	12.73	14.29	11.80
450	13.66	13.98	13.66	14.91	13.66	13.98	12.42	**15.53**	14.60	13.98	12.11	**15.53**	13.04
500	14.60	13.35	14.91	15.22	14.60	13.35	**15.53**	14.91	**15.53**	15.22	15.22	14.91	12.42
550	14.91	15.84	15.53	16.15	14.91	15.84	16.46	**16.77**	16.46	16.15	16.15	**16.77**	11.80
600	14.60	16.77	17.08	**17.70**	14.60	16.77	15.84	17.39	16.77	16.46	15.53	17.39	12.73
650	16.46	16.46	17.70	**20.50**	16.46	16.46	17.70	16.46	17.39	16.77	17.39	16.46	12.42
700	16.46	16.46	18.32	**18.94**	16.46	16.46	17.70	17.08	17.70	16.77	17.39	17.08	12.73
750	15.84	17.08	19.25	**19.57**	15.84	17.08	15.84	17.39	18.63	16.77	15.53	17.39	12.42
800	16.15	16.46	19.25	**20.50**	16.15	16.46	17.39	17.39	18.32	17.70	17.08	17.39	13.66
850	16.15	16.46	18.63	**19.57**	16.15	16.46	17.08	18.01	19.25	16.77	17.08	18.01	14.60
900	17.70	17.08	**19.57**	18.32	17.70	17.08	18.01	18.63	18.94	15.84	18.01	18.63	15.53
950	18.32	17.39	**19.57**	17.70	18.32	17.39	18.94	18.32	18.63	16.77	18.94	18.32	14.60
1000	18.94	17.08	**20.50**	18.32	18.94	17.08	18.63	17.39	18.94	17.70	18.63	17.39	15.22

A los 12 experimentos de AA, tenemos que añadir un nuevo experimento, por el cual obtendremos de forma aleatoria los documentos que vamos incorporando a la colección de entrenamiento. De este modo podemos comparar el rendimiento entre

los modelos de *AA* y el modelo aleatorio (que supone una técnica base realista ya que es lo que ocurre en los hospitales pues no tienen criterios específicos para codificar episodios).

Los resultados de *Top candidato* para todos los experimentos se muestran en la tabla 2. Se muestran en negrita el mejor resultado para cada tamaño del conjunto de entrenamiento.

Los experimentos nos demuestran que con AA mejoramos *Top candidato*, en relación a una selección aleatoria a lo largo de todo el proceso. Tras incorporar 1000 documentos obtenemos mediante CAN un incremento del 34,7%, en comparación con el método aleatorio.

No existen mayores diferencias entre los distintos métodos de AA pero, en todo caso, CAN, SAN y SMW resultan ligeramente más robustos. Los resultados nos manifiestan que la dimensión de "peso", con pocos documentos en la colección de entrenamiento, obtiene los mejores resultados (por ejemplo, SMW y SRW). En cambio, a medida que el número de documentos de la colección de entrenamiento se incrementa, los resultados más favorables están en los experimentos que utilizan la dimensión de clase *Avg(A)* sin necesidad de realizar pesado por clases, como lo demuestran los resultados de SAN y CAN.

Con pocos documentos en la colección de entrenamiento, SMW (Puntuación – Máximo – Peso) funciona mejor que las otras combinaciones, en cambio al aumentar el número de documentos la dimensión *Avg(A)* nos facilita los mejores valores para cualquiera de las dimensiones de "evidencia", *Puntuación (S*) o *Confianza (C)*. Podemos deducir, que con pocos documentos, aquellos que pertenecen claramente a una clase, son los más representativos para el clasificador. En cambio, a medida que el número de documentos se incrementa, la media de los valores obtenidos para todas las clases, es más informativo para el clasificador. No podemos contrastar directamente estos resultados con otros experimentos [7] de AA con colecciones multiclase, ya que las métricas utilizadas no son las mismas y las colecciones son diferentes. Para nuestra colección la dimensión de peso no funciona bien cuando se combina con la dimensión de clase *Avg(A)*, en cambio consigue mejores resultados con la combinación de las otras dimensiones. Los gráficos 1, 2, 3 y 4 nos muestra la evolución de las métricas en función del número de documentos que vamos incorporando a la colección de entrenamiento para los experimentos SAN, CAN, SMW y aleatorio

Fig. 1. Top candidato

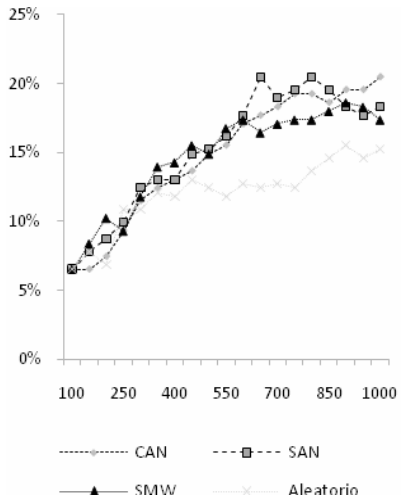


Fig. 2. Top 10

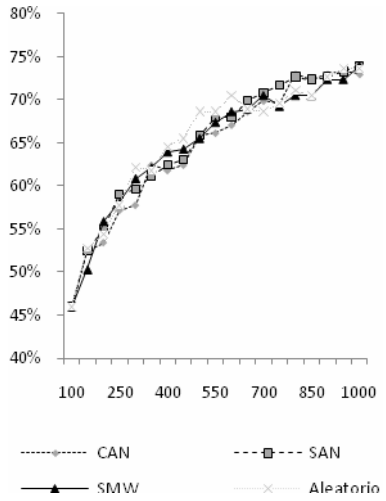


Fig. 3. Recall 15

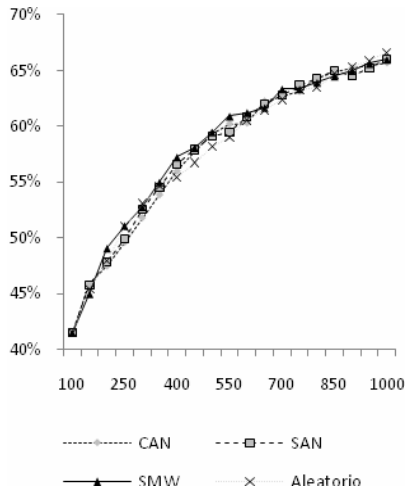
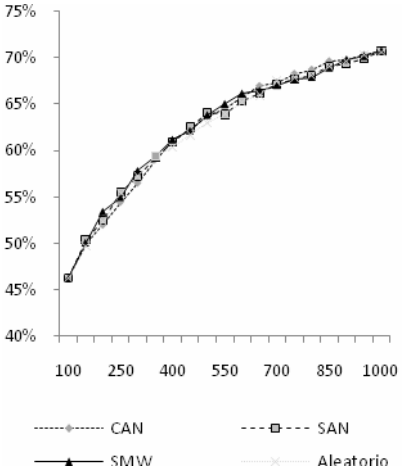


Fig. 4. Recall 20



5 Conclusiones

En los experimentos de aprendizaje activo realizados con todas las estrategias, observamos que solo *Top candidato* mejora claramente con respecto a una selección aleatoria. Para las otras métricas *Top 10, Recall 15* y *Recall 20* no se aprecia una mejora clara. Una de las posibles explicaciones es que las métricas *Top 10, Recall 15* y *Recall 20* están caracterizadas por definir un rango de posibles códigos validos sin importarle el lugar que estos ocupan dentro del ranking. Es decir para *Recall 20* tiene el mismo valor obtener un código correcto en la segunda posición dentro del rango de los 20 posibles códigos validos, que en la posición 19. Sin embargo, para el médico codificador la primera situación le demanda mucho menos esfuerzo. El aprendizaje activo mejora la posición en el rango de posibles códigos correctos, pero esta mejora no se transmite en el valor absoluto de estas métricas. Esto lo demuestra los resultados de *Top candidato* y *Top 10, y*a que, mejorando los códigos correctos que ocupan la primera posición (*Top candidato)*, este ascenso de posición del código correcto en el ranking, no se refleja en *Top 10*.

En nuestro trabajo futuro consideraremos también la posibilidad de utilizar otros clasificadores que sean más interpretables (en la línea de que el médico codificador pueda comprender mejor las sugerencias del sistema).

Agradecimientos

Agradecemos el apoyo económico de fondos FEDER, del Ministerio de Ciencia e Innovación y de la Xunta de Galicia a través de los proyectos de investigación con referencias TIN2008-06566-C04-04, 2008/068 y 07SIN005206PR.

Bibliografía

1. Clasificación internacional de enfermedades, 9ª revisión: modificación clínica CIE-9-MC. España. Ministerio de Sanidad y Consumo. ISBN 10: 84-340-1136-0
2. Larkey, L. and Croft, W. B., "Automatic Assignment of ICD9 Codes to Discharge Summaries," Center for Intelligent Information Retrieval Technical Report (1995).
3. D. Lojo, D. Losada, A. Barreiro. CIE-9-MC code Classification with knn and SVM. 3rd International Work-conference on the Interplay between Natural and Artificial Computation, IWINAC 2009, Santiago de Compostela (Spain), Jun 2009, 499-508, LNCS.
4. Larkey, L. and Croft, W. B. , "Combining Classifiers in Text Categorization," Proceedings of the 19th International Conference on Research and Development Information Retrieval (SIGIR96), Zurich, Switzerland, pp. 289-297
5. Cohn, D., Atlas, L., Ladner, R.: Improving generalization with active learning.Machine Learning 15(2), 201–221 (1994)
6. Tong, S., Koller, D.: Support vector machine active learning with applications to text classification. Journal of Machine Learning Research 2, 45–66 (2001)
7. Andrea Esuli and Fabrizio Sebastiani. Active Learning Strategies for Multi-Label Text Classification**.** Proceedings of the 31st European Conference on Information Retrieval (ECIR'09)*,* Toulouse, FR, 2009
8. Joachims, T.: Making large-scale SVM learning practical. In: Advances in Kernel Methods - Support Vector Learning. MIT press, Cambridge (1999)

A Descriptive Approach to Modelling Learning

Miguel Martinez-Alvarez, Fabrizio Smeraldi, and Thomas Roelleke

Queen Mary, University of London
{miguel, fabri, thor}@dcs.qmul.ac.uk

Abstract. The importance of Learning Algorithms in Computer Science and other fields has been increasing in the last years. At the same time, Descriptive Approaches have significantly impacted different domains, being SQL for data access and management one of the main examples. This paper investigates a descriptive approach for learning.
In the field of IR, learning techniques and descriptive approaches have already been applied independently. For the former, one of the most significant examples is the "Learning to Rank" task, while Probabilistic Datalog, which is a representative of descriptive approaches, has been applied for solving different IR-tasks, providing a high level representation of search strategies.
The main contributions of this paper are a knn classifier implemented in PDatalog, showing that the expressiveness of 2nd generation Probabilistic Datalog is sufficient for modelling lazy-learners, its evaluation for text classification on the Reuters-21578 collection, and a descriptive modelling of polynomial Mercer Kernels.

1 Introduction

Search tasks (information needs) are often more complex than captured by keyword-based query processing (e.g "Find the companies offering Probabilistic Reasoning Technology that are funded by the Government"). Even though there is a wide range of techniques available for classification, summarization and other tasks, combining those into a maintainable and scalable framework is a challenge.

Today's IR systems are often designed for a particular purpose. Therefore, the transfer and re-use of code is difficult, re-engineering seems the most effective process. This problem is comparable to what happened in the Software Industry, when Software Engineering evolved from implementing programs focused on specific tasks, to the development of frameworks for general tasks that could be adapted for specific ones, or, for the Database community, when SQL was adopted as the standard abstraction method for accessing information. In addition, the majority of IR systems are developed using traditional languages (e.g. C/C++, Java), commonly involving complicated knowledge transfer processes and maintainability. In a dynamic and technology-oriented environment a large amount of code is generated over time, and re-use and change management are challenging.

2 Background

2.1 Descriptive Approaches

We can assert that an approach is descriptive, also known as declarative, if "it involves stating what is to be computed, but not necessarily how it is to be computed" [10].

Descriptive approaches allow to define functionality in a high-level making the implementation clearer and the knowledge transfer easier. As a result, productivity will be increase [10]. In addition, they allow quick conceptual modifications in prototyping environments.

By using a descriptive approach it is possible to define different models and tasks as modules and then "concatenate" them, processing the information as a pipeline where the output of one module is the input of the following one. This solution does not involve any coding process because the modules selection is specified in the user interface level. This solution provides the flexibility needed for specifying and combine different IR tasks and/or models. Furthermore, it is possible to represent complex objects and structured data while maintaining scalability levels beyond any database system. Probabilistic Datalog (explained in Section 2.2) is the descriptive approach used in the experiments.

This is a significant line of research to pursue as it joins the requirements and techniques of areas such as semantic web, databases, logic, and uncertain reasoning. The long-term goal is to achieve a descriptive and composable IR technology that provides a framework of modules that information "engineers" can compose into a task-specific solution for an IR task. The ultimate goal is to achieve a framework of logical building blocks that offers classifiers, retrieval models, information extractors, and other functions, and those functional blocks can be composed in a possibly web-based service infrastructure. Thereby, high-level languages can be used that are translated to PDatalog for the purpose of composition and execution.

There are numerous studies about Datalog and PDatalog related to different tasks, some of which are summarized in section 2.2. In addition, some research has been done related to abstraction layers using descriptive approaches for different tasks. For instance, [3] proposed a declarative specification language (Dyna) for modelling NLP algorithms that can be automatically compiled to C++. They concluded that it is extremely helpful for their NLP research, even if it was slower than "hand-crafted" code. Other example is the description of a general framework that synthesizes and extends deductive and semiring parsing, adapting them for translation [11]. This work shows that logics make an attractive shorthand for description, analysis and construction of translation models. It also explains that descriptive approaches could be very beneficial when implementing large-scale translation systems which the authors identify as a major engineering challenge, requiring a huge amount of resources. In addition, the logical description has helped them to understand and compare the common elements for different models/algorithms and their differences.

2.2 Probabilistic Datalog

PDatalog is a probabilistic logical retrieval framework that combines deterministic Datalog (a query language used in deductive databases) and probability theory [7, 18]. It was extended in [19, 22] to improve its expressiveness and scalability for modelling IR models (ranking functions). In addition, it is a flexible platform for modelling and prototyping different IR tasks. Furthermore, probabilistic versions of Datalog are regarded for the semantic web as a platform layer on which other modelling paradigms (ontology-based logic) can rest and rely upon [17, 20].

The potentially high impact of this research lies in the fact that PDatalog is an abstraction layer that gives access to more than just the modelling of learning algorithms. PDatalog has been used as an intermediate processing layer for semantic/terminological logics in different IR tasks such as *ad-hoc* retrieval [13, 14], annotated document retrieval [6] and summarization [4].

Two generations of PDatalog have been developed and a third one is being developed. The main differences between them can be summarized as follows:

In 1st generation PDatalog, probabilistic rules have to be used to model the conditional probabilities. In addition, it has no means to express the probability estimation i.e. the "learning probabilities". Therefore, this process had to be external to PDatalog. In the 2nd generation PDatalog, Bayesian goals and subgoals support, on one hand the modelling of probability estimation, and, on the other hand, the modelling of conditional probabilities. Finally, the 3rd generation will involve high-level predicates related to specific tasks.

1st Generation PDatalog The 1st generation PDatalog for IR was introduced in [8]. The main idea is to allow for probabilities in facts and rules.

Figure 1 describes the syntax utilized in traditional Datalog and in the 1st generation of PDatalog. A PDatalog rule consists of a head and a body. A head is a goal, and a body is a subgoal list. A rule is evaluated such that the head is true if and only if the body is true. So far, the syntax is the one of ordinary Datalog. Specific to the PDatalog utilized here is the specification of materialised views (':=' indicates that the goal is to be materialised) and the probabilistic facts and rules.

2nd Generation PDatalog The 2nd generation of PDatalog provides extended expressiveness using probability estimation and conditional probabilities. It also improved scalability because probabilistic rules are not required and extensional relations and assumptions can be used in order to achieve efficient and scalable programs.

A simplified version (for improving readability) of the syntax specification for the 2nd generation PDatalog is outlined in Figure 2. Everything between a pair of curly brackets, i.e. '{' and '}', is optional. The assumption between predicate name and argument list is the so-called *aggregation* assumption (aggAssump). For example, for disjoint events, the sum of probabilities is the resulting tuple

Traditional Datalog	
fact	::= NAME '(' constants ')'
rule	::= head ':-' body
head	::= goal
body	::= subgoals
goal	::= NAME '(' arguments ')'
subgoal	::= pos_subgoal | neg_subgoal
pos_subgoal	::= atom
neg_subgoal	::= '!' atom
atom	::= NAME '(' arguments ')'
argument	::= constant | variable
constant	::= NAME | STRING | NUMBER
variable	::= VAR_NAME
arguments	::= | argument ',' arguments
constants	::= | constant ',' constants
subgoals	::= | subgoal ',' subgoals
1st Generation Probabilistic Datalog	
prob_fact	::= prob fact
prob_rule	::= prob rule

Fig. 1. 1st Generation PDatalog

probability. In this case, the assumptions 'DISJOINT' and 'SUM' are synonyms, and so are 'INDEPENDENT' and 'PROD'. The assumption in a conditional is the so-called *estimation* assumption (estAssump). For example, for disjoint events, the subgoal "index(Term, Doc) | DISJOINT(Doc)" expresses the conditional probability $P(Term|Doc)$ derived from the statistics in the relation called "index". Complex assumptions such as DF (for document frequency) and MAX_IDF (max inverse document frequency) can be specified to describe in a convenient way probabilistic parameters commonly used in IR.

Expressions with complex assumptions can be decomposed in PDatalog programs with traditional assumptions only. However, for improving the readability and processing (optimization), complex assumptions can be specified. The decomposition of complex assumptions is shown in [19].

2.3 Machine Learning

The Machine Learning community has developed a variety of tools for different applications. Arguably among the most successful approaches to supervised learning are Bayesian methods, ensemble methods such as boosting or algorithms for the selection of experts, Support Vector Machines (SVMs) [9] and kernel methods in general. While these in no way exhaust the field, they have sufficient generality and a range of applicability that goes from multimedia to learning ranking to real–time expert selection.

In particular, incorporating kernel methods and ensemble methods in an IR framework is highly desirable because ensemble methods can act as a "wrapper" for other classifiers seen as black-boxes, while kernel methods provide an easy

goal	::= tradGoal | bayesianGoal | aggGoal
subgoal	::= tradSubgoal | bayesianSubgoal | aggGoal
tradGoal	::= see 1st Generation
tradSubgoal	::= see 1st Generation
bayesGoal	::= tradGoal '|' {estAssump} evidenceKey
bayesSubgoal	::= tradSubgoal '|' {estAssump} evidenceKey
evidenceKey	::= '(' variables ')'
aggGoal	::= NAME {aggAssump} '(' arguments ')'
aggSuboal[1]	::= NAME {aggAssump} '(' arguments ')'
tradAssump	::= 'DISJOINT' | 'INDEPENDENT' | 'SUBSUMED'
irAssump	::= 'SEMI_SUBSUMED' | 'DF' | 'MAX_IDF' | ...
probAssump	::= tradAssump | irAssump
algAssump	::= 'SUM' | 'PROD'
aggAssump	::= probAssump
estAssump	::= probAssump | complexAssump

Fig. 2. 2nd Generation PDatalog: Bayesian Goals

way to integrate multimedia content through mapping to an appropriate vector space. Unfortunately there is no unified theory for these algorithms. However, some mathematical structures underlie one or more approaches and can be used to abstract them, as sketched in figure 3.

To begin with, both Adaboost and SVMs are solutions to optimization problems; thus a sufficiently flexible minimization engine can in principle handle both. However, both methods can potentially be abstracted in much more meaningful and descriptive ways, that we try to outline below.

Adaboost was originally derived in the context of PAC (Probably Approximately Correct) learning; some algorithms of this family have already been integrated in DataLog [15]. Two alternative interpretations of Adaboost look promising from the point of view of creating a high-level operational description of the procedure. As demonstrated in [5], Adaboost can be seen as an approximation to an additive regression model that optimises likelihood on a logistic scale. In this perspective, Adaboost rounds become the iterations of a greedy backfitting algorithm that tries to minimise the difference between the current estimate and the target function according to an exponential loss. Direct minimisation of the binomial log-likelihood leads to variant of the algorithm known as LogitBoost [5], that has a comparable performance. Therefore, one can essentially cast Adaboost as a maximum-likelihood regression procedure. This approach has the advantage that the "weak" classifiers that Adaboost combines (often binary predicated) can be interpreted as (coarse) estimates of the posterior probability of each class given the data, thus linking to a Bayesian framework. In another approach, Adaboost has been derived in terms of relative entropy [12], with a strong relation to classic information theory results.

SVMs, on the other hand, can nowadays be seen to fit in a different trend in Learning Theory that strives to cast learning in geometric terms. This can be traced back at least to the seminal work by Cucker and Smale [2], that systematised learning from examples and bounds on the generalisation error in

the framework of Functional Analysis. Incidentally, this work represents a shift towards considering the topological and geometric aspects of learning. In the same ideal line, we can count subspace methods, such as have been used in information retrieval [16], and manifold learning (see for example [21]).

On the other hand, the SVM optimization problem is more easily understood in geometrical terms (maximizing the margin). SVMs implicitly map the data into a higher dimensional feature space, in which the decision function is a hyperplane. This feature space is indeed a Hilbert space defined in terms of its scalar product (i.e., the kernel). If one considers a logic over its linear subspaces, the decision function can straightforwardly be interpreted as a proposition in the logic. More in general, kernel methods provide a flexible and efficient way of designing a mapping between the raw data and propositions in the logic. Thus research on kernel engineering could be harnessed, for instance, to provide a transparent integration of different data modalities. Tapping into this potential, however, requires a descriptive layer that is sufficiently rich to supports the logical, probabilistic and geometric concepts involved.

3 Modelling Learning in Probabilistic Datalog

The implementation of learning algorithms using a probabilistic logical approach presents the difficulty of modelling a huge number of methods with various theoretical foundations. The different nature of the methods implies that some of the techniques are easier to model than others. For example, Bayesian methods can be directly express in PDatalog, while the adaptation of SVM is not trivial.

For those methods that cannot be represented, two options would be considered. Firstly, research about the creation of a probabilistic approximation of the algorithm is required. Secondly, if that approximation cannot be found, the expressiveness of the language will be revised, studying how it could be modified. Some of the features we would want to include in the 3rd generation of PDatalog are the possibility of using sample spaces and fixed point criteria for allowing the implementation of some learning techniques based on optimization.

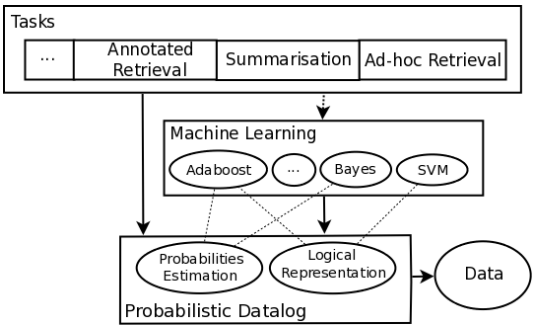


Fig. 3. Overview of a module-based IR System

Figure 3 shows a general diagram that illustrates how machine learning could be combined in a general IR-System implemented in PDatalog. The top level represents how different modules can be used for solving tasks such as annotated retrieval, summarization or ad-hoc retrieval. The intermediate layer models the machine learning algorithms that can be access by other modules if required. This it allows other modules to be tuned used ML techniques. The bottom layer illustrates the characteristics of PDatalog that support the definition of the different modules of the system.

Using this approach, the modules can use any fact derived in the others as long as they are selected for execution. As a result, different configurations can be specified by selecting appropriate sets of modules, without changing any code.

3.1 K Nearest Neighbours Classifier

A k-NN classifier implemented in PDatalog is presented in figure 4. The code shown has been used for text classification on the Reuters-21578 collection (results in section 4.2). A sample of the relations "tf" (modelling the frequency of each term in a document) and "part_of" (assigning training documents with its correct classes) is also illustrated.

The functionality required for the k-NN classifier has been composed from three different modules: Firstly, a document-frequency-based feature selection is applied, obtaining the terms that appear in most documents.

Secondly, a similarity measure is provided by a module that define a modification of the dice similarity following formula 1. This module computes, for each training document and a test document, the product of the weight of matching terms divided by the sum of the importance of all terms in the training document.

$$dice\_sim(testD, trainD) = \frac{\sum w(\text{term}, \text{trainDoc}) w(\text{term}, \text{testDoc})}{\sum w(\text{term}, \text{trainDoc})} \qquad (1)$$

where "w(t,d)" represent the weight of the term "t" in the document "d", tf-idf in our case.

Finally, a module is needed for defining the exact behaviour of the k-NN classifier. It assigns the similarity score respect to each of the "k" most similar documents to the classes labelled for them. After this, these scores are aggregated given a final score for each of the classes previously chosen (Formula 2).

$$class\_score(class, testD) = \sum_{\text{trainDoc} \in \text{class}} sim(\text{testDoc}, \text{trainDoc}) \qquad (2)$$

Although this is the most common technique, there are other strategies for calculating the score of each class. For example, counting the number of neighbours from each class, without taken into account the score of their similarity.

tf		
Value	Term	Document
23	economy	d40
5	expectation	d23
12	provider	d23
7	reuters	d1

part_of		
Value	Document	Class
1	d1	cocoa
1	d5	grain
1	d5	wheat
1	d5	oil

(c) Basic/Extensional Data Representation

```
1  _sort(doc_frequency);
2  # 1. Select the 3000 terms that appear in more documents.
3  df_selected(Term) : doc_frequency(Term):3000;

5  # 2. For each tuple, is_selected(Term) is 1 if the term is one of the selected
       terms
6  is_selected(Term) :- df_selected(Term)|(Term);

8  #3. Compute the term frequency for the selected terms
9  selected_term(Term, Doc) :- selected(Term) & tf(Term, Doc);
10 sum_selected_term(Term, Doc) :- SUM selected_term(Term, Doc);
```

(d) Feature Selection

```
1  # 1. Normalize factor: Sum of feature weights ( tfidf ) for each training document
2  sum_tfidf SUM(TrainDoc) :- tfidf_train(Term, TrainDoc);
3  inv_norm_factor(TrainDoc) :- sum_tf_idf INV (TrainDoc);

5  # 2. Product of tfidf for the matching terms with each training document
6  match(TrainDoc, Term) :- tfidf_test(Term) & tfidf_train(Term, TrainDoc);
7  sum_match SUM(TrainDoc) :- match(TrainDoc, Term);

9  # 3. Similarity score
10 similarity(TrainDoc) :- sum_match(TrainDoc) & inv_norm_factor(TrainDoc);
```

(e) Similarity Measure

```
1  _sort( similarity ); _sort(sum_top_k_classes);
2  # 1. Select the k training documents most-similar to the test document
3  top_similar(TrainDoc) :- similarity(TrainDoc):30;

5  # 2. Associate test document to the classes of the top-retrieved training docs
6  top_k_classes(Class) :- top_retrieved(TrainDoc) & part_of(TrainDoc, Class);

8  # 3. Aggregate scores for each class
9  sum_top_k_classes SUM(Class) :- top_k_classes(Class) | DISJOINT();
```

(f) Classification Strategy

Fig. 4. Modelling of k-NN classifier with document-frequency-based feature selection, dice similarity measure and weighted neighbours

3.2 Implementing polynomial Mercer Kernels in Probabilistic Datalog

Central to SVMs and other Kernel methods such as Kernel PCA is the use of a Mercer Kernel as a non-linear scalar product. Using a Mercer kernel $K(\boldsymbol{u}, \boldsymbol{v})$ implicitly amounts to carrying out a non-linear mapping Φ of vectors $\boldsymbol{u}$, $\boldsymbol{v}$ from the original data space $\mathbb{R}^n$ into a higher–dimensional feature space $\mathbb{F}$, in which the standard scalar product is computed: $K(\boldsymbol{u}, \boldsymbol{v}) = \Phi(\boldsymbol{u}) \cdot \Phi(\boldsymbol{v})$ (see for instance [1]). The advantage is that the mapping Φ never needs to be computed explicitly, which allow handling high-dimensional or infinite-dimensional feature spaces $\mathbb{F}$ efficiently and transparently.

As an example, we provide an implementation in Probabilistic Datalog (Figure 3.2) of two kernels widely used in SVM classifiers, belonging to the polynomial family $K(\boldsymbol{u}, \boldsymbol{v}) = (\boldsymbol{u} \cdot \boldsymbol{v} + c)^d$. Namely, we implement the homogeneus quadratic kernel $K_H = (\boldsymbol{u} \cdot \boldsymbol{v})^2$ and the non-homogeneous quadratic kernel $K_N = (\boldsymbol{u} \cdot \boldsymbol{v} + 1)^2$.

In the case of a two–feature data space $\mathbb{R}^2$ we can make the mapping Φ explicit by expanding the definition of the kernel:

$$K_H = (\boldsymbol{u} \cdot \boldsymbol{v})^2 = (u_1 v_1 + u_2 v_2)^2 \tag{3}$$

$$= (u_1^2 v_1^2 + 2 u_1 u_2 v_1 v_2 + u_2^2 v_2^2) = \Phi_H(\boldsymbol{u}) \cdot \Phi_H(\boldsymbol{v}) \tag{4}$$

from which we conclude that the implicit map is

$$\Phi_H(\boldsymbol{u}) = (u_1^2, \sqrt{2} u_1 u_2, u_2^2) \tag{5}$$

that transforms vectors in the data space $\boldsymbol{R}^2$ into vectors in $\mathbb{F}_H = \mathbb{R}^3$.

Similarly, by expanding the definition of K_N it can be seen that

$$\Phi_N(\boldsymbol{u}) = (u_1^2, \sqrt{2} u_1 u_2, u_2^2, \sqrt{2} u_1, \sqrt{2} u_2, 1) \tag{6}$$

so that $\mathbb{F}_N = \mathbb{R}^6$.

As can be seen, polynomial kernels implicitly compute the correlation and higher order moments of the original features, which arguably explains their effectiveness. In the general case of an n–dimensional data space $\mathbb{R}^n$ we have $F_H = \mathbb{R}^{n(n+1)/2}$ and $F_N = \mathbb{R}^{(n+1)(n+2)/2}$.

4 Feasibility Study

These experiments have been realised as a starting point for proving the feasibility of using declarative approaches for modelling a specific kind of learning methods, namely classifiers. The results illustrate that k-NN can be modelled in PDatalog (the implementation is shown in Figure 4). Furthermore, its performance could be comparable with the quality of algorithmic approaches. However, more learning methods have to be implemented and tested in different environments.

```
1  # 1. Modelling the dot product between U and V and a constant (1)
2  kernel_match(Feature, U, V) :- value(Feature, U) & value(Feature,V);
3  # If the next line is ignored/commented the homogeneus kernel will be computed
4  kernel_match SUM (Feature, U, V) :- constant(Feature, U, V);

6  # 2. Aggregate of product and constant (UV + 1)
7  sum_kernel_match(Feature, U, V) :- kernel_match SUM (Feature, U, V);

9  # 3. Quadratic polynomy (UV + 1)^2 or (UV)^2
10 K_quad_polyn(U, V) :- sum_kernel_match(Feature, U, V)
11                       & sum_kernel_match(Feature, U, V);
```

Fig. 5. Modelling of quadratic Mercer kernel

4.1 Experiment set-up

For this experiment, the "ApteMod" split of Reuters-21578 collection has been used. It is a corpus that contains news-wire histories divided in 7770 documents for training and 3019 for testing. We only consider documents that belong to classes with at least one train and one test documents. Stop-words removal and stemming have been applied.

A feature selection has been carried out based in the document frequency of each term. The terms that appear in more documents have been selected. The document representation has been realized using the tf-idf of each feature previously selected with respect to each document. The algorithm for calculating the similarity between documents is a slightly modification of the dice similarity (Formula 1). Finally, the assignation of each class is realised by choosing the best class after aggregate the weight of each neighbour.

This configuration was chosen due to its simplicity in order to realize a feasibility study. Feature selection based on document frequency is comparable to the performance of more complicated approaches such as Information Gain (IG) or Chi-squared (χ^2) with up to 90% term removal [24]. The number of neighbours considered (k) and the number of features selected have been specified taken into account the ranges recommended in [9, 23].

4.2 Results

The results present the quality measures for text classification using our k-NN classifier on the Reuters-21578 collection. Only one class was assigned for each testing document. We have considered this strategy for simplicity reasons, assuming that this environment is good enough for the feasibility study. However, Yang pointed out that this simplification is not optimal for k-NN.

Our approach achieved 74.88% in terms of micro-averaged F1. Even though this value is not yet comparable with the quality reported in [23](81.4%) and [9](82.3%), it is close enough for the feasibility study at this stage. In addition, the differences in performance are caused, at least in part, by the fact that an inferior feature selection algorithm has been used.

K	Representation	Similarity	FS	MicroF1
30	Norm-tf-idf	dice	DF-1000	0.7328
45	Norm-tf-idf	dice	DF-2000	0.7488

Fig. 6. F1 measures for Reuters-21578

5 Discussion and Further Work

This paper outlines the benefits of incorporating learning algorithms in PDatalog. In addition, its integration in a module-based IR system allows the application of learning techniques combined with different IR models and tasks. The main contributions of this paper are to present and discuss the issues related with modelling learning in PDatalog, showing the practical example of a k-NN implementation, its evaluation for text classification on the Reuters-21578 collection, and a first attempt to model Mercer kernel methods.

We show that the modular modelling of k-NN in PDatalog is not only possible, but also leads to an understandable and compact implementation. In addition, results suggest that descriptive approaches could achieve comparable quality with respect to other paradigms. However, direct comparisons, using exactly the same configuration, and experiments over more collections should be done for having stronger support for this claim.

The expected benefit of applying a descriptive approach is an increase of productivity while using learning methods and the capability of combining different tasks. Probabilistic learning algorithms such as PINs or Bayesian methods and lazy-learners like k-NN can be represented using 2nd generation PDatalog. Other methods like SVM might require more expressiveness of the language or alternative processing techniques. An implementation of two polynomial Mercer kernels is provided, representing the first step forward for achieving this objective.

References

1. R. Courant and D. Hilbert. *Methods of mathematical physics*. Interscience, 1943.
2. F. Cucker and S. Smale. On the mathematical foundations of learning. *Bullettin of the AMS*, 39:1–49, 2002.
3. J. Eisner, E. Goldlust, and N. A. Smith. Compiling comp ling: practical weighted dynamic programming and the dyna language. In *HLT '05*, pages 281–290
4. J. F. Forst, A. Tombros, and T. Roelleke. Polis: A probabilistic logic for document summarisation. In *Studies in Theory of Information Retrieval*, p.201–212, 2007.
5. J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. *Annals of statistics*, 28(2), 2000.
6. I. Frommholz. Annotation-based document retrieval with probabilistic logics. In ECDL 2007, pages 321–332
7. N. Fuhr. Probabilistic datalog - a logic for powerful retrieval methods. In *ACM SIGIR*, pages 282–290, 1995
8. N. Fuhr. Optimum database selection in networked ir. In *NIR'96. SIGIR'96 Workshop on Networked Information Retrieval*, 1996.

9. T. Joachims. Text categorization with support vector machines: Learning with many relevant features. In *ECML-98*, pages 137–142
10. J. W. Lloyd. Practical Advantages of Declarative Programming. In *Joint Conference on Declarative Programming*, 1994.
11. A. Lopez. Translation as weighted deduction. In *EACL 2009*, pages 532–540
12. P. Malacaria and F. Smeraldi. On Adaboost and optimal betting strategies. In *ICDM*, pages 326–332. 2009.
13. C. Meghini, F. Sebastiani, U. Straccia, and C. Thanos. A model of information retrieval based on a terminological logic. In *ACM SIGIR*, pages 298–308, 1993
14. H. Nottelmann. Pire: An extensible ir engine based on probabilistic datalog. In *ECIR*, pages 260–274, 2005.
15. H. Nottelmann and N. Fuhr. Learning probabilistic datalog rules for information classification and transformation. In *CIKM '01*, pages 387–394
16. B. Piwowarski, I. Frommholz, Y. Moshfeghi, M. Lalmas, and K. van Rijsbergen. Filtering documents with subspaces. In *32nd ECIR Conference*, 2010.
17. A. Polleres. From SPARQL to rules (and back). In *16th WWW*, pages 787–796. 2007
18. T. Roelleke and N. Fuhr. Information retrieval with probabilistic datalog. In *Uncertainty and Logics - Advanced models for the representation and retrieval of information*. Kluwer Academic Publishers, 1998.
19. T. Roelleke, H. Wu, J. Wang, and H. Azzam. Modelling retrieval models in a probabilistic relational algebra with a new operator: The relational Bayes. *VLDB Journal*, 17(1):5–37, January 2008.
20. S. Schenk. A SPARQL semantics based on Datalog. In *KI '07*, pages 160–174, 2007.
21. J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.
22. H. Wu, G. Kazai, and T. Roelleke. Modelling anchor text retrieval in book search based on back-of-book index. In *SIGIR '08 Workshop on Focused Retrieval*, pages 51–58, 2008.
23. Y. Yang and X. Liu. A re-examination of text categorization methods. In SIGIR '99, pages 42–49, 1999.
24. Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of ICML-97*, 1997.

Sobre el número de términos de expansión en métodos de realimentación por relevancia para sistemas de recuperación de información estructurada

Luis M. de Campos, Juan M. Fernández-Luna,
Juan F. Huete, Carlos Martín-Dancausa

Departamento de Ciencias de la Computación e Inteligencia Artificial
E.T.S.I. Informática y de Telecomunicación, CITIC-UGR
Universidad de Granada
{lci,jmfluna,jhg,cmdanca}@decsai.ugr.es

Resumen La realimentación por relevancia es una técnica que se puede utilizar para refinar una consulta inicial formulada por un usuario a un sistema de recuperación de información, teniendo en cuenta los resultados obtenidos por el propio sistema y evaluados por el usuario para dicha consulta. La consulta modificada se construye incluyendo nuevos términos extraidos de los documentos considerados como relevantes (posiblemente ponderados de algún modo). En este trabajo pretendemos estudiar experimentalmente la influencia que el número de términos añadidos tiene sobre la calidad de los resultados obtenidos, en el contexto de un sistema de recuperación de información estructurada (XML).

1. Introducción

Con el objetivo de mejorar la calidad de los resultados ofrecidos por un sistema de recuperación de información (SRI) como respuesta a una consulta formulada por un usuario, se puede emplear alguna técnica de modificación automática de consultas, como la denominada *realimentación por relevancia* [23], que trata de integrar información sobre la relevancia o no de los documentos recuperados inicialmente (proporcionada por el usuario) en la consulta original. Esta técnica generalmente utiliza un proceso denominado *expansión de la consulta*, por el que se seleccionan y añaden a la consulta original nuevos términos, junto con el *repesado* (reponderación) de los términos que componen la consulta original, aunque la expansión de consultas suele tener mayor impacto en los resultados.

Son muchas las cuestiones que influyen en el funcionamiento de un método de realimentación por relevancia: qué documentos o partes de documentos (y cuántos) se deben de tener en cuenta (por ejemplo, sólo aquellos juzgados como relevantes, o también los no relevantes), cómo reponderar los términos de la consulta original, qué términos se deben seleccionar como candidatos a ser incluidos en la nueva consulta (por ejemplo aquellos que aparecen sólo en documentos

relevantes), cómo incorporar los términos seleccionados a la consulta (qué tipo de ponderación emplear), y cómo seleccionar tales términos de entre los candidatos. Respecto a esta última cuestión, lo habitual es ordenar por importancia los términos candidatos con algún criterio y seleccionar los k mejores, donde k es un parámetro predeterminado. En este trabajo nos vamos a centrar precisamente en esa cuestión: el número (máximo) de términos de expansión seleccionados, y su influencia en el rendimiento obtenido.

El contexto en el que vamos a desarrollar nuestro estudio no es el de la recuperación de información "clásica", sino la *recuperación estructurada* [17], donde los documentos (habitualmente representados en XML) ya no se consideran unidades indivisibles sino que presentan una estructuración y están formados por una serie de componentes (denominados *unidades estructurales*) interrelacionados entre sí, que necesitan ser indexados y recuperados tanto, como un todo como de forma separada, en relación a las necesidades de un usuario. Esta distinción puede ser importante desde el punto de vista de la realimentación por relevancia, porque al poder recuperarse componentes de documentos en lugar de documentos completos, puede ser más sencillo para un usuario establecer la relevancia o no de tales componentes.

Así pues, el propósito de este trabajo es realizar un estudio experimental sobre la influencia en la calidad de los resultados obtenidos del número de términos de expansión utilizados en la realimentación por relevancia para sistemas de recuperación de información estructurada, e intentar obtener conclusiones útiles sobre esta cuestión.

Existe una gran cantidad de literatura científica sobre métodos de realimentación por relevancia, tanto para documentos planos [1,10,22] como estructurados [3,15,18,20,24,25], así como muchos sistemas de recuperación de información estructurada (siendo los proceedings del Workshop INEX una excelente fuente de información [11,12,13]). Obviamente los resultados obtenidos podrán estar influenciados por la elección tanto del SRI estructurado como del método de realimentación por relevancia. No es pues nuestro objetivo obtener conclusiones aplicables en todas las circunstancias, pero creemos que nuestro estudio, aunque realizado sobre un SRI y un método de realimentación concretos, puede al menos ilustrar ciertas tendencias. Además, no conocemos ningún estudio de estas características realizado sobre un SRI estructurado, donde los mecanismos de evaluación son diferentes a los de la recuperación de información clásica[1].

El trabajo está organizado de la siguiente forma: en la sección 2 se describe brevemente el SRI utilizado en este estudio. La sección 3 describe a su vez el método de realimentación por relevancia que emplearemos. La sección 4 contiene los detalles del estudio experimental, su diseño y resultados obtenidos. Finalmente la sección 5 presenta las conclusiones y algunas propuestas de trabajo futuro.

[1] En [19] se realiza un estudio sobre el número de términos de expansión para pseudo-realimentación por relevancia en SRI clásicos.

2. El sistema de recuperación de información estructurada

El SRI que vamos a emplear en nuestros experimentos es Garnata [8], un sistema específicamente diseñado para trabajar con documentos estructurados en XML y basado en modelos gráficos probabilísticos (redes bayesianas y diagramas de influencia), concretamente en el denominado *Context-based Influence Diagram (CID) model* [4,5].

La forma en que el modelo CID implementado en Garnata calcula la relevancia de cada componente o unidad estructural de un documento es combinando dos tipos de información diferentes. Por un lado la especificidad de la unidad respecto de la consulta: cuantos más términos de la unidad aparezcan en la consulta, más relevante resulta la primera, es decir más claramente la unidad trata solamente (de al menos una parte) del tema de la consulta y no de otros. Por otro lado, la exhaustividad de la unidad respecto de la consulta: cuantos más términos de la consulta emparejan con los de la unidad, más relevante resulta esta última, o sea más claramente la unidad abarca completamente el tema de la consulta.

Estas dos dimensiones sobre la relevancia de una unidad respecto de una consulta se calculan de forma diferente. A grandes rasgos, el funcionamiento básico del sistema es el siguiente: dada una consulta Q formada por un conjunto de términos, dichos términos se instancian como completamente relevantes (es decir $p(t|Q) = 1 \, \forall t \in Q$) en la red bayesiana que representa colección de documentos (véase la figura 1) y, mediante un proceso de propagación de la evidencia, se calcula la probabilidad de relevancia de cada una de las unidades estructurales U de los documentos, $p(U|Q)$, lo que mide la especificidad. La exhaustividad se obtiene definiendo la utilidad de cada unidad en función de la proporción de términos de la consulta que aparecen en dicha unidad. Entonces, transformando la red bayesiana en un diagrama de influencia, se calcula la utilidad esperada de recuperar las diferentes unidades estructurales resolviendo el diagrama de influencia, y se devuelve una lista de unidades en orden decreciente de utilidad esperada.

3. El método de realimentación por relevancia

El método de realimentación por relevancia que emplearemos junto con el SRI Garnata, que realiza expansión de términos pero no repesado, es una versión simplificada del propuesto en [7]. Una vez que el sistema ha devuelto una lista de resultados, como respuesta a una consulta, y el usuario ha juzgado como relevantes o no una serie de unidades estructurales (normalmente las m primeras de la lista), se extraen y analizan los términos que aparecen en esas unidades, con el objetivo de formular una consulta ampliada que incluya algunos de esos términos.

Los términos candidatos a ser incluidos en la consulta expandida son aquellos que no pertenecen a la consulta original, aparecen en unidades consideradas

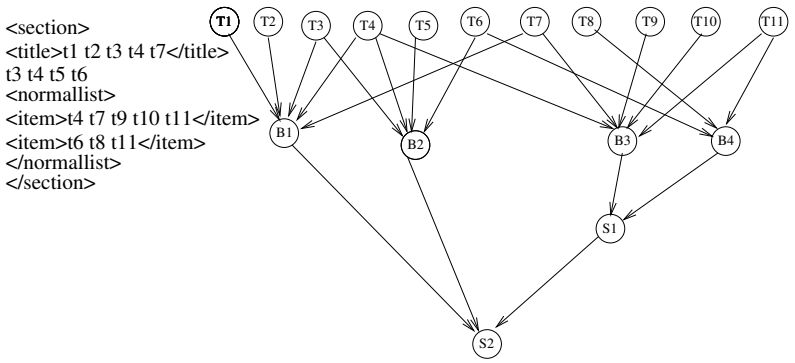


Figura 1. Documento XML y la correspondiente red bayesiana. Ti representan términos índice; la unidad B1 corresponde a la marca `<title>`, y B3 y B4 a la marca `<item>`; las unidades S1 y S2 corresponden a las marcas `<normallist>` y `<section>` respectivamente; B2 es una unidad virtual usada para almacenar el texto que hay dentro de S2 que no está incluido en ninguna otra unidad contenida en S2.

como relevantes y no aparecen en ninguna unidad juzgada como no relevante. La importancia que tiene cada uno de ellos se evalua de forma probabilística, estimando su probabilidad de relevancia $p(t|Q)$ de la siguiente forma:

$$p(t|Q) = \frac{n_{tr}}{n_r} \tag{1}$$

donde n_{tr} es el número de unidades relevantes que contienen el término t y n_r es el número de unidades, de entre las m juzgadas, consideradas relevantes. Los términos candidatos se ordenan de forma decreciente en función de su probabilidad de relevancia estimada mediante la expresión anterior, y se incluyen en la consulta los k primeros.

La nueva consulta ampliada se forma pues con los términos de la consulta original t_Q (manteniendo sus probabilidades sin repesar, $p(t_Q|Q) = 1$), y los k mejores términos candidatos t_C, con las probabilidades estimadas mediante la ecuación (1). El método de inferencia de Garnata ha sido adaptado para manejar consultas con diferentes probabilidades asociadas a los términos que las forman.

4. Estudio experimental

En esta sección describiremos el diseño experimental, la colección documental empleada, las medidas de evaluación del rendimiento y la metodología de evaluación, así como los resultados obtenidos.

4.1. Colección documental y medidas de evaluación

La colección de documentos XML utilizada en los experimentos es la que se ha empleado en las últimas ediciones del workshop INEX [11,12,13], denominada

Wikipedia [9]. Consiste en una versión en XML de la versión inglesa de Wikipedia al comienzo del año 2006 y contiene 659388 artículos (con un tamaño de unos 4600 Megabytes). Las consultas y los correspondientes juicios de relevancia utilizados son los desarrollados para INEX'2006 e INEX'2007 (217 consultas en total, 114 en 2006 y 103 en 2007).

Respecto a la evaluación, hemos considerado la tarea *focused* de INEX, donde el objetivo es recuperar las partes de los documentos más apropiadas pero eliminando los posibles solapamientos entre los resultados obtenidos (por ejemplo no se pueden devolver simultáneamente una sección de un documento y un párrafo de esa sección). La forma en que Garnata realiza esta tarea se describe en [6], y consiste básicamente en filtrar de arriba a abajo la lista original de resultados, eliminando las unidades que se solapan con unidades anteriores.

Las medidas de evaluación del rendimiento consideradas son las empleadas en INEX a partir del año 2007 para la tarea *focused* [16], es decir la precisión interpolada (iP) para ciertos niveles de exhaustividad seleccionados (iP[0.01], iP[0.05] e iP[0.10]), y la precisión interpolada promedio (AiP), todas ellas promediadas para las 217 consultas, y calculadas a partir de la lista ordenada de los 1500 resultados (sin solapamiento) obtenidos por el SRI.

4.2. Metodología de evaluación

El objetivo es comparar los resultados obtenidos por el SRI para las consultas originales y las consultas expandidas construidas tras conocer la relevancia o irrelevancia de las m primeras unidades estructurales de la lista de resultados (obtenida directamente de los juicios de relevancia). Esta comparación debe hacerse con cierto cuidado, puesto que podemos obtener resultados aparentemente muy positivos debido a que los elementos ya juzgados como relevantes muy probablemente serán colocados en las primeras posiciones de la lista de resultados para la consulta expandida (esto es lo que se conoce con el nombre de *ranking effect*). Este uso de los datos empleados para el *entrenamiento* en la propia evaluación tiene un efecto de sobreajuste que artificialmente mejora los resultados. Es por ello que para evaluar métodos de realimentación por relevancia se emplean técnicas como la de la colección residual o el *freezing* [2].

En nuestro caso emplearemos el método de la colección residual, pero adaptado al caso de documentos estructurados (donde hay que tener en cuenta que si se ha juzgado como relevante una unidad, entonces también disponemos de información de relevancia sobre otras unidades que pudieran aparecer en la lista de resultados, como las unidades contenidas en ella o las contenedoras) y a la tarea *focused*. Concretamente, el método es el siguiente:

- Consulta original: de la lista de resultados (con los solapamientos ya eliminados) se eliminan también las m primeras unidades ya juzgadas.
- Consulta expandida: se obtiene la lista de resultados (sin eliminar solapamientos) y se le añaden, en las primeras posiciones, todas las m unidades ya juzgadas; a continuación se utiliza el filtro para eliminar solapamientos y finalmente se eliminan de la lista resultante los elementos juzgados añadidos anteriormente.

4.3. Resultados

Hemos realizado experimentos en los que se ha ido variando tanto el número m de unidades estructurales juzgadas como el número k de términos añadidos a la consulta (con el método descrito en la sección 3). Concretamente hemos experimentado con los valores $m = 5, 10, 20$ y $k = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10$, lo que arroja un total de 30 experimentos (cada uno de ellos con las 217 consultas consideradas). Los resultados obtenidos para las diferentes medidas de rendimiento se presentan en la tabla 1.

	$m = 5$				$m = 10$				$m = 20$			
k	iP[0.01]	iP[0.05]	iP[0.10]	AiP	iP[0.01]	iP[0.05]	iP[0.10]	AiP	iP[0.01]	iP[0.05]	iP[0.10]	AiP
sin RL	0.384	0.309	0.254	0.096	0.334	0.245	0.197	0.069	0.261	0.180	0.122	0.046
1	0.403	0.345	0.298	0.112	0.351	0.279	0.223	0.079	0.323	0.234	0.174	0.062
2	0.379	0.337	0.290	0.112	0.343	0.271	0.228	0.084	0.306	0.235	0.190	0.066
3	0.393	0.340	0.297	0.118	0.370	0.295	0.256	0.092	0.304	0.237	0.195	0.070
4	0.374	0.327	0.288	0.113	0.365	0.307	0.262	0.095	0.312	0.239	0.198	0.073
5	0.382	0.339	0.301	0.119	0.358	0.294	0.256	0.095	0.308	0.246	0.203	0.075
6	0.376	0.330	0.294	0.116	0.351	0.298	0.249	0.096	0.314	0.256	0.209	0.075
7	0.367	0.319	0.285	0.116	0.350	0.287	0.240	0.094	0.307	0.250	0.209	0.074
8	0.353	0.309	0.281	0.114	0.337	0.282	0.250	0.094	0.296	0.241	0.202	0.074
9	0.333	0.298	0.271	0.112	0.333	0.278	0.248	0.095	0.295	0.239	0.204	0.074
10	0.318	0.285	0.267	0.110	0.316	0.268	0.240	0.093	0.293	0.247	0.207	0.076
mejor	0.501	0.444	0.393	0.152	0.489	0.393	0.339	0.124	0.439	0.351	0.278	0.098

Tabla 1. Resultados de los experimentos.

Resulta más ilustrativo observar los datos de la tabla 2, donde en lugar de los valores de las diferentes medidas, se presentan los porcentajes de mejora obtenidos mediante la realimentación por relevancia en relación a los resultados sin realimentación. Podemos observar en primer lugar que en casi todos los casos (excepto para $m = 5$ y valores bajos de exhaustividad) la realimentación mejora los resultados, aunque los porcentajes son bastante variables, desde mejoras insignificantes a otras muy importantes. Parece pues que la realimentación por relevancia (al menos el método propuesto) es en general una técnica útil también en SRI estructurados.

Los mismos resultados de la tabla 2 se presentan de forma gráfica en la figura 2, donde pueden apreciarse de forma más clara ciertas tendencias, de las que podemos obtener algunas conclusiones interesantes:

- Los resultados son progresivamente mejores conforme aumenta el número de unidades estructurales juzgadas como relevantes o no por el usuario (5, 10 o 20). Este resultado es acorde con nuestra intuición, puesto que cuantos más elementos sean juzgados, más información aportan sobre la verdadera naturaleza y el contexto de las unidades que son relevantes para una consulta dada, y en consecuencia mejor trabajo puede realizar el método de realimentación.
- Los resultados también son sistemáticamente mejores conforme aumentamos el nivel de exhaustividad de la medida de precisión interpolada iP (0.01, 0.05

	$m = 5$				$m = 10$				$m = 20$			
k	%iP[0.01]	%iP[0.05]	%iP[0.10]	%AiP	%iP[0.01]	%iP[0.05]	%iP[0.10]	%AiP	%iP[0.01]	%iP[0.05]	%iP[0.10]	%AiP
1	4.92	11.70	16.95	16.13	5.13	13.83	13.12	14.27	23.90	29.86	43.11	35.66
2	-1.23	9.29	14.05	16.41	2.92	10.91	15.47	22.64	17.19	30.54	56.04	44.07
3	2.34	10.22	16.56	22.39	10.86	20.52	29.77	33.16	16.59	31.66	60.43	53.26
4	-2.61	5.95	13.11	18.01	9.54	25.44	32.61	37.48	19.55	33.12	62.94	60.62
5	-0.51	9.70	18.18	23.94	7.18	19.98	29.84	37.51	18.09	37.04	66.63	64.06
6	-2.14	6.83	15.63	21.07	5.22	21.67	26.11	38.97	20.50	42.41	71.70	64.98
7	-4.38	3.24	12.11	20.93	4.74	17.16	21.86	37.13	17.67	38.90	72.05	63.62
8	-7.93	-0.03	10.44	18.51	0.80	15.26	26.88	36.86	13.42	33.98	65.61	62.16
9	-13.37	-3.37	6.62	16.11	-0.27	13.50	25.74	38.16	13.19	32.91	67.74	63.00
10	-17.26	-7.53	4.84	14.35	-5.34	9.43	21.92	35.01	12.40	37.18	69.85	67.59
mejor	30.65	43.77	54.29	58.42	46.68	60.57	71.91	80.42	68.38	94.99	128.09	114.84

Tabla 2. Porcentajes de mejora obtenidos mediante la realimentación por relevancia en relación a los resultados sin realimentación.

y 0.10), así como al considerar valores promedio (AiP). Esto parece indicar que la realimentación por relevancia es una técnica especialmente beneficiosa para aumentar la exhaustividad, aunque también mejora (en menor cuantía) medidas de precisión inicial, que se centran más en los primeros resultados obtenidos.

- Con respecto al número de términos de expansión que produce los mejores resultados, parece depender considerablemente de cuántas unidades hayan sido juzgadas, y en menor cuantía de la medida de evaluación del rendimiento considerada. Cuando se han juzgado pocas unidades ($m = 5$) parece que no es útil expandir con gran número de términos, obteniendo los mejores resultados con uno (o muy pocos) términos de expansión. Para un número intermedio de unidades evaluadas ($m = 10$), el número óptimo de términos de expansión aumenta, oscilando entre 3 y 6. Y de nuevo vuelve a aumentar cuando se evaluan más unidades ($m = 20$), donde ya el número óptimo de términos incluso llega al máximo prefijado de 10. Parece por tanto que hay una correlación positiva entre el número de unidades evaluadas y el número de términos de expansión que conviene utilizar. También parece que existe alguna influencia de la medida de rendimiento considerada, de modo que para medidas asociadas a niveles de exhaustividad bajos (p.e. 0.01) la tendencia es a preferir menos términos de expansión que para medidas con niveles de exhaustividad más elevados (p.e. 0.10) o medidas promedio.

Conviene indicar que todos estos comentarios corresponden a experimentos en los que el número de términos de expansión permanece fijo para todas las consultas. En las tablas y figura anteriores también aparecen los resultados obtenidos cuando para cada consulta se selecciona el número de términos que mejores resultados produce. Esos resultados representan una cota superior sobre las mejoras que podríamos obtener si pudiéramos establecer algún procedimiento

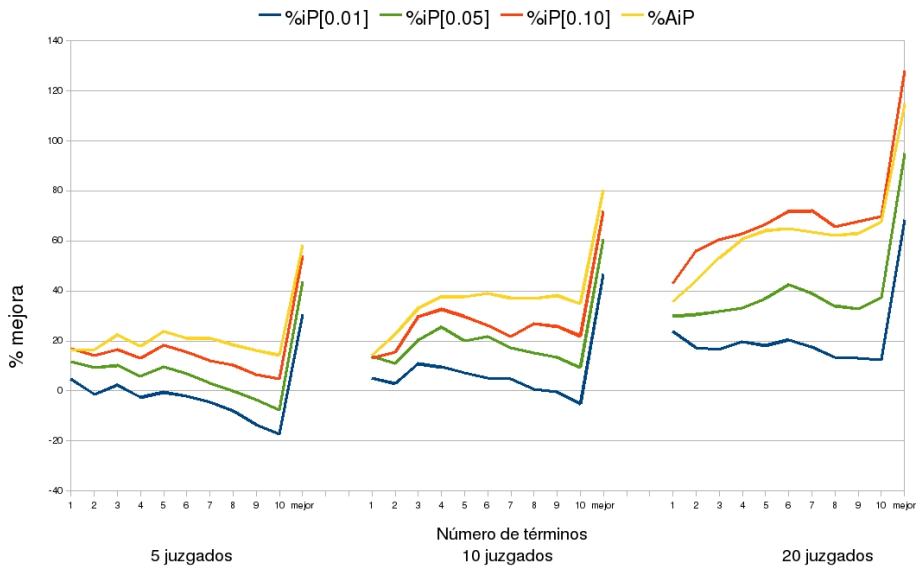


Figura 2. Representación gráfica de los porcentajes de mejora obtenidos mediante la realimentación por relevancia.

de selección del número óptimo de términos de expansión en función de, entre otros datos, las características de las consultas.

Como se puede observar tanto en la tabla 2 como en la figura 2, las mejoras son espectaculares en todos los casos, con porcentajes de mejora respecto de no utilizar realimentación que oscilan entre un mínimo del 30 % hasta un máximo del 128 %. También se mejoran los resultados con respecto a los mejores obtenidos usando un número fijo de términos de expansión, doblando, triplicando e incluso sextuplicando los porcentajes de mejora anteriores. Por ejemplo, en la figura 3 podemos observar, para el caso $m = 10$ y para todas las consultas, las diferencias entre los valores de AiP obtenidos cuando se selecciona el mejor número de términos y cuando no se usa realimentación, así como las diferencias para el caso en que se seleccionan 6 términos de expansión (que es el valor de k que ofrece mejores resultados cuando se fija el número de términos) y cuando no se usa realimentación. En la figura 4 se presentan los resultados análogos, para el caso $m = 5$ e iP[0.01].

Se puede apreciar que se siguen manteniendo las tendencias antes comentadas para el caso de usar un número fijo de términos: cuanto mayor es el número de unidades juzgadas mejores son los resultados, y lo mismo ocurre conforme aumenta el nivel de exhaustividad de las medidas de rendimiento (aunque en términos relativos, los incrementos en rendimiento, con respecto a los obtenidos usando un número fijo de términos, son mayores conforme disminuye el nivel de exhaustividad).

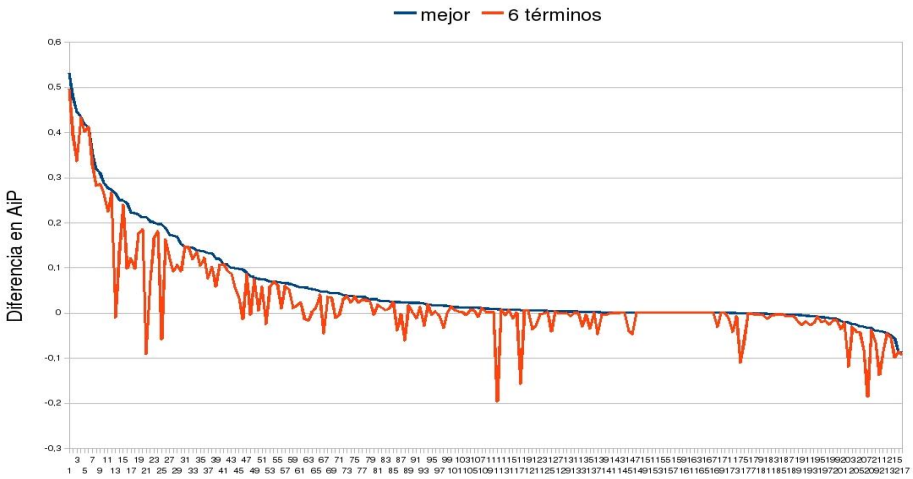


Figura 3. Diferencias en AiP, respecto del caso de no usar realimentación, cuando se usa el mejor número de términos y cuando se usa el mejor número fijo de términos (6), para el caso $m = 10$.

Podemos también observar en la figura 3, pero quizás más claramente en la figura 5, que incluso seleccionando el mejor número de términos de expansión, existen un buen número de consultas para las que se obtienen resultados negativos, es decir, consultas en las que el uso de la realimentación por relevancia resulta contraproducente. Concretamente en el caso representado en la figura 5, del total de 217 consultas hay 45 (21 %) para las que se obtienen peores resultados, y 22 (10 %) en las que la realimentación no tiene efecto alguno. En la tabla 3 se indica el número de consultas que obtienen peores o iguales resultados para el resto de casos.

m	iP[0.01]	iP[0.05]	iP[0.10]	AiP
5	104	94	95	85
10	79	74	81	67
20	69	67	70	65

Tabla 3. Número de consultas en las que la realimentación por relevancia empleando el mejor número de términos de expansión obtiene resultados peores o iguales que sin hacer realimentación.

Por tanto, resulta evidente que aunque en términos globales la realimentación por relevancia es claramente beneficiosa (más aun si se puede determinar el número apropiado de términos de expansión que conviene emplear), hay también muchas situaciones en las que no es útil utilizarla.

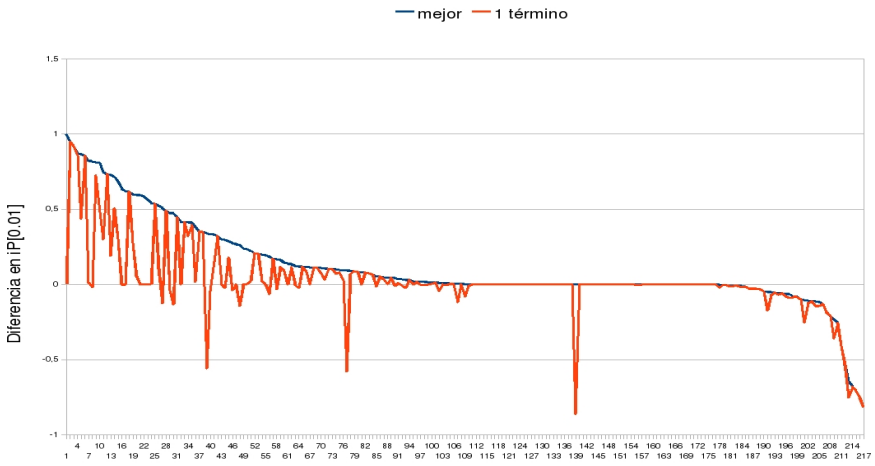


Figura 4. Diferencias en iP[0.01], respecto del caso de no usar realimentación, cuando se usa el mejor número de términos y cuando se usa el mejor número fijo de términos (1), para el caso $m = 5$.

5. Conclusiones y trabajos futuros

En este trabajo hemos realizado un estudio experimental sobre el uso de la realimentación por relevancia en sistemas de recuperación de información estructurada, centrándonos en el número de términos de expansión utilizados y su influencia en la calidad de la recuperación. Los experimentos indican que: (1) la realimentación en general mejora los resultados de todas las medidas de rendimiento utilizadas (empleando el método de la colección residual adaptado al ámbito de la recuperación estructurada); (2) los resultados son mejores para medidas más centradas en la exhaustividad; (3) también mejoran sistemáticamente conforme más información sobre relevancia se utiliza (mayor número de unidades estructurales evaluadas); (4) con respecto al número de términos de expansión que produce los mejores resultados, existe una correlación positiva con el número de unidades evaluadas y negativa con el nivel de exhaustividad deseado; (5) si en lugar de utilizar un número fijo de términos de expansión se emplea un valor variable, es posible obtener resultados mucho mejores; (6) en un porcentaje importante de las consultas la realimentación no genera mejores resultados, independientemente del número de términos empleado.

Como consecuencia de estos resultados, nos planteamos la importancia de estudiar para el futuro mecanismos automáticos para determinar, en función de las características de las consultas, si conviene utilizar realimentación y en caso afirmativo cuál debería de ser el número de términos de expansión a utilizar.

Agradecimientos. Este trabajo ha sido financiado por la Consejería de Innovación, Ciencia y Empresa de la Junta de Andalucía y el Ministerio de Ciencia

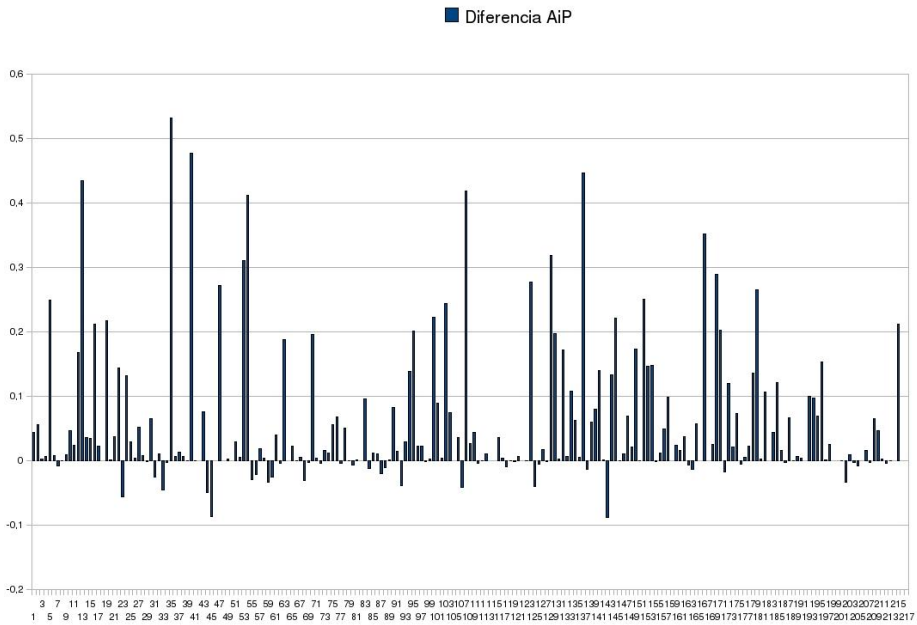


Figura 5. Diferencias entre los valores de AiP obtenidos cuando se selecciona el mejor número de términos y cuando no se usa realimentación, para todas las consultas y $m = 10$.

e Innovación mediante los proyectos P09-TIC-4526 y TIN2008-06566-C04-01, respectivamente.

Referencias

1. C. Buckley, G. Salton & J. Allan. The effect of adding relevance information in a relevance feedback environment. Proceedings of the seventeenth annual international ACM SIGIR conference, 292–300, 1994.
2. Y.K. Chang, C. Cirillo & J. Razon. Evaluation of feedback retrieval using modified freezing, residual collection & test and control groups. G. Salton (ed.), The SMART Retrieval System: Experiments in Automatic Document Processing, chapter 17, 355–370. Prentice Hall, Englewood Cliffs, 1971.
3. C.J. Crouch, A. Mahajan & A. Bellamkonda. Flexible XML retrieval based on the extended vector model. Advances in XML Information Retrieval, Lecture Notes in Computer Science 3493:149–153, 2004.
4. L.M. de Campos, J.M. Fernández-Luna & J.F. Huete. Using context information in structured document retrieval: An approach using influence diagrams. Information Processing and Management 40(5):829–847, 2004.
5. L.M. de Campos, J.M. Fernández-Luna & J.F. Huete. Improving the context-based influence diagram for structured retrieval. Lecture Notes in Computer Science 3408:215–229, 2005.
6. L.M. de Campos, J.M. Fernández-Luna, J.F. Huete, Carlos Martín-Dancausa & A.E. Romero. The Garnata Information Retrieval System at INEX'07. Focused Access to XML Documents, Lecture Notes in Computer Science 4862:57–69, 2008.

7. L.M. de Campos, J.M. Fernández-Luna, J.F. Huete & C. Martín-Dancausa. Content-oriented relevance feedback in XML-IR using the Garnata information retrieval system. Flexible Query Answering Systems, Lecture Notes in Artificial Intelligence 5822:617–628, 2009.
8. L.M. de Campos, J.M. Fernández-Luna, J.F. Huete & A.E. Romero. Garnata: An information retrieval system for structured documents based on probabilistic graphical models. Proceedings of the IMPU'06 conference, 1024–1031, 2006.
9. L. Denoyer & P. Gallinari. The Wikipedia XML corpus. SIGIR Forum, 40:64–69, 2006.
10. E.N. Efthimiadis. Query expansion. Annual review of information science and technology (ARIST) 31:121–187, 1996.
11. N. Fuhr, M. Lalmas & A. Trotman (Eds.). Comparative Evaluation of XML Information Retrieval Systems, 5th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2006. Lecture Notes in Computer Science 4518, 2007.
12. N. Fuhr, M. Lalmas & A. Trotman (Eds.). Focused Access to XML Documents, 6th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2007. Lecture Notes in Computer Science 4862, 2008.
13. S. Geva, J. Kamps & A. Trotman (Eds.). Advances in Focused Retrieval, 7th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2008, Lecture Notes in Computer Science 5631, 2009.
14. D. Harman. Relevance feedback revisited. Proceedings of SIGIR, 1–10. ACM Press, 1992.
15. L. Hlaoua, M. Boughanem & K. Pinel-Sauvagnat. Combination of evidences in relevance feedback for XML retrieval. Proceedings of the sixteenth ACM conference on Conference on information and knowledge management 893–896, 2007.
16. J .Kamps, J. Pehcevski, G. Kazai, M. Lalmas & S. Robertson. INEX 2007 evaluation measures. Focused Access to XML Documents, Lecture Notes in Computer Science 4862:24–33, 2008.
17. M. Lalmas. XML Retrieval. Morgan & Claypool Publishers, 2009.
18. Y. Mass & M. Mandelbrod. Relevance feedback for XML retrieval. In Advances in XML Information Retrieval, Lecture Notes in computer Science 3493:154–157, 2004.
19. P. Ogilvie, E. Voorhees, J. Callan. On the number of terms used in automatic query expansion. Information Retrieval 12:666–679, 2009.
20. H. Pan, A. Theobald & R. Schenkel. Query refinement by relevance feedback in an XML retrieval system. 23rd International Conference on Conceptual Modeling, Lecture Notes in computer Science 3288:854–855, 2004.
21. J. Rocchio Jr. Relevance feedback in information retrieval. G. Salton (ed.), The SMART Retrieval System: Experiments in Automatic Document Processing, chapter 14, 313–323. Prentice Hall, Englewood Cliffs, 1971.
22. I. Ruthven & M. Lalmas. A survey on the use of relevance feedback for information access systems. Knowledge Engineering Review 18(2):95–145, 2003.
23. G. Salton & C. Buckley. Improving retrieval performance by relevance feedback. Journal of the American Society for Information Science 41(4):288–297, 1990.
24. R. Schenkel & M. Theobald. Feedback-driven structural query expansion for ranked retrieval of XML data. Advances in Database Technology, Lecture Notes in Computer Science 3896:331–348, 2006.
25. Using relevance feedback in XML retrieval. Intelligent Search on XML Data, Lecture Notes in Computer Science 2818:133–143, 2003.

Estrategias de Optimización de Consultas XPath Flexibles sobre XML Wavelet Trees*

Nieves R. Brisaboa[1], Ana Cerdeira-Pena[1], Gonzalo Navarro[2], Gabriella Pasi[3]

[1] Database Lab., Univ. da Coruña, Spain. {brisaboa,acerdeira}@udc.es
[2] Dept. of Computer Science, Univ. of Chile, Chile. gnavarro@dcc.uchile.cl
[3] DISCO, Univ. degli Studi di Milano Bicocca, Italy. pasi@disco.unimib.it

Resumen El almacenamiento autoindexado y comprimido de documentos es una área de investigación muy activa y prometedora debido a que las estructuras de datos que usan permiten no sólo ahorrar espacio de almacenamiento en bases de datos documentales y bibliotecas digitales, sino que además proporcionan ahorros significativos de tiempo de procesamiento. En [2] se presentó una nueva estructura de datos denominada XML Wavelet Tree (XWT), un autoíndice comprimido para documentos XML creado mediante la adaptación de un autoíndice para textos planos [3] basado en el uso de wavelet trees. Además, se mostraba cómo el XWT podía ser usado para responder eficientemente consultas típicas XPath sobre estructura y contenido de los documentos XML. En este trabajo mostramos cómo el XWT puede usarse también para resolver consultas *flexibles* [5,8,9]. Estas consultas constituyen una extensión de XPath que trata de introducir cierta flexibilidad en las búsquedas, adecuándolas más al ámbito de la recuperacion de información; ya que permiten no sólo combinar restricciones sobre estructura y contenido, sino también puntuar y hacer *ránking* de las respuestas, es decir, de los distintos documentos y/o fragmentos de documentos recuperados. Por último, presentamos diversas estrategias heurísticas de optimización del plan de resolución de las consultas y las evaluamos experimentalmente.

1. Introducción

En los últimos años, las bases de datos documentales han cobrado una gran importancia debido al gran crecimiento experimentado por biblitecas digitales y la propia expansión de la Web. En este contexto, el lenguaje de marcado XML [1] se ha convertido en el estándar *de facto* para la representación de información semi-estructurada, y se han definido lenguajes de consulta como XPath y XQuery [1] que permiten la realización de consultas tanto por estructura como por contenido. Sin embargo, estos lenguajes asumen que el usuario tiene pleno conocimiento de la estructura de los documentos, de manera que solamente permiten formular consultas *exactas*: bien se obtiene un resultado que cumple exactamente las restricciones expresadas en la consulta, bien el resultado de la consulta es vacío. Esta suposición resulta poco realista. Es por ello que, recientemente, tanto desde el ámbito de las BD como desde el área de la RI se

* Financiado parcialmente por el MEC ref. TIN2009-14560-C03-02; por el MICINN ref. AP2007-02484 (FPU), para Ana-Cerdeira Pena; y por Fondecyt ref. 1-080019 (Chile), para el tercer autor.

han venido desarrollando distintos trabajos que tratan de introducir cierto grado de *flexibilidad* en las búsquedas que, por contenido y estructura, se realizan sobre la información almacenada en documentos XML [6,11,14].

Uno de estos trabajos es el que se recoge en [5,8,9], donde los autores presentan una extensión al lenguaje de consulta XPath. A través de este lenguaje, el usuario puede utilizar palabras clave para realizar búsquedas por contenido (como en RI), y restricciones *flexibles* sobre la estructura del documento, permitiendo así la recuperación de documentos y/o fragmentos de documentos XML con una estructura *aproximada* a la definida en la consulta. A diferencia del lenguaje XPath, donde la evaluación de una consulta produce un conjunto de resultados, la evaluación de una consulta *flexible* devuelve un *ránking* de fragmentos/documentos XML, al igual que ocurre en el ámbito de la RI.

La definición de estos lenguajes ha dado lugar al estudio de nuevas estructuras de datos para la representación e indexación de documentos XML que permitan una evaluación eficiente de las consultas, también *flexibles*. En este sentido, el uso de representaciones comprimidas y autoindexadas, permite no sólo ahorrar espacio de almacenamiento, sino que además proporciona ahorros significativos de tiempo de procesamiento. En [2] se presentó una nueva estructura de datos denominada XML Wavelet Tree (XWT), un autoíndice comprimido para documentos XML creado mediante la adaptación de un autoíndice para textos planos [3] basado en el uso de wavelet trees. Además, se mostraba cómo el XWT podía ser usado para responder eficientemente consultas típicas XPath sobre estructura y contenido de los documentos XML. En este trabajo, basándonos en las consultas *flexibles* definidas en [5,8,9], mostramos y analizamos con experimentos cómo el XWT puede usarse también para resolver de manera eficiente estas nuevas extensiones de XPath empleando distintas estrategias heurísticas de evaluación de las consultas, en función de la frecuencia e incluso número de descendientes de los elementos XML implicados en la consulta.

2. Trabajo relacionado

En [3] se presentó una nueva forma de organizar los bytes que forman los códigos de un texto comprimido utilizando una técnica de compresión estadística semi-estática, basada en palabras y orientada a bytes [7,4,12]. Básicamente la idea consiste en ir sustituyendo cada palabra del texto por un código formado por bytes, pero en vez de almacenarlos de manera consecutiva se reorganizan en diferentes nodos de un árbol, denominado Wavelet Tree (WT). El XML Wavelet Tree (XWT) [2], por su parte, sigue la misma estrategia, pero adaptada para trabajar con documentos XML. Así, se distinguen dos vocabularios diferentes, uno para los *tags* o etiquetas, y otro para el resto de palabras del documento XML (*no-tags*), siendo el texto comprimido mediante la técnica de compresión *(s,c)-DC* [4]. Este compresor permite distinguir entre bytes que sólo pueden aparecer como último byte de un código marcando el final de éste (*stoppers*), y bytes que nunca finalizan un código (*continuers*). El número de *stoppers* y *continuers* de cada vocabulario depende de sus respectivos tamaños y distribución de frecuencias de sus palabras, y se eligen de manera que minimicen la ratio de compresión [4]. El hecho de usar este compresor permite, además, reservar cier-

tos bytes para usar como primeros bytes de los códigos asignados a *tags*; de tal forma que simplemente viendo el primer byte de un código ya podamos saber si está codificando una etiqueta o una palabra que no es una etiqueta. Asimismo, esta propiedad permite, implícitamente, mantener localizados todos los *tags* en determinados nodos del árbol XWT (ver la rama *B4* de la Fig. 1), y por ende, la estructura de nuestro documento XML, proporcionando así una forma eficiente de resolver consultas estructurales (para más detalles, consultar [2]).

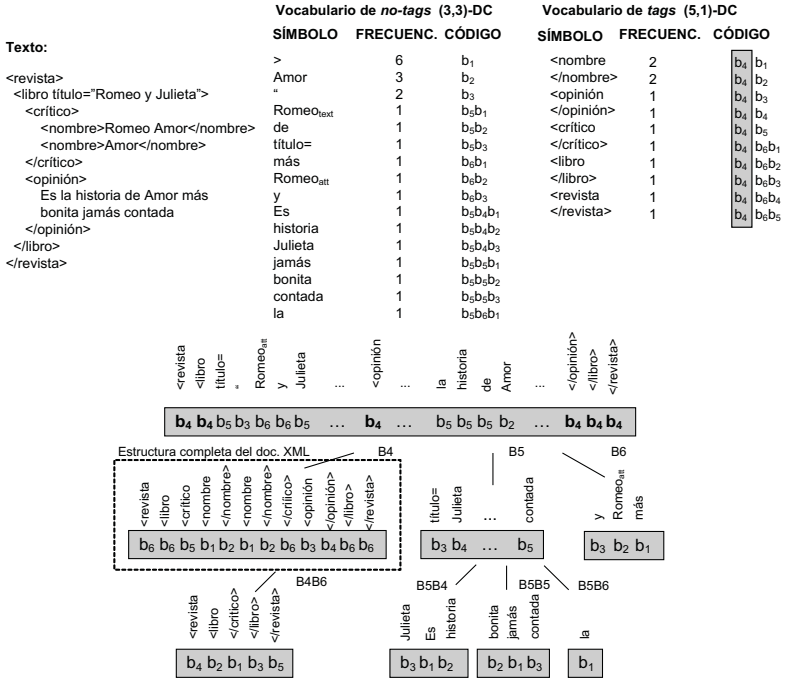


Fig. 1. Ejemplo de XWT.

La construcción del XWT se realiza en dos fases. En la primera de ellas, se lleva a cabo el *parsing* del documento XML de entrada y se crean los dos vocabularios previamente descritos. A continuación, las palabras de cada uno de los vocabularios son codificadas siguiendo un esquema de codificación *(s,c)-DC*. Tal y como se puede apreciar en la parte superior de la Fig. 1, b_1, b_2 y b_3 son los bytes usados como *stoppers* para los códigos de palabras que no son etiquetas, mientras que para las etiquetas, los *stoppers* son el conjunto de bytes formado por b_1, b_2, b_3, b_4 y b_5. Es importante fijarse también cómo el byte b_4 aparece como primer byte de todos los códigos correspondientes a etiquetas, pero no en ninguno de los códigos del vocabulario de palabras que no lo son (ver bytes sombreados en la columna `CÓDIGO` del vocabulario de *tags*). La segunda fase consiste en realizar una segunda pasada sobre el texto, reemplazando cada palabra por su código correspondiente, y organizar sus bytes en los distintos nodos del XWT. La raíz del árbol contiene el primer byte de cada código, siguiendo el mismo orden de las palabras que dichos códigos codifican en el texto original. Cada

nodo X del segundo nivel, contiene los segundos bytes de aquéllos códigos cuyo primer byte es x, siguiendo nuevamente el orden del texto. Es decir, el segundo byte correspondiente a la i-ésima ocurrencia del byte x en la raíz, estará situado en la posición i en el nodo X. Esta misma idea se utiliza para crear los niveles inferiores del árbol, que tendrá tantos como número de bytes tenga el código más largo. En la Fig. 1, se muestra un ejemplo de construcción del XWT. En la parte inferior de la figura se puede observar que el byte b_5 es el quinto byte de la raíz porque es el primer byte del código asignado a *Julieta*, la quinta palabra del texto. Por su parte, su segundo byte, b_4, está situado en la segunda posición del nodo $B5$, porque *Julieta* es la segunda palabra en la raíz que empieza por b_5. Del mismo modo, su tercer byte, b_3, está en el tercer nivel en la primera posición del nodo $B5B4$, porque su segundo byte es el primer b_4 en el nodo $B5$.

En general, el XWT permite representar un documento ocupando aproximadamente el 35 % de su tamaño original, y construirlo consume prácticamente el mismo tiempo que comprimir el documento; es decir, la reorganización de los códigos se realiza sin apenas coste a mayores sobre el tiempo de compresión.

2.1. Operaciones sobre el XWT

Dado que el XWT se basa en la misma estrategia de creación de un WT, las operaciones de búsqueda y recuperación de texto se resuelven usando dos operaciones básicas, *rank* y *select*. Sea $B{=}b_1,\ b_2,\ldots,b_n$ una secuencia de bytes:

- $rank_b(B,p) = i$ si el byte b aparece i veces en B hasta la posición p.
- $select_b(B,j) = p$ si la j-ésima ocurrencia de b en B está en la posición p.

A través de operaciones *rank* es posible recuperar la palabra de una posición cualquiera del texto, mientras que usando operaciones *select* se puede localizar eficientemente una determinada ocurrencia de una palabra. Por ejemplo, supongamos que queremos localizar la primera ocurrencia de la palabra *Julieta* en el ejemplo de la Fig. 1. La búsqueda comenzará en el nodo hoja $B5B4$, ya que el código de *Julieta* es $b_5b_4b_3$. En este nodo buscamos la posición de la primera ocurrencia del byte b_3 (el último byte de *Julieta*), calculando $select_{b_3}(B5B4, 1) = 1$. De esta forma obtenemos que esa posición es la 1. Todas las palabras cuyo último byte se almacena en el nodo $B5B4$ están representadas en el nodo $B5$ con el byte b_4, y siguen el mismo orden que en el texto original. Así, el valor 1 que hemos obtenido con la operación *select* anterior nos indica que el primer byte b_4 del nodo $B5$ es el correspondiente a la primera ocurrencia de *Julieta* en el documento. Nuevamente, calculando $select_{b_4}(B5, 1) = 2$, obtenemos que nuestro código es el segundo empezando por b_5 en el nodo raíz. Finalmente, calculamos $select_{b_5}(raiz, 2) = 5$, que nos dice que la primera ocurrencia de *Julieta* aparece en la posición 5 del texto. De forma análoga se puede recuperar una palabra del texto, empezando por la raíz del XWT y realizando operaciones *rank* sucesivas hasta alcanzar el nodo hoja que corresponda.

Puede observarse que el rendimiento del XWT depende directamente de la implementación de las operaciones *rank* y *select*[1]. Dado que su descripción queda fuera del ámbito de este trabajo, no se explicarán aquí; pero es posible encontrar

[1] Se basa en una estructura de contadores parciales que evita tener que contabilizar el número de ocurrencias de un byte desde el principio de un nodo del XWT. Existe un *tradeoff* espacio-tiempo.

más detalles en [3]. Allí también se muestra cómo los WT compiten con los II ganándoles en todo tipo de operaciones, cuando se dispone de poco espacio.

En el contexto de los lenguajes de consulta XML, también se ha demostrado el buen comportamiento del XWT, a la hora de resolver consultas típicas XPath sobre estructura y contenido de los documentos XML. En [2] se muestran varios ejemplos de procedimientos que permiten la búsqueda eficiente de atributos con un determinado valor, o la búsqueda de *elementos* XML que contengan una determinada palabra, para resolver consultas XPath del estilo *//resumen* [*contains*(., *Madrid*)]. Puesto que el XWT es una representación exacta de un documento XML, cualquier operación que se haga sobre el documento original, puede también realizarse sobre esta representación. Por lo tanto, todas las consultas XPath y cualquier extensión a ellas, se pueden resolver usando el XWT, pero de manera mucho más eficiente, haciendo uso de las propiedades de autoindexación que ofrece. En este artículo nos centramos en mostrar cómo el XWT puede resolver también eficientemente consultas con restricciones *flexibles* como las propuestas en [5,8], centrándonos en dos de las restricciones estructurales más importantes allí definidas, *below* y *near*.

3. Consultas XPath flexibles

El lenguaje de consulta XPath asume un modelo *booleano* de selección de elementos XML; es decir, particiona el conjunto de elementos de un documento XML en aquéllos que cumplen exactamente la condición que expresa una consulta, y aquéllos que no la cumplen. Sin embargo, en algunos escenarios asumir este modelo no resulta apropiado. Aún cuando un documento XML tiene asociado su correspondiente esquema, éste puede no estar disponible al usuario. Incluso distintos documentos XML compartiendo el mismo esquema pueden ser muy diferentes. Además, un mismo árbol XML puede ser descrito en ocasiones utilizando diferentes esquemas. En consecuencia, los usuarios deben realizar lo que se conoce como consultas *ciegas* [10], consultas sin un conocimiento preciso del esquema o estructura del documento XML con el que se está trabajando. En estos casos, la posibilidad de utilizar un lenguaje de consultas *flexible* que permita la realización de consultas *aproximadas* resulta de gran ayuda. Con este objetivo, en [5,8,9] los autores presentan un nuevo lenguaje que permite al usuario especificar restricciones *flexibles* tanto sobre la estructura como sobre el contenido de documentos XML. Dichas restricciones se definen como extensiones de XPath, de tal forma que las consultas del lenguaje propuesto se pueden expresar utilizando la sintaxis estándar XPath y las nuevas restricciones. Estas restricciones *flexibles* expresan una condición *aproximada* sobre la estructura y/o contenido de los documentos y su evaluación produce una puntuación o valor que expresa el grado de compatibilidad de la estructura y/o contenido de los fragmentos recuperados con respecto a la condición *exacta*. A continuación se describen las dos restricciones estructurales flexibles en las que nos centramos en este trabajo.

3.1. Añadiendo flexibilidad a la estructura: below y near

La restricción *below*, insertada como eje dentro de un *path* de búsqueda, selecciona todos aquellos elementos XML, atributos o texto de elementos que

son descendientes del elemento sobre el que se expresa la condición. Por ejemplo, la consulta *//noticia BELOW //imagen*, recuperaría todos los fragmentos correspondientes a elementos *imagen* que tuviesen como ancestro un elemento etiquetado como *noticia*. En la Fig. 2 a), 2 b) y 2 c) se pueden ver ejemplos de posibles fragmentos recuperados por la consulta anterior. Por su parte, *near* es una generalización de *below*, ya que extiende la selección a elementos que son descendientes, ancestros o que "rodean" al elemento de la condición. La siguiente consulta, *//noticia NEAR //imagen*, recuperaría todos los fragmentos correspondientes a elementos *imagen* cercanos a algún elemento etiquetado como *noticia*, pero no necesariamente en su mismo *path* desde la raíz (ver Fig. 2 d)).

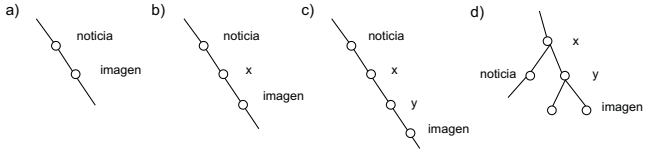


Fig. 2. Ejemplos de fragmentos recuperados por *below* (a), b), c)) y por *near* (d)).

A la hora de evaluar ambas restricciones se utiliza una función de evaluación que define el grado de *matching* entre el *path* especificado en la consulta flexible y el *path* del fragmento/s recuperado/s. En el caso de *below*, si tenemos una consulta del estilo *//A BELOW //B*, el *path* que mejor se ajustaría a la consulta (i.e. el fragmento *ideal*) sería aquél donde B es hijo directo de A; es decir, A/B. Cuanto mayor es la distancia entre A y B, menor debe ser la relevancia del *path*; esto es, la función de evaluación debe ser inversamente proporcional a la distancia entre A y B. Definimos así dicha función como $Match(c_p,\ f_p) = 1/d(A,B)$, donde c_p es el *path* de consulta, f_p es el *path* del fragmento recuperado, y $d(A,B)$ es la distancia entre los elementos A y B en f_p, dada por el número de aristas de separación entre ellos. El valor $Match(c_p,\ f_p)$ es lo que se denomina *Retrieval Status Value* (RSV) del fragmento recuperado con respecto a la consulta. En el siguiente ejemplo, se pueden ver los distintos fragmentos y su respectivo RSV en respuesta a la consulta *//libro BELOW //autor*, considerando 3 la longitud máxima de un *path* en el documento de búsqueda y donde $*$ representa un nodo cualquiera en el *path* que va de *libro* a *autor*: *i*) *libro/autor*, RSV=1; *ii*) *libro/*$*$*/autor*, RSV=1/2=0.5; *iii*) *libro/*$*$*/*$*$*/autor*, RSV=1/3=0.33.

En el caso de *near* los *paths* que mejor se ajustan a esta restricción son A/B y B/A. A mayor distancia entre A y B, menos relevante será un fragmento, con lo que es posible seguir definiendo la misma función de evaluación que para *below*, es decir, $Match(c_p,\ f_p) = 1/d(A,B)$. En teoría, se pueden identificar infinitos *paths* cuando se utilizan las restricciones *below* y *near*; en la práctica, se considera un límite máximo definido por el usuario. Por ejemplo, *//libro* $NEAR_3$ *//autor*, devolvería todos aquellos fragmentos correspondientes a elementos *autor*, que estuviesen "cerca" de algún *libro*, a una distancia no superior a 3.

4. Consultas XPath flexibles sobre XWT

Tal y como se explicó en la Sección 2, la estructura de un documento XML se representa de modo compacto y localizado en nodos específicos del XWT, ya

que los *tags* se representan en ellos siguiendo el mismo orden que en el documento. Por lo tanto, es posible resolver consultas *flexibles* sobre la estructura de un documento haciendo uso únicamente de esos nodos y omitiendo el resto del texto comprimido. Con ello se consigue un gran ahorro del tiempo de procesamiento. Asimismo, el XWT hace uso también de una estructura de bits [13], que permite la navegación entre los distintos elementos del documento XML, proporcionando de manera eficiente el padre o incluso el i-ésimo hijo de un elemento dado.

Below: a la hora de resolver consultas con la restricción *below*, como por ejemplo, *//revista* $BELOW_2$ *//volumen*, que trataría de recuperar todos los elementos *volumen* descendientes de algún elemento *revista*, a distancia no superior a 2, podemos proceder de dos formas diferentes: `1)` localizar los elementos *volumen* y buscar una ocurrencia de *revista* entre sus ancestros, a distancia menor o igual que 2; `2)` recorrer los elementos *revista* y chequear sus descendientes en búsqueda de elementos *volumen* hasta alcanzar la máxima distancia permitida. Para saber en qué dirección se debe operar y optimizar los tiempos de procesamiento, hemos desarrollado distintas estrategias heurísticas. Sea *//A* $BELOW_d$ *//B* nuestra consulta; f_A y f_B, las frecuencias respectivas de A y B; y h_{A_1} y h_{A_2}, el número medio de hijos a distancia 1 y 2 de A, respectivamente; se proponen las siguientes estrategias de evaluación que son analizadas en la Sección 5: *a*) Si $f_A < f_B \rightarrow$ `2)`; si no $\rightarrow$ `1)`; *b*) Utilizar siempre `1)`; *c*) Si $f_A * (h_{A_1} + h_{A_2}) < f_B * 2 \rightarrow$ `2)`; si no $\rightarrow$ `1)`.

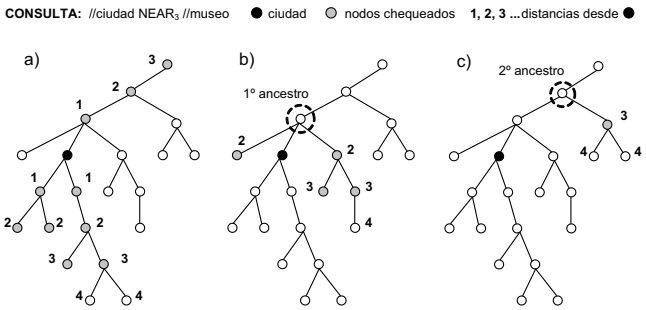


Fig. 3. Restricción *near*: a) operaciones *below*, b) descendientes del 1^{er} ancestro, c) descendientes del 2^o ancestro

Near: en el caso de la restricción *near* se sigue un procedimiento diferente. Supongamos la siguiente consulta: *//ciudad* $NEAR_3$ *//museo*, con la que pretendemos recuperar todos los elementos *museo* que tengan como ancestro, descendiente o a su alrededor, y siempre a distancia no mayor que 3, un elemento *ciudad*. Dado que se trata de una generalización de *below*, primero se realizan dos operaciones *below*, *//ciudad* $BELOW_3$ *//museo* y *//museo* $BELOW_3$ *//ciudad*. Con ellas cubrimos todos los casos de relaciones ancestro-descendiente (o descendiente-ancestro) que se pudiesen dar entre *ciudad* y *museo* en el documento XML (ver Fig. 3 a)). A continuación, se procede tomando las ocurrencias del elemento menos frecuente y chequeando los descendientes de ancestros consecutivos, hasta alcanzar la máxima distancia permitida. Es decir, siguiendo el ejemplo expuesto, supongamos que *ciudad* es el elemento menos frecuente. Una vez

localizada una ocurrencia de *ciudad*, vamos a su primer ancestro y chequeamos sus descendientes, hasta de segundo nivel, en búsqueda de un elemento *museo* (puesto que la máxima distancia permitida es 3 y el primer ancestro se encuentra a distancia 1. Ver Fig. 3 b)). Después se repite el mismo proceso, pero con el segundo ancestro, y chequeando sus descendientes de primer nivel (ya que el segundo ancestro está a distancia 2 y la máxima permitida es 3. Ver Fig. 3 c)). En general, si la distancia máxima es d, el proceso continúa hasta alcanzar el i-ésimo ancestro, donde $i = d - 1$, reduciendo el nivel de los descendientes a comprobar en cada paso del algoritmo.

Consultas por contenido y estructura: para resolver consultas con restricciones sobre la estructura y el contenido de un documento XML usando el XWT, se sigue la siguiente estrategia. Supongamos que tenemos la siguiente consulta (ver Fig. 4): *//libro* [$contains(., viaje)$] $BELOW_2$ *//imagen*. Se empieza localizando la primera ocurrencia de *viaje* en la raíz del XWT, mediante operaciones *select* consecutivas desde el correspondiente nodo hoja. Luego, se contabiliza el número de ocurrencias de las etiquetas de apertura y cierre de *libro* que hay hasta esa posición. Con ello podemos saber cuántas ocurrencias de *libro* contienen a esa ocurrencia de la palabra. Sin embargo, sólo nos interesan aquellos libros que además satisfagan la restricción estructural *libro* $BELOW_2$ *imagen*, la cual se chequeará para cada una de las ocurrencias halladas. En el ejemplo, se ve cómo la primera ocurrencia de *viaje* está contenida dentro de la primera ocurrencia de *libro*; pero al no cumplir ésta la restricción *below*, no se añade al resultado.

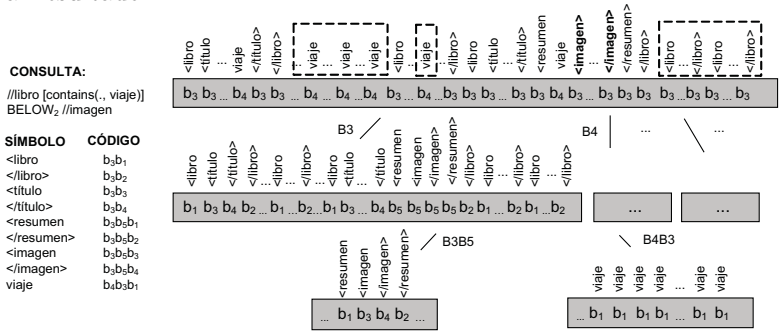


Fig. 4. Ejemplo de consulta sobre contenido con una restricción estructural flexible.

A continuación, en vez de repetir el mismo proceso con la siguiente ocurrencia de la palabra (en este caso, la 2ª ocurrencia de *viaje*), podemos saltar parte del texto si localizamos la siguiente ocurrencia de *libro* que satisfaga la restricción estructural, y que sea posterior a la ocurrencia de *viaje* que habíamos localizado en el paso previo. Siguiendo el ejemplo de la Fig. 4, la ocurrencia de *libro* que nos interesa es la tercera, no la segunda, ya que ésta no cumple la restricción impuesta por *below*. Una vez localizada, basta con mirar si contiene al menos una ocurrencia de *viaje*. Dado que contiene la 6ª ocurrencia de *viaje*, los elementos *imagen* relacionados con la tercera ocurrencia de *libro* (ver ocurrencias destacadas en negrita en la Fig. 4) formarán parte del resultado. Es importante observar que haciendo esto conseguimos ahorrar tiempo, al no tener que procesar

todas aquellas ocurrencias de la palabra que se encuentran antes de la tercera ocurrencia de ⟨*libro* y que no aportan nada a la búsqueda (ver ocurrencias de *viaje* encerradas en rectángulos en la Fig. 4). Tras esto, se repite el proceso completo tomando la primera ocurrencia de *viaje* posterior a la etiqueta de apertura de la ocurrencia de *libro* con la que hemos operado (si se permite el anidamiento de elementos iguales), o bien posterior a su etiqueta de cierre (si no se permite el anidamiento de elementos iguales). Con ello conseguimos, nuevamente, saltar aquellas ocurrencias de *libro* que no contienen ninguna ocurrencia de *viaje*, ahorrando así su procesamiento (ver ocurrencias de *libro* encerradas en un rectángulo en la Fig. 4). Nótese que este mismo procedimiento puede ser utilizado con cualquier otra restricción estructural, y cuando la condición sobre el contenido se aplica sobre una frase o incluso el valor de un cierto atributo.

5. Resultados experimentales

Para evaluar la eficiencia del XWT resolviendo consultas *flexibles* se han realizado una serie de experimentos[2]. Con este propósito se han utilizado tres documentos XML diferentes[3], cuyas propiedades (tamaño (en MBytes), número de elementos XML (EN)(x10^3), máximo nivel de anidamiento (MP), ratio de compresión (R, en %), y tiempo de compresión (TC) y descompresión (TD) (en seg.)) se muestran en el lado izquierdo de la Fig. 5. Se puede observar cómo el XWT es capaz de representar cada documento ocupando únicamente el 30 %-35 % de su tamaño original. Para correr los experimentos hemos usado una implementación del XWT que gasta un 3 % de espacio a mayores en las estructuras de contadores parciales que permiten agilizar las operaciones de *rank* y *select*.

Fig. 5. Características y propiedades de compresión de los documentos utilizados y tiempos medios de ejecución para *below*

doc.	tam.	EN	MP	R	TC	TD
nasa	23.89	476	8	31.64	2.90	0.28
0.5d	55.32	832	12	32.18	6.65	0.66
1d	111.12	1,666	12	31.91	12.46	1.32

	BELOW	nasa FR (ms)	nasa HP (ms)	nasa HJ (ms)	0.5d FR (ms)	0.5d HP (ms)	0.5d HJ (ms)	1d FR (ms)	1d HP (ms)	1d HJ (ms)
d_2	f_{baja}	7.30	10.47	**6.66**	37.02	44.84	**17.70**	74.34	89.46	**33.65**
	f_{alta}	33.26	38.29	**29.00**	93.16	115.41	**55.36**	194.73	237.77	**112.00**
d_3	f_{baja}	7.73	10.54	**6.68**	37.81	45.88	**17.78**	75.14	93.92	**35.74**
	f_{alta}	33.46	38.66	**29.46**	97.64	124.75	**56.13**	200.52	241.81	**113.22**

Distinguimos dos escenarios de prueba diferentes. Por un lado, evaluamos el comportamiento del XWT a la hora de resolver consultas estructurales con las restricciones *below* y *near* (p.ej. *//libro* $BELOW_3$ *//ref*). Por otro lado, combinamos dichas restricciones con una búsqueda por contenido dentro de la misma consulta (p.ej. *//libro* [$contains(., CERI)$] $BELOW_3$ *//ref*). En ambos casos, los conjuntos de consultas de prueba (formados cada uno por 20 consultas) se han dividido según la frecuencia f (alta o baja) de los elementos de la consulta, de acuerdo con las características de cada documento, y se han considerado dos distancias máximas, d, entre elementos, 2 (d_2) y 3 (d_3).

En la parte derecha de la Fig. 5, se presentan los tiempos medios de ejecución para *below* utilizando las estrategias heurísticas de optimización *a*) (FR), *b*)

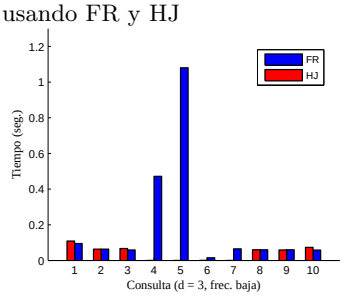
[2] En una máquina Intel®Pentium®Core 2 Duo 2.13 GHz, 4 GB dual-channel DDR-667Mhz RAM, con Ubuntu 8.04 GNU/Linux (kernel version 2.6.24.23). El compilador usado fue *gcc* version 4.2.4.

[3] `nasa` (http://xml.nasa.gov/); `0.5d` y `1d` (*XMark Project* - http://monetdb.cwi.nl/xml/)

(HP) y *c*) (HJ) descritas en la Sección 4. Como se puede observar, XWT invierte tiempos del orden de unos pocos milisegundos para resolver las consultas, independientemente de la estrategia. Sin embargo, es posible darse cuenta también que la estrategia que mejor resultados ofrece es HJ (ver valores en negrita). Esto es debido a que tomar el elemento menos frecuente (FR) no siempre es una buena opción, ya que puede suceder que éste tenga un número arbitrariamente grande de descendientes, los cuales habría que chequear si dicho elemento se encuentra en el lado izquierdo de la consulta. En ese caso, optar por el más frecuente, que estaría en el lado derecho, puede suponer un ahorro de tiempo; ya que implicaría seguir una dirección *hijo-padre*, y chequear, a lo sumo, tantos ancestros como indicase la distancia. Por su parte, seguir siempre esta dirección, como se hace en HP, basándose en el hecho de que cada elemento tiene un sólo padre y por tanto teniendo que chequear sólo un *path* hasta la raíz, resulta demasiado estricto.

Fig. 6. Tabla de tiempos medios de ejecución para *near* y gráfica de tiempos para ejemplos de consultas *near* sobre el documento `0.5d` usando FR y HJ

NEAR		nasa		0.5d		1d	
		FR (ms)	HJ (ms)	FR (ms)	HJ (ms)	FR (ms)	HJ (ms)
d_2	f_{baja_1}	57.16	**11.72**	122.44	**69.07**	212.45	**139.47**
	f_{baja_2}	10.82	11.72	68.41	69.07	145.57	139.47
	f_{alta}	**56.32**	67.92	**129.91**	174.02	**272.08**	334.84
d_3	f_{baja_1}	58.91	**12.86**	179.24	**76.49**	583.55	**162.94**
	f_{baja_2}	12.46	12.86	78.82	76.49	167.57	162.94
	f_{alta}	**86.60**	100.77	**171.99**	207.05	**352.89**	424.86



Teniendo en cuenta los resultados obtenidos en *below*, para el caso de estudio de *near* se optó por utilizar las estrategias heurísticas FR y HJ, a la hora de resolver las operaciones *below* en las que se basa. En la tabla de la Fig. 6 se muestran los resultados (en negrita aparecen marcados los que mejores valores ofrecen, para cada caso). En ella se puede observar cómo efectivamente, si realizamos un análisis comparativo entre *below* y *near*, la diferencia de tiempos cuando incrementamos la distancia es mayor en *near* que en *below*. Cuando trabajamos con *below*, cada incremento en la distancia puede implicar chequear los descendientes de un nivel más abajo (en el caso de elementos que tienen suficiente nivel de anidamiento) o un ancestro más arriba (en el mejor de los casos). Sin embargo, en *near* incrementar la distancia implica visitar los ancestros de uno o más niveles hacia arriba y luego chequear todos sus descendientes hasta un determinado nivel. A medida que se toman ancestros más alejados, mayor es la posibilidad de estar cerca de la raíz, y por tanto generalmente, mayor el número de descendientes a chequear.

Si ahora nos centramos en comparar el efecto de usar las estrategias FR y HJ dentro de las operaciones *below* que realiza *near*, se obtienen conclusiones realmente interesantes. En el caso de emplear FR, las operaciones *below* se hacen, al igual que en la última parte de *near* (correspondiente al chequeo de descendientes de ancestros sucesivos), sobre el elemento menos frecuente. Esto implica que sólo es necesario localizar las ocurrencias del elemento menos frecuente para resolver la consulta *near*. Esta característica, sin embargo, no siempre puede ser

aprovechada cuando se utiliza HJ. De acuerdo con esta heurística, si en al menos una de las dos operaciones *below* incluídas dentro de *near* se toma como elemento sobre el cual operar aquél que no es el menos frecuente, se tienen que localizar las ocurrencias de ambos elementos implicados en la consulta, al ser necesarios los dos para la resolución global de *near*. Si nos fijamos en los valores obtenidos para el caso de consultas que trabajan sobre elementos de frecuencia alta (ver filas f_{alta} en la Fig. 6) vemos cómo resulta mejor emplear la estrategia FR, ya que cuando se trabaja con estas frecuencias, utilizar HJ y tener que localizar los elementos de ambos lados de la consulta, consume más tiempo que el que se pueda perder empleando FR, yendo siempre por el menos frecuente. Si ahora observamos los valores de consultas que trabajan sobre elementos de frecuencia baja (ver filas f_{baja_1} en la Fig. 6) se da una situación diferente; es la estrategia HJ la que obtiene mejores resultados. Sin embargo esto se debe, en parte, a la desviación sobre la media que provocan determinadas consultas. Es decir, a la hora de trabajar con elementos de frecuencia baja, si se utiliza FR, hay determinadas consultas en que operar con el elemento menos frecuente puede aumentar mucho los tiempos si éste tiene un número de descendientes elevado (tal y como se vio cuando se analizaron los resultados para *below*). En f_{baja_1} se presentan los tiempos medios de ejecución teniendo en cuenta estas consultas, mientras que en f_{baja_2} se muestran esos mismos tiempos sin considerar esos casos especiales. En esta última se puede apreciar cómo, sin tener en cuenta esas consultas, las estrategias FR y HJ darían resultados muy similares. Asimismo, en la gráfica de la Fig. 6 se muestran los tiempos de ejecución de algunas consultas para el documento `0.5d` empleando las estrategias FR y HJ, donde los casos 4, 5, 6 y 7 constituyen ejemplos de estas consultas especiales. En ellas se observa cómo el seguir la estrategia HJ evita tiempos elevados.

Tabla 1. Tiempo medio de ejecución de consultas por contenido combinadas con *below* y *near*

		nasa		0.5d	
		BELOW (ms)	NEAR (ms)	BELOW (ms)	NEAR (ms)
d_3	f_{baja}	<u>2.81</u>	<u>4.21</u>	202.13	3790.61
	f_{alta}	<u>21.79</u>	472.84	<u>49.32</u>	2140.43

En el segundo escenario hemos usado los mismos conjuntos de prueba que en el contexto previo (donde sólo se trabajaba con restricciones estructurales) y hemos añadido una restricción sobre el contenido de los elementos, empleando palabras con frecuencia entre 50 y 100, elegidas aleatoriamente de los respectivos vocabularios *no-tags* de palabras. En este caso, se ha tomado 3 como distancia máxima. En la Tabla 1, se muestran los tiempos medios de ejecución. Se puede apreciar que hay situaciones en las que el hecho de usar la estrategia de *salto* explicada en la Sección 4 produce un importante ahorro (comparar valores subrayados en Tabla 1 y en las tablas de las Fig. 5 y 6); aunque no siempre es posible beneficiarse de ella (p.ej. cuando los elementos de la consulta no están relacionados estructuralmente, o cuando el elemento sobre el que se establece la condición de *contenido* es el más frecuente). En este tipo de consultas es necesario desarrollar nuevas estrategias que permitan aprovechar toda la potencia del *salto*, también en esas situaciones.

6. Conclusiones y trabajo futuro

El XWT es una estructura para la representación comprimida ($\approx 35\,\%$) y autoindexada de documentos XML, cuyas propiedades hacen que cualquier operación que se pueda realizar sobre el documento original, se pueda resolver de forma mucho más eficiente sobre el XWT que si se hace sobre el propio documento XML plano o incluso comprimido. En [2], se mostró cómo usando esta estructura era posible resolver eficientemente consultas típicas de XPath por estructura y contenido. En este trabajo se muestra, a través de experimentos, cómo el buen comportamiento del XWT se mantiene a la hora de resolver también extensiones de XPath, que tratan de introducir cierta *flexibilidad* en las búsquedas, acercándolas más al ámbito de la RI. Se han desarrollado distintas estrategias heurísticas de optimización para la resolución de las consultas, y se ha estudiado su influencia en la eficiencia, comprobándose cuáles ofrecían mejores resultados. Pero es necesario continuar este estudio, trabajando en el desarrollo de nuevas y mejores estrategias. Especialmente, es nuestro interés desarrollar nuevos algoritmos que extiendan los beneficios de la estrategia de *salto* en consultas que combinan restricciones por contenido y estructura, a todos los casos.

Referencias

1. XML 1.0, XPath 2.0 and XQuery 1.0. `http://www.w3.org/TR`.
2. N. R. Brisaboa, A. Cerdeira-Pena, G. Navarro. A compressed self-indexed representation of XML documents. *ECDL'09*, pp. 273–284, 2009.
3. N. R. Brisaboa, A. Fariña, S. Ladra, G. Navarro. Reorganizing compressed text. *SIGIR'08*, pp. 139–146, 2008.
4. N. R. Brisaboa, A. Fariña, G. Navarro, J. R. Paramá. (s, c)-dense coding: An optimized compression code for natural language text databases. *SPIRE'03*, pp. 122–136, 2003.
5. A. Campi, E. Damiani, S. Guinea, S. Marrara, G. Pasi, P. Spoletini. A fuzzy extension of the XPath query language. *J. Intell. Inf. Syst.*, 33(3):285–305, 2009.
6. T. T. Chinenyanga, N. Kushmerick. An expressive and efficient language for XML information retrieval. *J. Am. Soc. Inf. Sci. Technol.*, 53(6):438–453, 2002.
7. J. S. Culpepper, A. Moffat. Enhanced byte codes with restricted prefix properties. *SPIRE'05*, pp. 1–12, 2005.
8. E. Damiani, S. Marrara, G. Pasi. A flexible extension of XPath to improve XML querying. *SIGIR'08*, pp. 849–850, 2008.
9. E. Damiani, S. Marrara, G. Pasi. Fuzzyxpath: Using fuzzy logic and IR features to approximately query XML documents. *IFSA'07*, pp. 199–208, 2007.
10. E. Damiani, L. Tanca. Blind queries to XML data. *DEXA'00*, pp. 345–356, 2000.
11. N. Fuhr, K. Grobjohann. Xirql: a query language for information retrieval in XML documents. *SIGIR'01*, pp. 172–180, 2001.
12. E. Moura, G. Navarro, N. Ziviani, R. Baeza-Yates. Fast and flexible word searching on compressed text. *TOIS*, 18(2):113–139, 2000.
13. K. Sadakane, G. Navarro. Fully-functional static and dynamic succinct trees. *CoRR*, abs/0905.0768, 2009.
14. A. Theobald, G. Weikum. Adding relevance to XML. *WebDB'00*, pp. 105–124, 2001.

Uso de mapas semánticos para la búsqueda crosslingüe de oraciones paralelas

Rafael E. Banchs and Marta R. Costa-jussà
Barcelona Media Innovation Center
Av Diagonal 177, 9th floor, 08018 Barcelona
{rafael.banchs,marta.ruiz}@barcelonamedia.org

Resumen

Este trabajo presenta el uso de una técnica de recuperación de información crosslingüe basada en escalamiento multidimensional para la identificación de oraciones paralelas entre lenguas diferentes. El método propuesto permite hacer una reducción no-lineal del espacio de representación de las oraciones que se puede aprovechar para identificar similitudes semánticas entre conjuntos de oraciones en distintas lenguas. La técnica se ilustra con una colección pentalingüe extraída de la Constitución Española, la cual está disponible en las cuatro lenguas oficiales del estado español e inglés. Presentamos una evaluación comparativa entre nuestro método y un sistema de búsqueda crosslingüe basado en la traducción automática de las consultas. Los resultados muestran que nuestro sistema mejora consistentemente en las 20 direcciones experimentales de búsqueda crosslingüe que permite nuestra colección de datos.

1. Introducción

La recuperación de información crosslingüe permite obtener resultados en una lengua destino a partir de una búsqueda o consulta realizada en una lengua fuente diferente. Debido al aumento de información multilingüe en Internet, la necesidad de avanzar en esta área ha tomado especial relevancia recientemente. En particular, la investigación en recuperación de información crosslingüe se ha visto apoyada por eventos como el Cross-Language Evaluation Forum (CLEF) y el NTCIR Asian Language Evaluation.

Básicamente, hay dos maneras fundamentales de afrontar la recuperación de información crosslingüe: mediante técnicas basadas en traducción automática, y mediante técnicas basadas en interlingua (Kishida, 2005).

Por un lado, las técnicas basadas en traducción automática se usan para traducir, o bien la consulta, o bien la colección de documentos. En cuanquiera de estos dos casos, la recuperación de información crosslingüe se reduce a un problema de recuperación de información monolingüe. No obstante, debido a la enorme extensión que comúnmente presentan las colecciones multilingües, resulta más práctico realizar la traducción de la consulta (Chen and Bao, 2009).

Por otro lado, los métodos basados en interlingua se usan para asociar textos relacionados con contenidos en diferentes lenguas a través de representaciones semánticas que son independientes de la lengua. Algunas técnicas convencionales de recuperación de información crosslingüe basadas en interlingua usan indexación semántica latente (LSI) para construir una representación vectorial multilingüe de una colección paralela de documentos (Dumais et al., 1996). Una vez construída esta representación vectorial, se pueden proyectar en ella nuevos documentos y consultas, y la tarea de recuperación se lleva a cabo usando una métrica de similaridad o distancia.

Este trabajo, como extensión de (Banchs and Costa-jussa, 2009), propone usar una técnica de mapeado semántico para implementar un sistema de recuperación de información crosslingüe. Concretamente, la tarea a ser considerada consiste en la identificación crosslingüe de oraciones paralelas. Es decir, dada una oración en la lengua L_A, el objetivo es encontrar su equivalente en cualquier otra lengua diferente L_B. Esta tarea es importante para ciertas aplicaciones como, por ejemplo, la recopilación de texto paralelo a nivel de oración (Utiyama and Tanimura, 2007) o la detección de plagio (Potthast et al., 2009).

Las técnicas de mapeado semántico se han usado tradicionalmente para asociar conceptos y términos relacionados (Evans et al., 1998), y se ha verificado que el uso de proyecciones no-lineales en este contexto constituye una aproximación más eficiente que el uso de métodos lineales. La idea fundamental de la metodología propuesta es construir un mapa semántico para cada lengua disponible en la colección y aproximar el problema de recuperación de información crosslingüe mediante el aprovechamiento de las similaridades entre los diferentes mapas. La construcción de estos mapas, no obstante, requiere la disponibilidad de colecciones de documentos paralelos.

El presente artículo se organiza de la siguiente manera. En la sección 2 se describe la metodología de recuperación de información crosslingüe propuesta. En la sección 3 se ilustra dicha metodología mediante la realización de experimentos sobre la colección pentalingüe de la Constitución Española. También, en esta sección, se compara la metodología propuesta con la técnica de recuperación de información crosslingüe basada en traducción automática de consultas, mostrando que la técnica propuesta supera a la de referencia. Finalmente, en la sección 4 se presentan las conclusiones más relevantes y los próximos pasos de nuestra

investigación.

2. Mapeado semántico

La idea principal del método de recuperación de información crosslingüe propuesto se centra en el problema de mapeado semántico. En este trabajo, proponemos usar técnicas no-lineales de proyección, conocidas como escamiento multidimensional (MDS), para construir mapas semánticos de documentos[1] en lugar de términos o conceptos. Si las representaciones obtenidas realmente responden a relaciones semánticas entre documentos, entonces podemos esperar que para una colección paralela de documentos, se obtengan mapas similares para las distintas lenguas. Esta sección explora e ilustra esta idea en profundidad.

El MDS constituye un método de proyección no-lineal para la visualización de datos que se puede usar también como una técnica de reducción de espacio (Cox and Cox, 2001). Dado un conjunto de relaciones entre los elementos de una colección, el objetivo del MDS es encontrar una representación de menor dimensión para la colección de manera que las relaciones entre los elementos se preserven de la mejor manera posible. Lo interesante sobre MDS es que las relaciones entre los elementos de la colección en cuestión pueden ser tanto de naturaleza cuantitativa (similaridades basadas en una métrica de distancia) como de naturaleza de cualitativa (relaciones ordinales o jerárquicas).

Dada una colección monolingüe, el proceso de cómputo de un mapa semántico mediante MDS se puede definir en tres pasos:

- obtener una representación vectorial para la colección mediante el estándard TF-IDF (Salton and Buckley, 1988);
- construir una matriz de similaridad para los documentos usando las distancias coseno (o alguna otra métrica similar) entre sus correspondientes vectores; y
- construir un mapa semántico de dimensión reducida para los documentos en la colección usando MDS.

Por otro lado, el procedimiento propuesto para la implementación de un sistema de recuperación de información crosslingüe mediante el uso de mapas semánticos se puede resumir en los siguientes tres pasos (Banchs and Costa-jussa, 2009):

[1] Tal y como se indicó en la introducción, la tarea a ser considerada en el presente trabajo es la identificación crosslingüe de oraciones paralelas; en este sentido, cuando hablamos de documentos en la presente sección y subsiguientes secciones del trabajo nos estaremos refiriendo realmente a oraciones.

- seleccionar con una colección multilingüe de documentos (conjunto de entrenamiento) y construir el mapa de recuperación usando MDS a partir de una de las lenguas;
- proyectar nuevos documentos y consultas desde cualquier lengua fuente en el mapa de recuperación usando proyecciones lineales diseñadas para tal fin; y
- extraer documentos relevantes para una consulta del mapa de recuperación usando una métrica de distancia.

La figura 1 muestra una representación esquemática de este procedimiento.

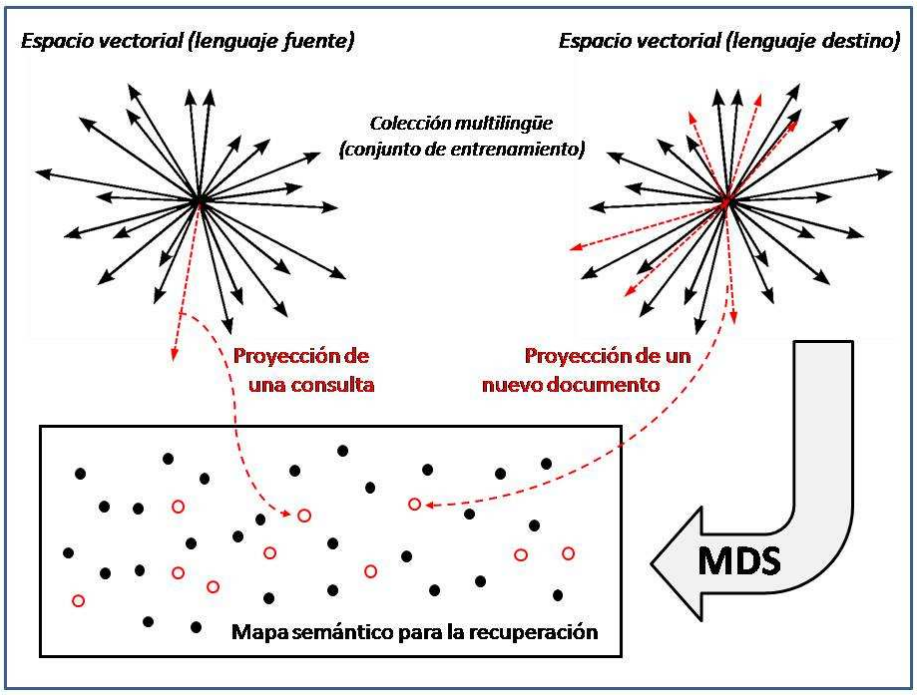


Figura 1: Representación esquemática del método de recuperación de información crosslingüe basado en mapas semánticos

Una vez construido el mapa semántico de recuperación, podemos inferir una transformación lineal T para proyectar en dicho mapa documentos o consultas desde un espacio original en cualquier lengua, como sigue:

$$M = TD \longrightarrow T = MD^{-1} \quad (1)$$

donde D es una matriz cuadrada NxN que contiene las distancias entre los documentos de entrenamiento en el espacio original (matriz de similaridad de documentos) y M es una matriz KxN que contiene las coordenadas de los documentos de entrenamiento una vez proyectados en el mapa semántico de dimensión reducida, siendo N el número total de documentos contenido en el conjunto de entrenamiento y K la dimensionalidad del mapa semántico construido.

Tal y como se explica en (Banchs and Costa-jussa, 2009), la transformación lineal descrita en (1) puede ser de naturaleza crosslingüe, para lo cual M se debe calcular en la lengua de recuperación y D se debe calcular en la lengua original del documento o consulta que se desea proyectar en el mapa. De esta forma, cualquier nuevo documento o consulta se puede situar en el mapa de recuperación utilizando la siguiente expresión:

$$m = Td \tag{2}$$

donde d representa un vector que contiene las distancias entre el documento (o consulta) ya proyectado en el conjunto de documentos de entrenamiento en la dimensión original, T es la matriz de transformación definida en (1), y m es el vector que contiene el resultado de las coordenadas del documento (o consulta) a proyectar en el mapa semántico de dimensión reducida.

Finalmente, como se pueden generar tantos mapas de recuperación como lenguas diferentes hay en la colección multilingüe, proponemos una nueva variante de nuestro método que combina diferentes mapas usando una votación. De acuerdo con esta estrategia, se construye un mapa de recuperación para cada lengua de la colección. A continuación, cada uno de los documentos y consultas del conjunto de evaluación son proyectados sobre cada uno de los mapas disponibles y se calculan las similaridades entre consultas y documentos. Como resultado se obtiene un ordenamiento para cada consulta, y por cada mapa de recuperación disponible, de todos los documentos que constituyen el conjuto de evaluación. Finalmente, se construye un ordenamiento global para cada consulta mediante la votación mayoritaría de cada uno de los ordenamientos obtenidos para dicha consulta en los distintos mapas.

3. Experimentos

Como mencionamos en la introducción, en este trabajo nos centramos en la tarea de la identificación crosslingüe de oraciones paralelas. En esta tarea en particular, se usa como consulta una oración en una lengua fuente, y se pretende recuperar esta misma oración en una o varias lenguas destino diferentes.

3.1. Colección de datos

La colección de datos utilizada es un conjunto pentalingüe de oraciones que se extrajo de la Constitución Española[2]. Las cinco lenguas en las que está disponible esta colección son: castellano, catalán, gallego, euskera e inglés.

Los textos de la Constitución están organizados en 169 artículos más algunas disposiciones reguladoras adicionales. Para la tarea, segmentamos todos los textos en oraciones. Se aplicó un filtro por longitud para eliminar todas aquellas oraciones de menos de 5 palabras. Esto se hizo con el fin de eliminar títulos y cualquier otro tipo de información no relevante. Posteriormente se llevó a cabo una randomización de las oraciones resultantes y se seleccionaron 200 oraciones (con sus respectivas paralelas en todas la lenguas) como conjunto de evaluación. Las tablas 1 y 2 resumen, respectivamente, las principales estadísticas de la colección completa y del conjunto de evaluación seleccionado. Finalmente, la tabla 3 muestra un ejemplo específico de oración extraído de la colección multilingüe.

Colección completa	Inglés	Castellano	Catalán	Euskera	Gallego
Num. oraciones	611	611	611	611	611
Num. de palabras	15285	14807	15423	10483	13760
Vocabulario	2080	2516	2523	3633	2667
Long. media oración	25.01	24.23	25.24	17.16	22.52

Cuadro 1: Estadísticas básicas del conjunto de datos completo.

Colección de test	Inglés	Castellano	Catalán	Euskera	Gallego
Num. oraciones	200	200	200	200	200
Num. de palabras	4667	4492	4669	3163	4175
Vocabulario	1136	1256	1273	1618	1316
Long. media oración	23.34	22.46	23.34	15.82	20.88

Cuadro 2: Estadísticas básicas del conjunto de datos de evaluación.

3.2. Evaluación comparativa

En esta subsección, comparamos la metodología propuesta con el método de recuperación de información crosslingüe basado en la traducción automática de

[2] www.la-moncloa.es

Lengua	Ejemplo de oración
Inglés	The capital of the State is the city of Madrid.
Castellano	La capital del Estado es la villa de Madrid.
Catalán	La capital de l'Estat és la vila de Madrid.
Euskera	Estatu hiriburua Madrid hiria da.
Gallego	A capital do Estado é a vila de Madrid

Cuadro 3: Ejemplo de oración de la Constitución Española extraído de la colección multilingüe.

las consultas (Chen and Bao, 2009). Como ya se mencionó, la tarea consiste en recuperar una oración en cualquiera de las 5 lenguas disponibles usando la misma oración en cualquiera de las otras 4 lenguas como consulta. La calidad de los sistemas se evalúa en términos de los aciertos en la primera posición (top-1) y las primeras cinco posiciones (top-5) de la lista de oraciones recuperadas para cada consulta, usando el conjunto de evaluación descrito en la tabla 2.

Para construir el sistema de recuperación de información crosslingüe basado en MDS, usamos 400 oraciones paralelas seleccionadas al azar de las 411 que quedaron al extraer las 200 correspondientes al conjunto de evaluación. Esta subcolección de 400 oraciones se usó para construir los mapas. Se construyó un mapa para cada una de las 5 lenguas disponibles en la colección, fijando la dimensión de los mapas en 350, lo cual implica una reducción de dimensionalidad entre un 83 % (en el caso del inglés) y un 90 % (en el caso del euskera). Siguiendo (1) y (2), se construyeron las matrices de transformación y se ubicaron todas las oraciones de evaluación en las 5 lenguas en cada uno de los mapas. Finalmente, se extrajeron las oraciones de evaluación más similares entre sí usando la distancia coseno como métrica de similaridad, obteniendo un ordenamiento individual para cada mapa. Dados estos ordenamientos individuales, se implementó la combinación por votación mayoritaria y se obtuvo un ordenamiento global. La tabla 4 muestra los resultados para este ordenamiento global.

Como sistema de referencia usamos el sistema basado en traducción de consultas (Chen and Bao, 2009). Este sistema implementa la recuperación de información crosslingüe concatenando un sistema de traducción automática con un sistema de búsqueda monolingüe. En nuestro caso, para la traducción de las consultas usamos la plataforma $Opentrad$[3] (Ramírez-Sánchez et al., 2006). Este sistema proporciona traducciones en el estado-del-arte entre las lenguas oficiales del estado español, el inglés y otras lenguas adicionales. Hay que tener en cuenta que este sistema no proporciona traducciones directamente desde cualquier lengua a

[3] www.opentrad.com

cualquier otra. Así pues, en los casos de gallego-catalán tuvimos que hacer una traducción en cascada pasando por el castellano. Adicionalmente, en el caso del euskera, el sistema proporciona traducciones pero sólo a nivel de oración, no pudiéndose automatizar para la traducción de de conjuntos de oraciones. Por esta razón, las 200 oraciones del euskera no fueron consideradas para los experimentos con este método. Por otro lado, el sistema recuperación de información monolingüe se implementó usando $SOLR$, que es una plataforma de código abierto basado en la librería $Lucene$[4].

La tabla 4 resume los resultados de los dos sistemas evaluados, tanto el propuesto (MDS) como el basado en traducción de consultas (MT-IR), para las 25 direcciones posibles y para cada una de las métricas de evaluación usadas (top-1 y top-5).

Lengua fuente	Sistema	Lengua destino									
		Inglés		Castellano		Catalán		Euskera		Gallego	
		top-1	top-5	top-1	top-5	top-1	top-5	top-1	top-5	top-1	top-5
Inglés	MT-IR	100	100	95.0	97.5	92.0	96.0	-	-	93.0	96.0
	MDS	100	100	**96.0**	**99.5**	**96.5**	**100**	**75.5**	**91.0**	**95.5**	**99.0**
Castellano	MT-IR	95.0	98.5	100	100	100	100	-	-	99.0	100
	MDS	**97.5**	**100**	100	100	100	100	**78.0**	**94.5**	**99.5**	100
Catalán	MT-IR	91.0	97.0	100	100	100	100	-	-	93.5	97.0
	MDS	**96.5**	**99.5**	100	100	100	100	**74.0**	**93.5**	**99.5**	**99.5**
Euskera	MT-IR	-	-	-	-	-	-	100	100	-	-
	MDS	**76.0**	**92.5**	**81.5**	**94.5**	**79.0**	**94.5**	100	100	**79.5**	**93.5**
Gallego	MT-IR	92.0	96.0	99.5	99.5	83.5	90.5	-	-	100	100
	MDS	**94.5**	**99.5**	**99.5**	**99.5**	**99.5**	**99.5**	**77.0**	**95.0**	100	100

Cuadro 4: Resultados comparativos entre el sistema propuesto (MDS) y el basado en traducción de consultas (MT-IR).

De la tabla 4 se pueden extraer las siguientes observaciones:

- Usando las dos métricas propuestas (top-1 y top-5), el método propuesto (MDS) mejora la técnica de referencia en todos los casos.
- Se observa que el método de traducción de consultas se ve perjudicado en los casos en que se usa una concatenación de sistemas de traducción (específicamente para el par catalán-gallego). La traducción en cascada generalmente concatena errores con lo que se obtiene una caída en la calidad de traducción. Así pues, en estos experimentos vemos como el comportamiento de este método depende de la calidad de la traducción.
- Se observa que, en general, los sistemas de recuperación de información crosslingüe obtienen mejores resultados cuanto más semejanza tienen las

[4]http://lucene.apache.org/solr/tutorial.html

lenguas (ver el caso de las lenguas romances castellano-catalán-gallego respecto a los pares de lenguas más distantes castellano-euskera o gallego-inglés).

Después de hacer un análisis de aquellas oraciones que no recuperan correctamente su oración paralela para cada uno de los métodos utilizados, hemos descubierto que ambos métodos se equivocan siempre en oraciones diferentes. Esto se puede justificar por la naturaleza diferente de ambos métodos. De hecho, un análisis más detallado revela que la naturaleza de los errores cometidos por cada método son muy diferentes entre sí.

MT-IR	
CONSULTA:	3.ª Ordenación del territorio, urbanismo y vivienda.
SALIDA MT:	3.ª Ordenación Of the **territory**, urbanismo and house.
SALIDA IR:	The Constitutional Court has jurisdiction over the whole Spanish **territory** and is entitled to hear:
REFERENCIA:	3.ª Town and country planning and housing.
CONSULTA:	La ley electoral determinará las causas de inelegibilidad e incompatibilidad de los Diputados y Senadores, que comprenderán en todo caso:
SALIDA MT:	The electoral **law will** determine the causes of inelegibilidad and incompatibility of the Deputies and Senators, that **will** comprise, anyway:
SALIDA IR:	Once the Statute is approved, the King **will** sanction it and promulgate it as **law**
REFERENCIA:	The Electoral Act shall establish grounds for ineligibility and incompatibility for Members of Congress and Senators, which shall in any case include those who are:
MDS	
CONSULTA:	La declaración de inconstitucionalidad de una norma jurídica con rango de ley, interpretada por la jurisprudencia, afectará a ésta, si bien la sentencia o sentencias recaídas no perderán el valor de cosa juzgada.
SALIDA:	**Those declaring the unconstitutionality of** an act or of a statute **with the force of an act and** all those which are **not limited to** the acknowledgment of an individual right, shall be fully binding on all persons.
REFERENCIA:	**A declaration of unconstitutionality of** a legal provision **having the force of an act and** that has already been applied by the Courts, shall also affect the case-law doctrine built up by the latter, but the decisions handed down **shall not lose** their status of res judicata.
CONSULTA:	Las Comunidades Autónomas designarán además un Senador y otro más por cada millón de habitantes de su respectivo territorio.
SALIDA:	Under no circumstances **shall** a federation of **Self-governing Communities** be allowed
REFERENCIA:	The **Self-governing Communities shall**, in addition, appoint one Senator and a further Senator for every million inhabitants in their respective territories.

Cuadro 5: Análisis de las salidas de los métodos de referencia (MT-IR) y propuesto (MDS).

La tabla 5 presenta un par de ejemplos de salidas erróneas del sistema de referencia (MT-IR) y del sistema propuesto (MDS), para el caso en el que el castellano era la lengua fuente y el inglés era la lengua destino. En el primer caso, la consulta es la salida del traductor. En el segundo caso, la consulta es la oración en castellano de la Constitución. Podemos observar que en el caso del MT-IR, los errores parecieran derivarse de la existencia de palabras poco frecuentes que tiene un gran

peso (en términos de TF-IDF) y que coinciden en la consulta y la oración recuperada. En cambio, en el caso del MDS, se puede observar una mayor semejanza entre la oración que devuelve el sistema y la que debería haber devuelto tanto en términos de palabras coincidentes, como del tema central de la oración y su semántica.

4. Conclusiones y trabajo futuro

Se ha presentado un procedimiento de mapeado semántico para identificar oraciones paralelas en distintas lenguas. El método se basa en el uso de una técnica no-lineal de reducción de espacio (escalamiento multidimensional) para identificar relaciones semánticas entre oraciones dada una colección multilingüe. Adicionalmente, el método propuesto implementa una combinación de los resultados obtenidos con los distintos mapas construidos, la cual permite un mejor aprovechamiento de la información multilingüe de la que se dispone. Se ha evaluado y comparado nuestro método con una estrategia estándar de recuperación de información crosslingüe, la basada en traducción de consultas, sobre una colección pentalingüe extraída de la Constitución Española. Los resultados demuestran que la técnica propuesta mejora consistentemente la técnica de referencia.

Como trabajo futuro proponemos seguir explorando la utilidad y beneficios de la técnica propuesta en las siguientes direcciones:

- la evaluación y prueba de su rendimiento en un escenario de entrenamiento en el que se utilice una colección multilingüe de documentos comparables, en lugar de paralelos;
- el estudio y evaluación de otras técnicas alternativas para la proyección de consultas y nuevos documentos en los mapas semánticos previamente construídos; y
- la evaluación del rendimiento del método propuesto, tanto en términos de calidad como de eficiencia, al utilizar colecciones paralelas de datos de mayor tamaño.

Bibliografía

R. E. Banchs and M. R. Costa-jussa. 2009. Extracción crosslingüe de documentos usando mapas semánticos no-lineales. *Revista del Procesamiento del Lenguaje Natural, SEPLN*, 43:169–176.

J. Chen and Y. Bao. 2009. Cross-language search: The case of google language tools. *First Monday*, 14(3-2).

M.F. Cox and M.A.A. Cox. 2001. *Multidimensional Scaling*. Chapman and Hall.

S. T. Dumais, T. K. Landauer, and M. L. Littman. 1996. Automatic cross-linguistic information retrieval using latent semantic indexing. In *SIGIR96 Workshop on Cross-Linguistic Information Retrieval*.

D.A. Evans, S.K. Handerson, I.A. Monarch, J. Pereiro, L. Delon, and W.R. Hersh. 1998. Mapping vocabularies using latent semantics. *G. Grefenstette (ed.) Cross-Language Information Retrieval*, pages 63–80.

K. Kishida. 2005. Technical issues of cross-language information retrieval: a review. *Information Processing and Management*, 41(3):433–455.

M. Potthast, B. Stein, A. Eiselt, A. Barrón, and P. Rosso. 2009. Overview of the 1st international competition on plagiarism detection. In *Workshop on Uncovering Plagiarism, Authorship, and Social Software Misuse*.

G. Ramírez-Sánchez, F. Sánchez-Martínez, S. Ortiz-Rojas, J. A. Pérez-Ortiz, and M. L. Forcada. 2006. Opentrad apertium open-source machine translation system: an opportunity for business and research. In *Proceeding of Translating and the Computer 28 Conference*, November.

G. Salton and C. Buckley. 1988. Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5).

M. Utiyama and M. Tanimura. 2007. Automatic construction technology for parallel corpora. *Journal of the National Institute of Information and Communications Technology*, 54(3):25–31.

RI con *n*-gramas: tolerancia a errores y multilingüismo*

Jesús Vilares, Miguel A. Alonso, Carlos Gómez-Rodríguez, Jorge Graña

Departamento de Computación, Universidade da Coruña
Campus de Elviña s/n, 15071 – A Coruña
{jvilares,alonso,cgomezr,grana}@udc.es

Resumen En este artículo presentamos el trabajo que en el Grupo LYS (Lengua y Sociedad de la Información) hemos venido desarrollando en fechas recientes en las áreas de recuperación de información tolerante a errores y recuperación de información multilingüe. El nexo común entre ambas líneas de investigación es el empleo de *n*-gramas de caracteres como unidad de procesamiento, en detrimento de soluciones más convencionales basadas en palabras o frases. El empleo de *n*-gramas nos permite beneficiarnos de su simplicidad, independencia del idioma y robustez. En el caso de recuperación tolerante a errores presentamos los resultados de un estudio que hemos realizado acerca de la robustez de nuestra solución frente a errores ortográficos y de escritura presentes en la consulta, así como la metodología que hemos diseñado para generar conjuntos de test empleando errores humanos reales. Finalmente, en el caso de la recuperación multilingüe, presentamos nuestro algoritmo de alineamiento a nivel de *n*-gramas para corpus paralelos, base de nuestra propuesta, y los resultados obtenidos en nuestros experimentos.

Key words: *n*-Gramas de caracteres; Recuperación de Información Tolerante a Errores; Recuperación de Información Multilingüe

1. Introducción

Formalmente, un *n*-grama de caracteres es una secuencia de caracteres dentro de una palabra [16]. Podemos, por ejemplo, dividir la palabra `"patata"` en cuatro 3-gramas superpuestos: `pat`, `ata`, `tat` y `ata`. La utilización de *n*-gramas tiene como principales ventajas su simplicidad, eficiencia, robustez, completitud e independencia del dominio. Estas ventajas no han pasado desapercibidas para la comunidad investigadora en Recuperación de Información (RI) y, por ejemplo, a día de hoy el empleo términos índice basados en *n*-gramas en entornos de RI es una práctica relativamente usual [10,16].

* Este trabajo ha sido parcialmente subvencionado por el Ministerio de Educación y Ciencia y FEDER (a través del proyecto HUM2007-66607-C04-03) y por la Xunta de Galicia (a través de los proyectos INCITE08-PXIB302179PR y PGIDIT07-SIN005206PR; y a través de la *"Red Gallega de Procesamiento del Lenguaje y Recuperación de Información"* y la *"Red Gallega de Lingüística de Corpus"*).

Mientras que los sistemas clásicos de RI suelen integrar conocimiento lingüístico y recursos tales como listas de *stopwords*, *stemmers*, lexicones, tesauros, etiquetadores [10], etc., la *tokenización* en n-gramas no precisa de ningún tipo de procesamiento o conocimiento más allá del propio texto: tanto consultas como documentos son simplemente *tokenizados* en n-gramas superpuestos en lugar de en palabras [11], para luego ser procesados como cualquier otro término por el motor de recuperación. Por la misma razón, esta aproximación es independiente del idioma y dominio, ya que no integra ningún tipo de recurso o conocimiento concreto sobre ellos, bien al contrario, el empleo de correspondencias a nivel de n-gramas constituye en sí mismo un mecanismo de normalización de términos que permite además trabajar con una gran diversidad de lenguas sin procesamiento alguno a mayores [11,16,10]. Finalmente está su propia robustez, consecuencia de la redundancia introducida por el propio proceso de *tokenización*.

Este trabajo presenta dos líneas de investigación en las que nuestro grupo, el Grupo LYS (Lengua y Sociedad de la Información)[1] de la Universidade da Coruña, ha venido trabajando recientemente. Ambas están basadas en el empleo de n-gramas de caracteres como unidad de procesamiento en dos contextos diferentes: la Recuperación de Información Tolerante a Errores (concretamente errores presentes en la consulta) y la Recuperación de Información Multilingüe.

El resto del artículo se estructura como sigue. En primer lugar presentamos en la Sección 2 nuestra propuesta para recuperación tolerante a errores. El diseño del sistema y de los experimentos realizados a tal efecto se describen en la Sección 3, analizando a continuación en la Sección 4 los resultados obtenidos. Nuestro trabajo en recuperación multilingüe se describe en la Sección 5, analizando seguidamente los resultados de nuestros experimentos en la Sección 6. Finalmente, presentamos nuestras conclusiones y trabajo futuro en la Sección 7.

2. Aplicación a tolerancia a errores

Si bien los modelos formales de RI están diseñados inicialmente para operar sobre textos sin errores, deberían ser capaces también de operar sobre textos con *errores ortográficos*[2], los cuales pueden reducir el rendimiento del sistema [6]. Actualmente, debido a ello y al fenómeno de globalización de la red, existe un creciente interés en la *Recuperación de Información Tolerante a Errores (RITE)*. Básicamente, se trata de sustituir los mecanismos de establecimiento de correspondencia exacta por otros más flexibles que permitan correspondencias aproximadas. En este contexto, el estado del arte recoge dos aproximaciones genéricas al problema [8]. La primera contempla la palabra como unidad de trabajo; la segunda, sin embargo, opera a nivel de subpalabra.

[1] http://www.grupolys.org

[2] Entendiendo como tales aquellos errores resultado del desconocimiento de la ortografía, errores tipográficos durante la escritura y errores derivados de la presencia de ruido en el proceso de generación (ej. OCR).

En nuestro caso estudiaremos el uso de n-gramas como términos índice para poder así aprovecharnos de su robustez al operar sobre consultas con errores. Como ya hemos comentado, el texto (tanto consultas como documentos) es meramente *tokenizado* en n-gramas superpuestos antes de enviárselo al sistema de recuperación de información. De esta forma, cualquier error ortográfico presente afectará únicamente a parte de ellos, por lo que los n-gramas restantes de esa palabra harán posible el establecimiento de correspondencias. De este modo el sistema estará mejor dotado para trabajar en ambientes con ruido, siendo no sólo capaz de hacer frente a errores ortográficos, sino también a palabras desconocidas, e incluso variantes morfológicas o de escritura, en contraposición a otros métodos clásicos de normalización como el *stemming*, la lematización o el análisis morfológico, que se ven todos ellos afectados negativamente por estos fenómenos.

3. Diseño de experimentos sobre tolerancia a errores

3.1. Marco de evaluación

Los trabajos previos en este área se centran en el inglés [6,2], un idioma de estructura léxica relativamente simple. En nuestro caso hemos optado por el español, una lengua mucho más compleja desde el punto de vista morfológico [17]. Para la realización de experimentos en este ámbito disponemos del corpus para español de la *robust task* del CLEF 2006 [12], formada por 454.045 teletipos de noticias (1,06 GB) de la agencia EFE[3]. En concreto, el conjunto de prueba está formado por los 60 *topics* del denominado *conjunto de entrenamiento* proporcionado para dicha *task*[4]. Éstos constan de tres campos: *título*, un breve título como su nombre indica; *descripción*, una somera frase de descripción; y *narrativa*, un pequeño texto especificando los criterios de relevancia. Siguiendo la política del CLEF, nuestros experimentos se han realizado empleando únicamente los campos *título* y *descripción*.

Asimismo, en todos los experimentos de este trabajo se ha empleado la plataforma de código abierto Terrier [15] como motor de recuperación, usando un modelo de ordenación InL2[5].

Como se ha apuntado, nuestra propuesta (que denominaremos *4gr*) se basa en la utilización de n-gramas de caracteres como términos de indexación en lugar de palabras. Para ello el texto es pasado a minúscula y se eliminan los signos de puntuación, pero no los ortográficos. El texto resultante es *tokenizado* en 4-grams [11], para ser finalmente procesado por el motor de indexación.

La *línea de base* con la que comparar nuestra propuesta será una aproximación clásica basada en *stemming* (que denominaremos *stm*) que emplea el *stemmer* Snowball[6], basado en el algoritmo de Porter, y la lista de *stopwords*

[3] Debemos precisar que los experimentos que aquí se muestran deben ser considerados *no oficiales* en tanto que no han sido validados por la organización del CLEF.

[4] *Topics* C050-C059, C070-C079, C100-C109, C120-C129, C150-159 y C180-189.

[5] Frecuencia inversa de documento con normalización 2 de Laplace.

[6] `http://snowball.tartarus.org`

proporcionada por la Universidad de Neuchâtel[7]. Ambos recursos son bien conocidos y de amplio uso entre la comunidad investigadora.

3.2. La generación de errores

Ambas aproximaciones han sido evaluadas mediante la introducción de errores ortográficos en los *topics* para poder así analizar su impacto en los resultados. Se ha probado un amplio rango de *tasas de error* T:

$$T \in \{0\,\%, 10\,\%, 20\,\%, 30\,\%, \ldots, 60\,\%\}$$

donde una tasa T implica que el $T\,\%$ de las palabras del *topic* incluyen errores[8], permitiéndonos así estudiar el comportamiento del sistema incluso para altas tasas de error propias de entornos ruidosos como aquellos en que la entrada se obtiene de dispositivos móviles o basados en escritura a mano (PDAs, bolígrafos digitales y tabletas digitalizadoras, por ejemplo), o incluso interfaces por habla.

Al contrario que en otras aproximaciones [14], los errores no han sido generados artificialmente, sino que hemos desarrollado una metodología que permite trabajar con errores humanos reales, mucho más complejos de generar y controlar, pero también mucho más próximos a la realidad. En una primera fase, se le pidió a un grupo de personas externo a nuestro trabajo que teclearan varias copias de los *topics* originales, preferiblemente tecleando rápido o en ambientes con distracciones (viendo la televisión, por ejemplo) y sin corregir los errores que pudiesen cometer. De esta forma se obtuvo un corpus base formado por 27 copias de los *topics*, donde una media del 7,70 % de sus palabras contenía errores de copia (i.e. T=7,70 %). En una segunda fase se incrementó dicho número de errores. Para ello se alinearon a nivel de palabra todas las copias, permitiendo así acceder, para cada posición, a todas las formas en que dicho término había sido tecleado por los copistas. De este modo se pudo identificar, si lo hubiese, el error más frecuente cometido al teclear aquella palabra en aquella posición, pudiendo así identificar errores de copia para un 65,62 % de los términos (i.e. T=65,62 %, un 60 % en la práctica). Finalmente, en una tercera fase, se generaron los conjuntos de prueba de forma que la tasa de error fuese incremental y acumulativa. Es decir, si un error dado aparece para T=20 %, deberá seguir existiendo para T=30 %, T=40 % y así sucesivamente, evitando de esta forma cualquier distorsión en los resultados. Para ello, todas las palabras que hubiesen sido tecleadas incorrectamente alguna vez se distribuyeron de forma aleatoria pero uniforme en 66 grupos[9]. De esta forma, si queremos generar un conjunto de prueba con una tasa de error T dada, basta con recorrer el texto del *topic* y quedarnos con la versión con errores si y sólo si dicha palabra pertenece a alguno de los T primeros grupos.

[7] `http://www.unine.ch/info/clef/`

[8] T=0 % corresponde al *topic* original. Asimismo, como veremos, el límite de T=60 % ha venido impuesto por el conjunto de prueba.

[9] La tasa máxima era T=65,62 %, lo que supone un grupo por cada 1 % de variación.

Cuadro 1. Resultados de nuestra propuesta para RITE.

	stm		***4gr***	
T	MAP	***%loss***	MAP	***%loss***
0	0,3427	–	0,3075	–
10	0,3289	-4,03	0,2908	-5,43
20	0,3049	-11,03	0,2767	-10,02
30	0,2804	-18,18	0,2642	-14,08
40	0,2194	-35,98	0,2430	-20,98
50	0,1789	-47,80	0,2254	-26,70
60	0,1374	-59,91	0,2061	-32,98
avg.	–	-29,49	–	-18,36

4. Resultados para tolerancia a errores

Los resultados obtenidos se muestran en la Tabla 1. Cada fila corresponde a una tasa de error T dada. Para cada configuración se muestra, por una parte, la precisión media o *mean average precision* obtenida (columna MAP), y por otra, la caída porcentual del rendimiento respecto a MAP para los *topics* originales (para T=0 %, columna *%loss*). La media de tales pérdidas se indica al final de la tabla en la fila *avg.*

En el caso de nuestra línea de base basada en *stemming* (*stm*), las cifras muestran un claro impacto negativo de los errores en el comportamiento del sistema incluso para las tasas de error más bajas, que se vuelve estadísticamente significativo[10] a partir de T=30 %. La pérdida de MAP media (fila *avg.*) es 29,49 %.

Respecto a los n-gramas (*4gr*), si bien el *stemming* obtiene unos mejores resultados iniciales en cuanto a MAP, nuestro planteamiento es menos sensible a los errores, de tal forma que al ir aumentando progresivamente la tasa de error, dicha diferencia se va reduciendo, llegando incluso a superar al *stemming* para $T \geq 40$ %. La pérdida de MAP, además, es ahora claramente menor, con una media de 18,36 % (*avg.*) frente al 29,49 % del *stemming*, poniéndose así en evidencia la robustez de la solución propuesta.

5. Aplicación a multilingüismo

La *Recuperación de Información Multilingüe (RIM)* es un caso particular de RI en el cual consultas y documentos están escritos en idiomas diferentes, por lo que se hace necesario algún tipo de fase de traducción intermedia empleando técnicas de *Traducción Automática (TA)* y así permitir el establecimiento de

[10] A lo largo de este trabajo se han empleado tests-t bilaterales sobre las MAP con α=0,05.

correspondencias. Sin embargo, al contrario que en los sistemas clásicos de TA, en las aplicaciones de RIM no es necesario respetar ciertas restricciones como la de devolver una única traducción o que ésta sea sintácticamente correcta [3].

Es por ello que muchos de estos sistemas emplean métodos de traducción palabra a palabra más sencillos. En esta dirección, el Johns Hopkins University Applied Physics Lab (JHU/APL) propuso ir todavía más lejos y relajar todavía más tales restricciones [10,11]. De esta forma, optan ya no por traducir palabras completas, sino que les basta traducir partes de ella, en concreto los n-gramas que la integran. Para ello emplean un algoritmo de traducción directa de n-gramas que permite traducir a nivel de n-grama de caracteres en lugar de a nivel de palabra, y que está basado en técnicas estadísticas para el alineamiento a nivel de n-grama de corpus paralelos en idiomas diferentes. Aunque desde un punto de vista lingüístico no estamos ante una traducción propiamente dicha, esta aproximación nos permite, en el caso de la RIM, extender las ventajas propias de la utilización de n-gramas como unidad de procesamiento también al proceso de traducción, permitiéndonos así evitar algunas de las limitaciones de las técnicas clásicas de traducción basadas en diccionarios, tales como la necesidad de normalizar las palabras o la imposibilidad de traducir palabras desconocidas. Además, como se trata de una solución que no emplea ningún tipo de procesamiento particular dependiente del idioma, puede ser empleada cuando la disponibilidad de recursos lingüísticos es reducida. Sin embargo, la propuesta original del JHU/APL adolecía de ser lenta, lo que constituía un problema a la hora de ensayar nuevas soluciones o modificaciones, además de integrar numerosas herramientas y recursos *ad-hoc*.

5.1. Descripción del sistema

Tomando como modelo el sistema propuesto por el JHU/APL [11], hemos desarrollado una solución propia intentando preservar las ventajas de la propuesta original pero evitando sus principales desventajas. Primeramente, en lugar de recursos *ad-hoc*, hemos optado por emplear recursos de libre distribución para así minimizar el coste de desarrollo y hacer el sistema más transparente. De este modo emplearemos, por ejemplo, la plataforma Terrier [15] como motor de recuperación y el bien conocido corpus paralelo Europarl [4] para el alineamiento.

Sin embargo, la principal diferencia la constituye el algoritmo de alineamiento de n-gramas, corazón de la solución, y que ahora consta de dos fases. Primeramente el corpus paralelo de entrada es alineado a nivel de palabra empleando la bien conocida herramienta estadística GIZA++ [13], obteniendo como salida las probabilidades de traducción entre las palabras de ambos idiomas. Este primer paso actúa como filtro, ya que sólo aquellos pares de n-gramas correspondientes a palabras alineadas serán considerados para su posterior procesamiento, en contraste con la aproximación original del JHU/APL, de granularidad mucho mayor al partir del corpus alineado a nivel de párrafo para luego procesar todos los n-gramas que contenían dos párrafos alineados. Además hemos introducido varios mecanismos de

filtrado con objeto de reducir el número de alineamientos ambiguos y así reducir el ruido introducido en el sistema. Por una parte, el alineamiento a nivel de palabra es bidireccional [5], es decir, aceptaremos un alineamiento *idiomaOrigen*→*idiomaDestino* (w_s, w_t), donde w_s y w_t denotan respectivamente la palabra origen y su traducción candidata, sólo si existe también el correspondiente alineamiento *idiomaDestino*→*idiomaOrigen* (w_t, w_s). Asimismo serán también desechados aquellos alineamientos cuya probabilidad sea menor que un umbral W=0,15. Esto permite, además, reducir notablemente el consumo de recursos del sistema, al disminuir drásticamente el número de pares de palabras a procesar: hasta un 70 % en el caso del alineamiento bidireccional y un 95 % en el caso del umbral.

Partiendo de los alineamientos de palabras obtenidos, en la segunda fase del algoritmo se procede a calcular las medidas de alineamiento entre n-gramas empleando para ello medidas estadísticas de asociación [9]. Esta solución permite acelerar el proceso de entrenamiento, al concentrar la complejidad en la fase de alineamiento a nivel de palabra y de este modo facilitar el poder probar nuevas medidas de asociación u otros procedimientos en la fase final de alineamiento a nivel de n-gramas propiamente dicha. A la hora de calcular las medidas de asociación estudiaremos las coocurrencias de los diferentes n-gramas que componen dos palabras previamente alineadas por GIZA++. Así, dado el par de n-gramas (g_s, g_t), donde g_s denota el n-grama en la lengua origen y g_t su traducción candidata, sus frecuencias de coocurrencia pueden organizarse en una *tabla de contingencia* resultante de clasificar sus coocurrencias entre ellos y con otros n-gramas presentes en la lista de entrada de palabras alineadas:

	$T = g_t$	$T \neq g_t$	
$S = g_s$	O_{11}	O_{12}	$= R_1$
$S \neq g_s$	O_{21}	O_{22}	$= R_2$
	$= C_1$	$= C_2$	$= N$

La primera fila corresponde a las observaciones donde la palabra en la lengua origen contiene el n-grama g_s, y la segunda fila a aquéllas en las que dicha palabra no lo contiene. Lo mismo ocurre para las columnas para la palabra en la lengua destino y g_t. Las cuentas de estas celdas se denominan *frecuencias observadas*[11]. R_1 y R_2 son las sumas parciales por fila de dichas frecuencias, y C_1 and C_2 las por columna. El número total de pares considerados, N, es la suma total de las frecuencias observadas.

Sin embargo, en este caso deberemos además ponderar la frecuencia de las observaciones en base a la probabilidad asociada a dichos alineamientos. Esto se debe a que no nos encontramos ante observaciones de coocurrencias *reales*, sino únicamente *probables*, ya que GIZA++ emplea un modelo de alineamiento estadístico que calcula la probabilidad de traducción para cada par de palabras

[11] O_{11}, por ejemplo, corresponde al número de alineamientos donde la palabra origen contiene g_s y su corrección candidata contiene g_t, mientras que O_{12} corresponde al número de alineamientos donde la palabra origen contiene g_s pero la corrección candidata no contiene g_t, etcétera.

que coocurran [13]. Por consiguiente, una misma palabra origen puede estar alineada con varias traducciones candidatas, con una probabilidad diferente para cada una. Tomemos como ejemplo el caso de las palabras en inglés `milk` y `milky`, y las españolas `leche`, `lechoso` y `tomate`; un posible alineamiento a nivel de palabra, con sus correspondientes probabilidades y n-gramas componente, podría ser:

source term	**candidate translation**	**prob.**
milk = {milk}	leche = {lech, eche}	0,98
milky = {milk, ilky}	lechoso = {lech, echo, chos, hoso}	0,92
milk = {milk}	tomate = {toma, omat, mate}	0,15

con lo que la tabla de contingencia resultante correspondiente a los pares de n-gramas *(milk, lech)* y *(milk, toma)* sería de la forma:

	$T =$ lech	$T \neq$ lech		$T =$ toma	$T \neq$ toma	
$S =$ milk	O_{11} =**1,90**	O_{12} =**4,19**	R_1 =**6,09**	O_{11} =**0,15**	O_{12} =**5,94**	R_1 =**6,09**
$S \neq$ milk	O_{21} =**0,92**	O_{22} =**2,76**	R_2 =**3,68**	O_{21} =**0**	O_{22} =**3,68**	R_2 =**3,68**
	C_1 =**2,82**	C_2 =**6,95**	N =**9,77**	C_1 =**0,15**	C_2 =**9,62**	N =**9,77**

Obsérvese que, por ejemplo, la frecuencia O_{11} correspondiente a *(milk, lech)* no es 2 como hubiera cabido esperar en otro caso, sino 1,90: el par aparece en dos alineamientos, `milk`–`leche` y `milky`–`lechoso`, pero dichas coocurrencias son ponderadas de acuerdo a la probabilidad de sus alineamientos:

$$O_{11} = 0{,}98 \text{ (de milk–leche)} + 0{,}92 \text{ (de milky–lechoso)} = \mathbf{1{,}90}$$

Una vez generada la tabla, podemos proceder a calcular las medidas de asociación entre cada par de n-gramas. En contraste con la aproximación original del JHU/APL [11], que emplea una medida *ad-hoc*, en nuestro caso hemos optado por medidas estándar: el *coeficiente de Dice* (*Dice*), la *información mutua* (*IM*) y el *log-likelihood* (*logl*), calculadas de la siguiente manera [9]:

$$Dice = \frac{2\,O_{11}}{R_1 + C_1} \qquad IM = log\frac{NO_{11}}{R_1 C_1} \qquad logl = 2\sum_{i,j} O_{ij}\, log\frac{NO_{ij}}{R_i C_j}$$

Retomando el ejemplo anterior, obsérvese que para el caso del coeficiente de Dice, por ejemplo, nos encontramos que la asociación para el par *(milk, lech)*, que es el correcto, es mucho mayor que para el par *(milk, toma)*, que es incorrecto:

$$Dice(\texttt{milk}, \texttt{lech}) = \tfrac{2*1{,}90}{6{,}09+2{,}82} = \mathbf{0{,}43} \quad Dice(\texttt{milk}, \texttt{toma}) = \tfrac{2*0{,}15}{6{,}09+0{,}15} = \mathbf{0{,}05}$$

6. Resultados para multilingüismo

Nuestra aproximación ha sido probada para el caso inglés→español empleando los *topics* en inglés y los documentos en español de la *robust task* del CLEF 2006 empleados en los experimentos de la Sección 3.1, salvo que en

Cuadro 2. Resultados de nuestra propuesta para RIM.

	stm	***4gr***$_{ES}$	***4gr***$_{EN}$	***Dice***$_N$	***Dice***$_T$	***IM***$_N$	***IM***$_T$	***logl***$_N$	***logl***$_T$
MAP	0,3427	0,3075	0,1314	0,2096	0,1589	0,1497	0,1487	0,2231	0,0893
0,00	0,7671	0,6831	0,2845	0,4957	0,4437	0,3782	0,4027	0,5489	0,2171
0,10	0,5974	0,5557	0,2181	0,3782	0,3053	0,3050	0,3000	0,4248	0,1394
0,20	0,5222	0,4722	0,1929	0,3310	0,2606	0,2315	0,2374	0,3620	0,1290
0,30	0,4447	0,4087	0,1610	0,2788	0,2033	0,1927	0,1877	0,2875	0,1082
0,40	0,3922	0,3365	0,1482	0,2320	0,1783	0,1737	0,1632	0,2425	0,0946
0,50	0,3364	0,2800	0,1387	0,2021	0,1535	0,1485	0,1458	0,2124	0,0855
0,60	0,2770	0,2466	0,1226	0,1729	0,1295	0,1269	0,1188	0,1769	0,0731
0,70	0,2241	0,2043	0,0958	0,1302	0,0896	0,0993	0,0980	0,1356	0,0655
0,80	0,1903	0,1722	0,0802	0,1013	0,0666	0,0718	0,0721	0,1015	0,0570
0,90	0,1435	0,1314	0,0704	0,0728	0,0476	0,0497	0,0476	0,0757	0,0540
1,00	0,0795	0,0695	0,0487	0,0452	0,0315	0,0179	0,0181	0,0445	0,0434

este caso los *topics* son los de lengua inglesa. Los parámetros del experimento son también los mismos que los descritos en la Sección 3.1.

El proceso de indexación es también el mismo pero el proceso de consulta sí varía, aunque siempre empleando 4-gramas. La consulta a traducir se descompone primero en n-grams, para luego sustituir dichos n-gramas por sus traducciones de acuerdo a un algoritmo de selección. La consulta traducida resultante es lanzada contra el sistema de recuperación. Actualmente existen dos *algoritmos de selección*: *(a) por rango*, que devuelve los N n-gramas traducción con las medidas de asociación más altas, para $N \in \{1, 2, 3, 5, 10, 20, 30, 40, 50, 75, 100\}$; *(b) por umbral*, que devuelve los n-gramas traducción cuya medida de asociación es superior o igual a un umbral T dado.

6.1. Resultados para el coeficiente de Dice

La Tabla 2 recoge un resumen de los resultados obtenidos en nuestros experimentos, mostrando las precisiones medias (MAP) obtenidas así como los resultados de *precisión respecto cobertura*.

En el caso del algoritmo de selección por rango empleando el coeficiente de Dice, los mejores resultados se obtuvieron con un número limitado de traducciones, siendo los mejores aquéllos para N=1, como se muestra en la columna *Dice*$_N$ de la Tabla 2.

A continuación estudiamos el caso del algoritmo de selección por umbral. En este caso, dado que el coeficiente de Dice toma valores en el rango $[0..+1]$, hemos optado por emplear una serie de umbrales T de la forma:

$$T \in \{0;\ 0,001;\ 0,01;\ 0,05;\ 0,10;\ 0,2;\ 0,3;\ 0,4;\ 0,5;\ 0,6;\ 0,7;\ 0,8;\ 0,9;\ 1\}$$

El mejor resultado, obtenido para T=0,40, se muestra en la columna *Dice*$_T$. Como se puede apreciar, los resultados no fueron tan buenos (significativamente) como los anteriores.

6.2. Resultados para la información mutua

En este caso, los resultados obtenidos para la selección por rango no fueron tan positivos como con el coeficiente de Dice. Los mejores, con N=10, se muestran en la columna IM_N.

En el caso de la selección por umbral, debemos tener ahora en cuenta que la medida IM puede tomar cualquier valor en el rango $(-\infty .. +\infty)$, donde además los valores negativos corresponden a pares de términos que se evitan mutuamente. De este modo, y para homogeneizar los test, los umbrales a emplear se calcularon de acuerdo a la siguiente fórmula:

$$T_i = \mu + 0{,}5\ i\ \sigma$$

donde T_i representa el i-ésimo umbral (con $i \in \mathbb{Z}^+$), μ representa la *media* de los valores de asociación obtenidos y σ representa su *desviación típica*. Nuestras pruebas demuestran que en el caso de selección por umbral, no hay diferencias significativas con los resultados de la selección por rango. Los mejores resultados, para $T = \mu$, se muestran en la columna IM_T.

6.3. Resultados para el *log-likelihood*

Para el algoritmo de selección por rango, los mejores resultados se obtuvieron limitando el número de candidatos, siendo aquéllos con N=1 los mejores, los cuales se muestran en la columna $logl_N$.

En el caso del algoritmo por umbral, como en el caso de la IM, no existe un rango fijo de valores posibles. Tras estudiar la distribución de los alineamientos de n-gramas obtenidos con respecto a sus valores de log-likelihood, se decidió emplear esta vez granularidades variables:

$$T_i = \begin{cases} \mu + 0{,}05\ i\ \sigma & -\infty < i \leq 2 \\ \mu + 0{,}50\ (i-2)\ \sigma & 2 < i < +\infty \end{cases}$$

donde T_i representa el i-ésimo umbral (con $i \in \mathbb{Z}$), μ representa la *media* de los valores de log-likelihood obtenidos y σ representa su *desviación típica*. Los resultados que obtuvimos fueron los más bajos de los todas las configuraciones estudiadas. Los mejores de ellos, para $T = \mu + 3\sigma$, se muestran en la columna $logl_T$.

Finalmente y con intención de completar nuestra evaluación, hemos comparado los resultados anteriores con varias líneas de base: una ejecución monolingüe en español empleando *stemming* (stm_{ES}); otra ejecución monolingüe en español empleando 4-gramas como términos ($4gr_{ES}$) que constituiría nuestro rendimiento objetivo deseado; y una última ejecución empleando 4-gramas lanzando directamente las consultas en inglés sin traducción alguna contra el índice en español ($4gr_{EN}$) que nos permite medir el impacto de las correspondencias casuales. Como podemos apreciar los mejores resultados fueron los obtenidos para el *log-likelihood* empleando el algoritmo de selección por rango, si bien la diferencia con aquéllos obtenidos con el coeficiente de Dice no

resultó ser significativa. Por otra parte, ambas aproximaciones se comportaron significativamente mejor que la información mutua. Los resultados obtenidos, pues, han sido muy positivos y alentadores, si bien necesitan todavía ser mejorados para seguir acercándose a nuestro rendimiento objetivo.

7. Conclusiones y trabajo futuro

Este trabajo plantea la utilización de n-gramas de caracteres como unidad de procesamiento en dos contextos diferentes: la Recuperación de Información Tolerante a Errores y la Recuperación de Información Multilingüe. El objetivo en ambos casos es beneficiarse de su simplicidad de uso e integración, su robustez y su independencia respecto al idioma y al dominio.

En el caso de la recuperación tolerante a errores ortográficos en las consultas, el empleo de n-gramas permite trabajar directamente sobre el texto con errores sin procesamiento alguno a mayores, ya que las correspondencias no se establecen ya a nivel de palabra completa, sino a nivel de sus subcadenas, posibilitando las correspondencias parciales y, por tanto, aumentando la robustez. De este modo los experimentos realizados mostraron que si bien las aproximaciones clásicas basadas en *stemming* son altamente sensibles a los errores ortográficos, los n-gramas confirmaron tener una robustez considerablemente mayor.

Asimismo, para realizar dichos experimentos se desarrolló una metodología que permite introducir de forma sencilla errores humanos reales en el conjunto de *topics* de entrada, posibilitando además elegir la tasa de error deseada, para así poder analizar el impacto de los mismos en el rendimiento del sistema.

En el caso de la recuperación multilingüe, este trabajo propone emplear n-grams no sólo como términos de indexación, como en el caso anterior, sino también como unidades de traducción. Para ello se describe un algoritmo para el alineamiento a nivel de n-grama de textos paralelos. Los alineamientos resultantes son empleados para la traducción de la consulta, también a nivel de n-grama. Tal aproximación permite evitar algunas de las limitaciones propias de las técnicas clásicas de traducción basadas en diccionarios, tales como la necesidad de normalizar las palabras o la imposibilidad de traducir palabras desconocidas. Además, como se trata de una solución que no emplea ningún tipo de procesamiento particular dependiente del idioma, puede ser empleada cuando la disponibilidad de recursos lingüísticos es reducida. Los resultados obtenidos han sido muy positivos y alentadores.

Con respecto al trabajo futuro, el principal problema de utilizar n-gramas como términos índice es el gran tamaño de los índices resultantes. Para reducir su tamaño queremos estudiar el empleo de técnicas de *pruning* [1] así como extender el concepto de *stopword* al caso de n-gramas para eliminar aquéllos más frecuentes y menos discriminantes. Tales *stop-n-grams* deberían ser generados automáticamente a partir de los textos de entrada [7] para así preservar su independencia respecto al idioma y el dominio. Pretendemos, además, ampliar nuestros experimentos a una mayor número de idiomas y tipos de consulta.

Referencias

1. D. Carmel, D. Cohen, R. Fagin, E. Farchi, M. Herscovici, Y.S. Maarek, and A. Soffer. Static index pruning for information retrieval systems. In *Proc. of SIGIR'01*, pages 43–50, 2001. ACM.
2. B. Croft, D. Metzler, and T. Strohman. *Search Engines: Information Retrieval in Practice*. Addison Wesley, 2009.
3. G. Grefenstette, editor. *Cross-Language Information Retrieval*, volume 2 of *The Kluwer International Series on Information Retrieval*. Kluwer Academic Publishers, 1998.
4. P. Koehn. Europarl: A Parallel Corpus for Statistical Machine Translation. In *Proc. of the 10th Machine Translation Summit*, pages 79–86, 2005. Corpus disponible en http://www.iccs.inf.ed.ac.uk/~pkoehn/publications/europarl/ (visitada en abril 2010).
5. P. Koehn, F.J. Och, and D. Marcu. Statistical phrase-based translation. In *Proc. of the 2003 Conf. of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (NAACL '03)*, pages 48–54, 2003. ACL.
6. K. Kukich. Techniques for automatically correcting words in text. *ACM Computing Surveys (CSUR)*, 24(4):377–439, 1992.
7. R.T.W. Lo, B. He, and I. Ounis. Automatically building a stopword list for an information retrieval system. In *Proc. of the 5th Dutch-Belgian Information Retrieval Workshop (DIR'05)*, 2005.
8. C.D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.
9. C.D. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing*. The MIT Press, 1999.
10. P. McNamee and J. Mayfield. Character N-gram Tokenization for European Language Text Retrieval. *Information Retrieval*, 7(1-2):73–97, 2004.
11. P. McNamee and J. Mayfield. JHU/APL experiments in tokenization and non-word translation. volume 3237 of *Lecture Notes in Computer Science*, pages 85–97. 2004.
12. A. Nardi, C. Peters, and J.L. Vicedo, editors. *Working Notes of the CLEF 2006 Workshop*, 2006. Disponible en http://www.clef-campaign.org (visitada en abril 2010).
13. F.J. Och and H. Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, 2003. Herramienta GIZA++ disponible en http://www.fjoch.com/GIZA++.html (visitada en abril 2010).
14. J. Otero, J. Vilares, and M. Vilares. Corrupted queries in Spanish text retrieval: error correction vs. n-grams. In *Proc. of ACM CIKM 2008 Workshop on Improving Non-English Web Searching (iNEWS'08)*, pages 39–46, 2008. ACM.
15. I. Ounis, C. Lioma, C. Macdonald, and V. Plachouras. Research directions in Terrier: a search engine for advanced retrieval on the web. *Novática/UPGRADE Special Issue on Web Information Access*, 8(1):49–56, 2007. Herramienta Terrier disponible en http://www.terrier.org (visitada en abril 2010).
16. A.M. Robertson and P. Willett. Applications of n-grams in textual information systems. *Journal of Documentation*, 54(1):48–69, January 1998.
17. M. Vilares, J. Graña, and P. Alvariño. Finite-state morphology and formal verification. *Journal of Natural Language Engineering, special issue on Extended Finite State Models of Language*, 3(4):303–304, 1997.

An Analysis of Crowdsourcing Relevance Assessments in Spanish

Omar Alonso[1] and Ricardo Baeza-Yates[2]

[1] Microsoft Corp.
1065 La Avenida, Mountain View, CA 94043
omar.alonso@microsoft.com
[2] Yahoo! Research Barcelona
Avinguda Diagonal 177, 8th Floor 08018 Barcelona
rbaeza@acm.org

Abstract. Recently Amazon Mechanical Turk has emerged as a viable platform for conducting relevance assessments. It is possible to conduct experiments extremely fast, with good results and at a very low cost. However, most of the previous work on crowdsourcing has been done in English. In this paper we present the results of a series of experiments using the Spanish part of CLEF, demonstrating that crowdsourcing platforms do work for other languages than English. Our findings reinforce the wisdom of the crowd for this kind of task and we show our methodology at work with several examples.

1 Introduction

In the world of the Web 2.0 and user generated content, one important sub-class of peer collaborative production is the phenomenon known as *crowdsourcing* [6, 9]. In crowdsourcing potentially large jobs are broken into many small tasks that are then outsourced directly to individual workers via public solicitation. One of the best examples is Wikipedia, where each entry or part of an entry could be considered as a task being solicited. However, successful examples of volunteer crowdsourcing are difficult to replicate and hence a financial compensation is used, usually as micro-payments of the order of a few cents per task. This is the model of the Amazon Mechanical Turk (AMT), where many tasks can be done quickly and cheaply.

Later we detail more AMT, but it is quite simple. Workers choose from a list of jobs being offered, where the reward being offered per task and the number of tasks available for that request are indicated. Workers can click on a link to view a brief description or a preview of each task. After seeing the preview, workers can choose to accept the task, where optionally, a qualification exam bust be passed to assign officially the task to them. Tasks are very diverse in size and nature, requiring from seconds to minutes to complete. On the other hand, the typical compensation ranges from one cent to less than a dollar per task and is usually correlated to the task complexity. In the case of information retrieval, crowdsourcing is perfect for relevance assessments as tasks are already small, so

they do not need to be divided in smaller tasks. For this reason crowdsourcing has been used successfully for *relevance assessments*. For example, Alonso & Mizzaro [1] compared a single topic to TREC and found that workers were as good as the original assessors, and in some cases they were able to detect errors in the golden set. A similar work by Alonso et al. [2] in the context of INEX with a larger data set shows similar results. Kazai et al. [7] propose the collective relevance assessment method for gathering relevance assessments for collections of digital books and videos.

Nevertheless, all the mentioned research have been done in English, as the restrictions of being a registered user in AMT imply that most turkers are American. So a natural question is: can we use crowdsourcing for other languages? According to the last USA census in 2006, Spanish is the second most spoken language. In fact, in 2006 there were 34 million people in the USA for which Spanish was their main language, that is, a 12% of the total population (if we consider Spanish as a second language the number grows to over 40 million). So the natural choice to answer the question above is Spanish to see if there are enough assessors that know a language different from English.

This question is also important considering that most linguistic or retrieval oriented golden sets are in English. Hence, would be very interesting if crowdsourcing could help in generating more resources in other languages. To evaluate the quality of crowdsourcing in Spanish we use one of the few golden sets across languages: CLEF (Cross Language Evaluation Forum), using just the Spanish sub-collection. As a side product, our experiments are also one example of how crowdsourcing should be used.

We found that it is possible to run crowdsourcing experiments in other languages than English on Mechanical Turk. Our largest experiment shows that CLEF experts and workers agree on 70% of the answers, which is very promising. We also found that the quality of the justifications is very high and it can be used as a mechanism to detect workers who are fluent in a particular language. In certain cases, workers were able to identify errors in the golden set and by looking at the justifications we can easily identify the disagreements.

This paper is organized as follows. First, we describe Amazon Mechanical Turk in Section 2. Second we describe our experiments and their results in Section 3. We end with some final remarks in Section 4.

2 Amazon Mechanical Turk

Crowdsourcing has emerged as a feasible alternative for relevance evaluation because it combines the flexibility of the editorial approach at a larger scale. Crowdsourcing is a term used to describe tasks that are outsourced to a large group of people instead of performed by an employee or contractor.

AMT is an example of a crowdsourcing platform[3]. AMT is an Internet service that gives developers the ability to include human intelligence as a core

[3] www.mturk.com

component of their applications. Developers use a web services API to submit tasks, approve completed tasks, and incorporate the answers into their software applications. To the application, the transaction looks very much like any remote procedure call. The application sends the request, and the service returns the results. People come to the web site looking for tasks and receive payment for their completed work. In addition to the API, there is also the option to interact using a dashboard that includes several useful features for prototyping experiments.

The individual or organization who has work to be performed is known as the *requester*. A person who wants to sign up to perform work is described in the system as a *worker*. The unit of work to be performed is called a HIT (Human Intelligence Task). Each HIT has an associated payment and an allotted completion time; workers can see sample hits, along with the payment and time information, before choosing whether to work on them. It is possible to control the quality of the work by using qualification tests. A qualification test is a set of questions (like a HIT) that the worker must answer to qualify and therefore work on the assignments.

One of the most important aspects of performing evaluations using crowdsourcing is to design the experiment carefully. Asking the right questions in a simple and effective way involves using guidelines for survey and questionnaire design.

3 Experiments

We conducted three main experiments to test our approach (see table 1). Each experiment has specific metadata in Spanish and English so it can be searchable in both languages. The actual HIT consisted on using the same topic, description, and narrative from the original CLEF distribution. Note that we discourage people who are not fluent in Spanish to take the test. We also pay a bonus in experiment three to get good quality justifications. We followed previous design guidelines for running experiments as well as dealing with worker quality [3]. All experiments shared the same metadata (*Relevancia, Evaluación, web, Máquinas de Búsqueda*) plus specific keywords *(Pena de muerte, Japón, Naciones Unidas, Los Juegos Olímpicos).*

ID	Type	# of documents	Collection	Topics
E1	Graded (3 point scale)	22	Wikipedia	145, 150, 152, 153, 157
E2	Binary	18	CLEF	145, 150, 152, 153, 157
E3	Binary	135 (124 + 11 honey pots)	CLEF	145, 150, 152, 153, 157

Table 1. Overview of the 3 experiments.

3.1 CLEF

CLEF is an initiative to evaluate cross-language information retrieval that started in 2000 [5]. The CLEF multilingual document collection contains several topics and for each language a different document database. In our case we work with the Spanish subset of CLEF, which is composed by over 450 thousand news documents of 1994 and 1995 of EFE, a Spanish news agency. The number of topics that we used in our experiments was 5 and for relevance assessments we just need the relevant documents for each topic plus a sample of non-relevant documents.

3.2 Enough People Know Spanish?

Most of the experimentation in Mechanical Turk is done in English. The first question is then, do we have enough workers who know Spanish? To answer that question, we designed a simple experiment (E1) with Wikipedia content given a short topic description. We also misclassified a few pages to make sure that we were able to trap potential spammers. We did not rejected any work at this moment. The goal was to see if there was interest in such experiments.

The results for E1 are as follows. There were 5 unique users for the 69 assignments. Of the total, 23/69 (33%) contain a justification. There were three workers per assignment and the total cost for the experiment was $1.72. Because we created the experiment, we assigned the relevance values and then compared to the average worker. There was a 60% agreement. This short experiment showed us that it was possible to run similar tasks but with CLEF content.

3.3 Quality of Assessments

The second experiment (E2) used a subset of the Spanish CLEF and the results are as follows. There were 5 unique users for 59 assignments. We did perform spam detection but did not rejected any work. The total cost for the experiment was $1.99. There was a very high agreement (90%) but only 3/59 (5%) of the assignments contained a justification.

For the rest of CLEF, the third experiment (E3), we used honey pots and spam detection. To increase the number of good quality feedback, we paid a $0.01 bonus if we found that the justification was useful. In this case there were 20 unique workers for 273 assignments. The total cost for the experiment was $14.28. The agreement was strong (70%) and 241/273 (88%) of the assignments contained a justification. We did reject 245 assignments that we considered were not good answers by workers. Figure 1 shows the worker activity for E3.

By honey pot we mean the inclusion of a document and relevance assessment that we know the answer upfront and we expect the worker to know it as well. If a worker misses the answer, then there is a possibility that the person was gaming the system or not paying attention. The idea is to include a number of honey pots in the experiment and use that information to detect potential spammers. In our case, if a worker misses more than 2 easy honey pots, then is a

candidate for exclusion from the results (and also the worker doesn't get paid). We also examine the speed of a worker for answering the question. If the time to complete an assignment looks similar across all completed work, then looks like a potential spammer.

We also found errors in the golden set. There were five documents where the majority of the workers (at least two out of three) disagree with the experts on the relevance assessment. Table 4 shows the document identifiers and examples of the justifications by the workers.

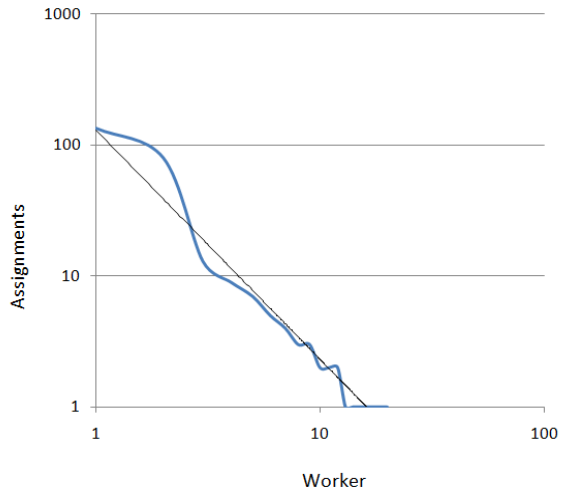


Fig. 1. Assignments by workers.

3.4 Justification Analysis

One way to assess the knowledge of Spanish is by analyzing the justifications. As we can see in Table 2 the explanations are reasonable well written for the first experiment. Notice that some people used uppercase letters, perhaps to avoid the need to use accents, although even in this case some people included the accents. On the other hand accents could be an issue if the assessor uses an English keyboard. It is worth mentioning that in some cases workers did assess a document and provided the justification in English as we can see on the last entry of Table 4. It is unclear if this particular example was done using a translation tool, but at least the worker was following instructions.

Interestingly enough, the length of the justifications increased in the second experiment (the average length went from 55.73 to 201 characters). However, the overall number of justifications in E2 decreased. After introducing the bonus

payment, the number of justifications went up (compared to E1 and E2). In terms of the length of the explanations, the third experiment was 139.34 characters on average. This is remarkable considering the small value of the bonus ($0.01).

Query	**Answer**	**Justification**
Shoemaker-Levy y Júpiter	Yes	Se refiere al cometa precitado y a su impacto en el planeta Júpiter, y sus consecuencias.
java	Somewhat	Habla sobre la versión de código abierto de dicho programa.
fb barcelona	No	Trata acerca de la ciudad, mas no del club de fútbol.

Table 2. Sample of justification answers by workers on the first experiment (E1).

Query	**Answer**	**Justification**
los derechos de la infancia	No	No es relevante porque mas bien es un articulo informativo sobre una reunion y la agenda que se va a revisar en dicha reunion y no hay datos referidos a la Convención de las Naciones Unidas sobre los derechos de la infancia.
AI contra la pena de muerte	Yes	Se lee claramente que AI esta en contra de la pena de muerte y que ha tomado acciones para que el gobierno jordano tome consciencia en cuanto a la sentencia y ejecución de la misma.
Importaciones de arroz en Japon	No	Habla de los recursos que el gobierno invertira para sacar al país de la recesión y sólo hace mención breve sobre los agricultores de arroz que han sido afectados por la importación de arroz.

Table 3. Sample of justification answers by workers on the second experiment (E2).

4 Summary and Conclusions

The results of our research are promising. We cannot tell for sure if all workers are Spanish-speaking or if they were using a translation tool and based their decisions on that. That said, the justification data is a very useful tool to identify workers who are fluent in that language. In addition, the overall quality of the justification/feedback is extremely high, and most probably they are fluent in Spanish, as justifications would have had lesser quality if a translation software was being used.

The initial results show that it is possible to assess multilingual content using a crowdsourcing platform, at least in the English-Spanish case. This is consistent with previous research and we are showing that this can be extended to non-English relevance tasks. Not only the results show an agreement with the CLEF

Query	Answer	Justification
Los juegos olimpicos y la paz	Yes	Yo creo que es desde ya muy relevante, ya que explica como cambiando una fecha de un partido se pudo mantener la paz y evitar conflictos. La fecha se cambio para que Alemania no jugara el dia del nacimiento de Hitler
Importaciones de arroz en Japón	No	El documento no contiene nada referido al tema que es importaciones de arroz en Japón, todo el documento esta referido a la Copa del Rey de España, es decir trata sobre resultados futbolísticos.
Ganadoras de Wimbledon	No	El documento no es relevante al tema que son las ganadoras del torneo de tenis en Wimblewdon, por el contrario el documento trata sobre lafinal que jugaran en el Roland Garros la alemana Steffi Graf y a la española Arantxa Sánchez Vicario.

Table 4. Sample of justification answers by workers on the third experiment (E3).

Topic	DocId	Justification
145	EFE19940222-12530	No es relevante ya que se refiere a un libro sobre economia y la economia entre EEUU y Japon y no se trata explicitamente el tema sobre las importaciones de arroz en Japon.
145	EFE19940315-09328	No, sector del petraleo en este caso.
145	EFE19940322-14353	Este documento no es relevande debido a que no menciona informacion sobre las importaciones de arroz en Japon en ninguno de sus parrafos. Es un documento que habla sobre el acero.
152	EFE19940202-01089	Documento no relevante por cuanto no proporciona informacion sobre la convencion de la ONU acerca de los derechos de la infancia. El documento trata sobre los nuevos derechos de los niños en Francia.

Table 5. Examples of disagreements between experts (CLEF) and workers.

experts, but also show examples when there is a disagreement. This finding was only possible because of the high quality of the feedback.

Compared to previous experience running English relevance experiments, these tests were slower. For a similar experiment in English that usually takes 24hrs to complete, the Spanish version finalizes in 5 days. One potential reason is that there are less Spanish workers than in English. Another reason is the current limitation on Euro payments from Amazon that may impact potential workers. Based on this, the percentage of Spanish-speaking turkers is less than 20% and more careful experiments may show that the actual percentage is consistent with the real percentage (today could be larger than 15%).

5 Acknowledgments

We thank Cristina González for helping us with the CLEF data set.

References

1. O. Alonso and S. Mizzaro. Can we get rid of TREC Assessors? Using Mechanical Turk for Relevance Assessment. *SIGIR Workshop on the Future of IR Evaluation*, 2009.
2. O. Alonso, R. Schenkel and M. Theobald. Crowdsourcing Assessments for XML Ranked Retrieval, *ECIR* 2010.
3. O. Alonso. Crowdsourcing for Relevance Evaluation (Tutorial Notes), *ECIR* 2010.
4. N. Bradburn, S. Sudman and B. Wansink. *Asking Questions: The Definitive Guide to Questionnaire Design*, Josey-Bass, 2004.
5. M. Braschler and C. Peters. The CLEF Campaigns: Evaluation of Cross-Language Information Retrieval Systems, *UPGRADE*, Vol. III, Issue no. 3, 2002.
6. J. Howe. *Crowdsourcing: Why the Power of the Crowd Is Driving the Future of Business*. Crown Business, New York, 2008.
7. G. Kazai, N. Milic-Frayling and J. Costello Towards Methods for the Collective Gathering and Quality Control of Relevance Assessments *SIGIR* 2009.
8. A. Kittur, E. Chi and B. Suh Crowdsourcing user studies with Mechanical Turk, *SIGCHI* 2008.
9. T. Malone, R. Laubacher, and C. Dellarocas *Harnessing Crowds: Mapping the Genome of Collective Intelligence*. MIT Press, 2009.
10. R. Mason, R. Gunst, and J. Hess *Statistical Design and Analysis of Experiments: With Applications to Engineering and Science*. John Wiley & Sons, Inc. 2003.
11. R. Snow, B. O'Connor, D. Jurafsky, A. Y. Ng. Cheap and Fast - But is it Good? Evaluating Non-Expert Annotations for Natural Language Tasks. *EMNLP* 2008.
12. W. Mason and D. Watts. Financial Incentives and the 'Performance of Crowds', *HCOMP Workshop at KDD*, 2009.
13. O. Nov, M. Naaman and C. Ye. What drives content tagging: the case of photos on Flickr. *SIGCHI*, 2008

Scientific information retrieval behavior: A case study in students of Philosophy

Jesús Tramullas, Ana Sánchez Casabón
Deparment of Information Sciences, University of Zaragoza
Ciudad Universitaria s/n 50009, Zaragoza (Spain)
{tramullas, asanchez}@unizar.es

Abstract. The behavior and patterns of recovery and processing of digital information by users is a recurring theme in the literature. The study of these behaviors are carried out through observation techniques and analysis of processes, actions and decisions undertaken by users in different situations. This paper presents the data resulting from the study of patterns of recovery and management of reference information of three consecutive courses of a specialized subject. The findings obtained showed a clear difference between patterns of information retrieval and obtained prior to the end of the training process, but there has been a significant change in the ultimate goal of users or appreciable changes in their prospects for application in other environments

Keywords: Information seeking behavior, information user studies, information literacy

1 Studies into information behaviour

The analysis and study of users' information behaviours is a classic theme which has been widely covered by investigation into information and documentation sciences and is reflected in the corresponding scientific publications. These studies have been covered with different methodological approaches [1] over a 50-year period, and have been extensively dealt with by Wilson [2], [3]. An international conference has been consolidated which is specifically dedicated to such studies, this being ISIC (*Information Seeking in Context*). Moreover, numerous specialised scientific journals have published studies into this matter.

One result of this evolution has been the increasing importance of qualitative study approaches. Most studies combine observation methodologies and the acquisition of quantitative data with qualitative group and individual assessment methods. Some examples of this kind of studies into information seeking behaviours analyses among university students are those of Zhang, Angehelescu and Yuan [4] and Heiström [5]. These works conclude that the level of command of a theme conditions the users' search behaviours rather than their skill to search, and that the way to approach study processes has an impact on the way in which users develop search processes, respectively. Junni [6] stressed that the amount of scientific bibliography used in

master's degree theses has increased with the growing use of digital information resources, but not the number of specialised sources consulted. Nicholas, et al [7] analysed the use of digital journals among the academic community, mainly from a quantitative perspective. If behaviour is limited to searching and discovering books, Rowlands and Nicholas [8] determined a predictive model for the university community according to the gender, discipline and academic status factors.

2 Study approach

The studies cited in the previous paragraph draw various conclusions about search patterns and use of information in different university contexts. Nonetheless, the strong contextual influence underlying them all means we have to accept their conclusions in the context they were done in. According to this principle, an analysis has been proposed in a Spanish university setting to identify any patterns and behaviours shared with other studies as far as possible or, conversely, to identify any typical contextual characteristics which may affect the environment they have been carried out in.

The curriculum of the Philosophy Degree taught at the University of Zaragoza contemplates training students in basic retrieval, processing and usage techniques to be used with scientific information. For such purposes, an optional course subject is taught with six credits known as *Information and Documentation for Humanities*. The Department of Information Sciences is responsible for teaching this optional course subject, which has been taught continuously between 2007-2010. The results of the learning established for this course subject are the following:

- Knowing the principles and foundations of scientific information and documentation processes.
- Developing scientific information search, selection and management processes.
- Include scientific documentation in all academic activities.

In order to accomplish these objectives, a series of learning activities are carried out which are included in the course subject's academic guidelines. The evaluation of these objectives and the determination of the level reached by students are performed by means of a personal scientific information retrieval and processing project.

Teaching this course subject in the aforementioned period has enabled us to not only determine the initial information retrieval patterns of a group of university students in relation to both reasoning and interfaces interaction processes, and their capacity to act from using these patterns, but to also obtain results in terms of the effectiveness of improvement actions. The following data acquisition techniques were used:

1. Direct observation of students' behaviours.

2. Personalised interviews and questionnaires about the activities carried out and the results.
3. Reports about the development and execution of personal projects.

3 Initial situation and analysis

Analysing the responses and observing the initial learning activities have enabled us to establish a low level of knowledge about the structure, organisation and usage of the digital information resources that are considered basic. The results are presented in Table 1.

Table 1. Preliminary use of information resources.

	Google	Advanced Google	OPAC	Reference Databases	Reposito ries	Subject Gateways
2007-08	12	1	2	0	0	0
2008-09	15	2	3	0	0	0
2009-10	14	4	4	0	0	0

The main source of information for students was the generic search engine, Google. Nonetheless, of the 47 students, only 7 reported a limited use of the advanced search interface and that they had not employed the thematic possibilities offered by Google (Table 2).

Tablae 2. Use of Google Services.

	Basic	Avanced	Scholar	Books	Blogs	Video	Image	News	Alerts
2007-08	12	1	0	0	0	0	1	0	0
2008-09	15	2	0	0	0	0	0	0	0
2009-10	14	4	0	1	0	1	2	0	0

This lack of knowledge about the advanced information retrieval services offered by Google was also accompanied by a mistaken personal perception of the students' own knowledge. At the beginning of the subject, the students were questioned about the level they believe they had of the knowledge and the use of the possibilities that this search engine offers. Of all the students, 86% stated that their level of competence was high or very high. This information was then verified with a series of 6 searches which entailed using different formulations of equations involving the selection of search terms, the application of several operators in the equations and the use of constraints. The results obtained reveal that the students' level of success did not reach 30%, and they showed they had no knowledge most of the time about the operators and options this search engine offers as they frequently employed a series of

terms or a given sentence as a search expression to select one option or another intuitively.

Their use and interpretation of the search results coincided with their shortcomings during the interrogation processes. Only 33% of the students were able to correctly identify the different elements shown in the list of hits and to use them properly. Along the same lines, only 40% of these students went on to a second page of hits, and only 3% went on to a third page if they thought it necessary. The navigation processes they started to access original documents with were performed in the same window or tab where the search results were displayed and, on most occasions, this was the reason why students seemed disoriented, leading them to recommence or abandon their search processes. They simply made a decision in virtue of the higher or lesser extent they needed the information they were searching or depending on their weariness of obtaining poor results. As Table 1 shows, when students did not obtain relevant results with Google, they used no other information sources because they did not know they existed.

Their use of library catalogues was rather limited and clearly insufficient. We noticed they only accessed the University of Zaragoza Library OPAC, and OPAC was only used to check the availability of material to loan, and never as a scientific information tool. However, it is logical that they do not use this tool very much because their knowledge of specialised resources is limited. Therefore it is impossible to assess the users' application and satisfaction with these tools in this first stage, at least in terms of them employing such resources as an information retrieval tool.

In relation to this perception, the students have a vague notion of the quality information concept. Although this concept has been the object of theoretical and practical formulations in highly structured information settings, studying it proves much more complex in non homogeneous and poorly structured information settings. The students did not show concern for the quality of the information, except for one of its basic aspects: reliability. The review and assessment of the information quality criteria were carried out as a work group, and the results enabled us to state that the students almost totally lack the logic and basic criteria which may be applied constantly and homogeneously.

4 Developing the teaching-learning process

The subject that was used as a base for this study addresses the informational literacy of the Philosophy Degree students. Its objectives are to improve the information retrieval, processing and assessment processes in general, and skills in searching, selecting and incorporating scientific information and documentation in academic work in particular. In accordance with this, the course subject established learning activities to:

1. Develop advanced information retrieval processes.
2. Identify and make good use of the information retrieval interfaces.

3. Know the main scientific information and documentation resources for their discipline.
4. Use optimised information retrieval processes.
5. Include the scientific information obtained in other products or services.

The intention of combining theory and practice is, on the one hand, for students to gain a clear vision of the structure and content of the information resources they employ and, on the other hand, to learn the technical characteristics of both the interface and the retrieval language of all these resources. The expression of learning is carried out by developing a personal information retrieval and inclusion project about a specific theme of the degree the students were studying. The aim of the project is to create a personal bibliographical information resource by means of a bibliographic management platform which is fed with the information covered by the resources used in the learning process.

This project also entails having to write a technical report which specifies the information sources or resources used, the information search processes carried out, the problems considered in the interaction process, the qualitative assessment of the validity and usefulness of the information retrieved, and the problems encountered to include or re-use the information in other products or services. The information contained in the reports which was collected in the three academic years (2007-2010) have enabled us to extract a series of results which are dealt with in the next section.

5 Results of the project assesment

The information collected in the review of the results obtained in the projects and the corresponding reports has enabled us to state that the information search behaviours conducted by students have constructively progressed. We stress that this progress is limited by the contextual factors relating to former training, the students' perception of the subject, and all the students' personal objectives.

The use of information resources has led to a remarkable change as most students started to use specialised resources, as Table 3 depicts. The changes are significant if compared with the data presented in Table 1.

Table 3. Use of information resources.

	Google Scholar	OPACs	Scientific Search eng.	Reference DB	Repositories	E-Journals	Subject Gatew.	Bibliogr. servers
2007-08	12	10	12	11	10	12	6	2
2008-09	15	11	15	15	10	13	9	7
2009-10	14	12	14	14	12	14	7	4

The training students received has enabled them to carry out information retrieval processes in a higher number of resources, particularly bibliographic and referential

databases, as well as in repositories. In principle, this greater use could indicate an improvement in the scientific information assessment and retrieval processes. Nevertheless, when we break down the data presented in Table 3 and bear in mind other factors, we find factors that are external to the process and limit their success.

Firstly, the languages used are a barrier for the students. A large number of students (56%) only used resources in Spanish and, consequently, they only included Spanish references in their bibliographies. When questioned about this, many students mentioned that this limit was owing to them not knowing other languages. This language-based restriction also affects the resources used themselves as those students with language problems searched and selected scientific information almost exclusively in OPACs and referential databases whose content was in Spanish.

Secondly, the theme and object of the study and, consequently, the information search, lack definition. Students lack specific training in the methodology to perform scientific works. This deficiency makes a suitable formulation of the informative need difficult, and leads to an insufficient capacity to select the terms that could better represent it. This consideration is reflected in the set of terms used in search equations. The reports reveal that the students define an average of between four and six terms which they believe may cover their information needs (Figure 1). Then they use them systematically in search interfaces in the search field of all the fields, if available, or in the title and/or descriptor fields.

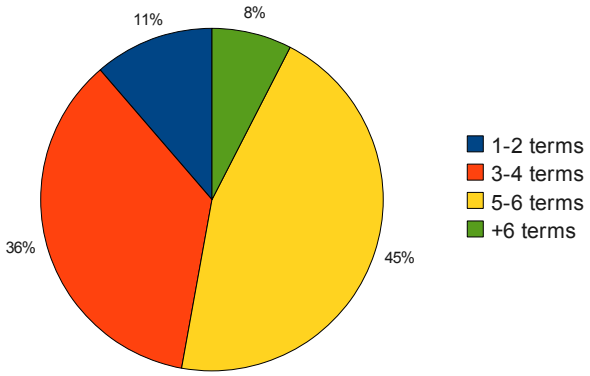


Fig. 1. Number of terms used in retrievals

When the students were asked about modifying the terms according to the relevance of the hits obtained, 64% reported they do not modify the terms if they do not obtain satisfactory hits because they think that the hits do not exist, whereas the remaining 36% modify terms if they consider the hits inadequate. The operator that students used in the equations was the AND-OR logic product, followed by the exact phrase operator. They barely use the remaining operators, and they do not consider the time limit operators given the longer durability of scientific publications on the matter.

Finally, we may also consider several matters from the reports regarding information retrieval and processing whose valuation by the students is totally qualitative:

- Information retrieval interfaces in specialised resources are complex and not very clear, and a model based on a simple interface like Google is preferred.
- The quality of the contents of the resources used is most irregular.
- The services offered by electronic journal suppliers are preferred.
- The contents offered by the OPACs are insufficient compared with other resources.
- Improved interoperability between resources would be desirable to automatically re-use contents.
- Repeating the contents in electronic journals, referential databases and repositories leads to confusion.

6 Conclusions

Wilson [2] brought attention to the relative value of informational behaviour studies which were carried out with small or specialised groups of users, defending the need for studies in broader fields so that a more extensive outlook is available to make an assessment. However, we consider that this approach cannot exclude that used in works like the present one because information retrieval has a strong contextual component which affects users' behaviours. Generalisation tends to avoid the fact that retrieval processes are eminently personal and individual. As O'Brien and Symons pointed out [9], the course level achieved and the academic discipline covered influence users' decisions, although it has been detected that the Web and classmates have a greater influence.

Two kinds of conclusions may be drawn from the research carried out: the first relate to the individual processes carried out by students, while the second correspond to the group's social and educational context. As for the former kind, the following is stressed:

1. Students simplify retrieval processes in terms of both time and resources.
2. Simple interfaces with a few visual or interaction-type elements are preferred.
3. Terms for the search expressions are selected in the initial stage of the process, and the trend is not to modify the selection in subsequent situations.
4. Search processes do not tend to be extended by navigating or exploring.
5. The whole process is like a linear episode rather than a recurrent one, and reconsidering search actions is rare.

Second searches, that is, those relating to the context that are mainly of a social and education kind, particularly influence users' behaviours:

1. The confusion between digital literacy ad informational literacy in the educational context means students are efficient when it comes to using tools, but inadequate when considering processes and assessing them.
2. Lack of training in scientific work processes has an impact on their lack of knowledge about them existing, their use and the possibilities that specialised resources offer.
3. The training for users that university libraries provide does not fulfil the objectives and results required.
4. The interest and development of social networks have led students to consider them and use them as a leading platform to make the most of Internet resources..

Referencias

1. Fisher, K., Erdelez, S. y Lynnc, E.F. (eds.) Theories of Information Behavior.. Medford, NJ: Information Today for the American Society of Information Science and Technology, 2005. (ASIST Monograph Series)
2. Wilson, T.D. Information needs and uses: Fifty years of progress. En B.C. Vickery, (ed.). Fifty years of information progress: A Journal of Documentation review, pp. 15- 51. London: Aslib (1994)
3. Wilson. T.D. Fifty years of information behavior research. Bulletin of ASIST, February/March 2010, http://www.asis.org/Bulletin/Feb-10/FebMar10_Wilson.html (2010).
4. Zhang, X., Anghelescu, H.G.B., y Yuan, X. Domain knowledge, search behavior, and search effectiveness of engineering and science students: An exploratory study. Information Research, **10**(2), paper 217, http://informationr.net/ir/10-2/paper217.html (2005)
5. Heinström, J. Fast surfing for availability or deep diving into quality – motivation and information seeking among middle and high school students. Information Research, **11**(4), paper 265. http://informationr.net/ir/11-4/paper265.html (2006)
6. Junni, P. Students seeking information for their Masters' theses: the effect of the Internet. Information Research, **12**(2) paper 305. http://InformationR.net/ir/12-2/paper305.html (2007)
7. Nicholas, D., Huntington, P. Jamali, H.R. Y Watkinson, A. The information seeking behaviour of the users of digital scholarly journals, Information Processing & Management, **42**(5), pp. 1345-1365 (2006)
8. Rowlands, I. y Nicholas, D. Understanding Information Behaviour: How Do Students and Faculty Find Books?, The Journal of Academic Librarianship, **34**(1), pp. 3-15 (2008).
9. O'Brien, H.L y Symons, S. The information behaviors and preferences of undergraduate students, Research Strategies, **20**(4), pp. 409-423 (2005).

On the Fly Query Segmentation Using Snippets

David J. Brenes[1], Daniel Gayo-Avello[2] and Rodrigo Garcia[3]

[1] Simplelogica S.L. david.brenes@simplelogica.net
[2] University of Oviedo dani@uniovi.es
[3] University of Oviedo rodrigo@innova.uniovi.es

Abstract. One of the most important issues in Information Retrieval is inferring the intents underlying users' queries, and query segmentantion, provided it is done fast, can be a valuable tool. Such techniques usually rely on a prior training phase involving large datasets. A costly process, specially in environments which are increasingly moving towards real time scenarios where latency to retrieve fresh informacion should be minimal. In this paper an 'on-the-fly' query segmentation method is proposed. It uses snippets which are later mined by means of a naive statistical algorithm. An initial evaluation of such a method is provided, in addition to a discussion on its applicability to different scenarios.

1 Introduction

One of the main purposes of Web Information Retrieval is helping users to fulfill their information needs. However, most of the time, the main problem is determining the users' intent. This is a complex task that can be improved by using a number of different techniques.

One useful approach is extracting MWUs[4] from the queries sent by the user. Unfortunately, query segmentation tend to be performed by means of statistical or machine learning methods which rely on a prior large corpus of training data which is processed in advance during the training phase because of the unfeasibility of doing it in real time when the queries are submitted.

Hence, performance is achieved at the cost of being unable to react to new types of queries. This can affect a considerable amount of users because a significant number of queries are submitted just once to the information retrieval system ([6], [1], [7]). More importantly, this issue is critical in an ecosystem driving towards 'real time' where users demand fresh content faster than ever.

In this paper the authors propose and evaluate a new query segmentation approach designed to be as light and fast as possible; avoiding costly preprocessing or large datasets. The main goal is being able to create a system which can adapt to real time changes by just analyzing the pertinent information for the current query.

[4] Multi-Word Units: a relevant group of words

2 Previous Research

Nonetheless to say, noun phrase segmentation is not a novel area and large research efforts have been applied to this question ([5], [8], [3] or [2]).

Concerning Web Information Retrieval, noun phrase segmentation has been applied to query segmentation in the same way it is applied to longer documents; that is, without taking into account the special features and requirements of search queries: shorter, much more schematic, with a very different vocabulary set.

3 Proposal Description

3.1 Characteristics Overview

A new Web enviroment is moving towards an ecosystem of real time applications where the context changes very quickly as the news spread. This leads to the main requirement of our approach: to offer good results not only for unexpected queries, but also for queries related to drifting topics of interest.

We are not saying, by no means, that methods relying on a prior training are not fast or feasible for this task. They are extremely fast (once trained) and they have been applied to query segmentation (e.g. [8]). It is the speed of topic drifting and how to handle that changing trends which we feel can't be properly achieved by using methods relying on training data and heavy result caching. Because of this, our approach consists of performing query segmentation on-the-fly. Of course, a balance should be reached as the results must be provided in a reasonable amount of time and, in fact, repeated queries could and should be cached in order to avoid duplicate computations.

Authors are also conscious that returning fresh results demands a light algorithm which may not achieve as good performance as previous approaches based on training data. However, we feel that a certain lack of precision could be tolerable in exchange for greater adaptability to drifting and evolving topics.

3.2 Snippet Sources

Our method exploits snippets which are short extracts of web documents usually containing one or more keywords from the user query. They are commonly seen in result pages of search engines, but we are open to also consider other sources. However, for the purpose of this paper, we have only used snippets from three major search engines (Google[5], Bing[6] and Yahoo![7]) exploiting their respective APIs. This way, we have been able to compare the performance of our method using different sources of snippets.

[5] http://code.google.com/intl/es-ES/apis/ajaxsearch/documentation/#fonje
[6] http://msdn.microsoft.com/en-us/library/dd251056.aspx
[7] http://developer.yahoo.com/search/boss/

3.3 Statistics Measures

In addition to our own method, we tested several statistical measures previously employed in MWU detection; namely, Mutual Information, SCP, Phi, Dice and LogLike. [4] devised a way to generalize those well-known measures from bigrams to n-grams of arbitrary length and, hence, we have used those generalized versions.

As we have already said, we have devised another method based on the use of snippets. The underlying idea is really simple: First, all possible n-grams in the query are produced. Then, the frequency for each n-gram in the snippets is to be found and those n-grams below a certain threshold are discarded. Finally, the query is segmented by using the most frequent remaining n-grams. The aforementioned threshold limits the minimum number of snippets a n-gram should appear in (for instance, a 0.5 threshold requires the n-gram to appear in half or more of the snippets).

We tested four different values for this parameter (namely, 0.25, 0.5, 0.75 and 1) so we can study how it affects to the performance of our method. However, because of space costraints only results for 0.25 and 1 values are shown.

4 Evaluation

4.1 Datasets

In order to evaluate the different query segmentation methods we needed segmented queries. Such queries could have been segmented by experts (e.g. the authors themselves or other scholars) or, much better, by users of a search engine. In the first case, one could assume a consistent set of segmentations while in the second one the data could be noisier but also larger and closer to the actual use scenario.

We were able to use one dataset of each kind. For the expert-segmented data we relied on a subset of the AOL query log [9] manually segmented by [3]; we will refer to this dataset as aol-bergsma-wang-2007 from now on. The user-segmented data was collected by the authors of this study using the SearchSpy feature from the Dogpile metasearch engine. This second dataset (searchspy-2010 from now on) just comprises queries with paired double quotes enclosing two or more terms.

4.2 Evaluation Methodology

At a first glance, one could think of using the percentage of correctly segmented queries to measure the effectiveness of the different methods. However, this approach does not seem to be the best choice because (1) we can think of different degrees of segmentation accuracy (`[new york] [travel guides]` better than `[new york] travel guides` better than `new [york travel] guides`) and (2) segmentation is a user-dependent task and two different users –even experts, as shown by [3]– can provide different segmentations.

Thus, we chose to use the well-known measures of precision and recall as used in other segmentation tasks; that is, taking into account not the queries but the blanks between segments both in the reference and the segmentation provided by each algorithm. For instance, in the query `new york travel guides` and assuming the correct segmentation is `[new york] [travel guides]`, the segmentation `[new york] travel guides` have a precision of 1 and a recall of 0.5, while segmentation `new [york travel] guides` have both 0 precision and 0 recall.

4.3 Positive Bias from Snippets Sources

We run the algorithms in three different 'flavors'. Each flavor rearranged the queries in a different way before submitting it to the search engine to obtain the snippets: (1) removing all the double-quotes; (2) sending the original queries; and (3) removing the double-quotes and reversing the order of the terms.

This way, we have (1) a base case in which queries are submitted with no segmentation information but keeping keyword ordering; (2) a best case where all the segmentations within the query are preserved; and (3) a worst case where neither segmentations nor keyword ordering are preserved.

By doing this it could be possible to appreciate, albeit somewhat indirectly, the impact term collocation could exert on the results and, consequently, on the snippets and the different methods performance. Thus, if term collocations were heavily used by search engines to produce results then the differences between flavors 1 and 2 should be minimal. Additionally, by reversing the order of the terms in the query all valid collocations are removed and spurious ones are introduced; this should make much harder to find valid segmentations. If even under this hard circumstances the methods manage to find correct segmentations we could safely discard the hypothesis of the search engines performing query segmentation and, thus, we could argue that the proposed methods are actually performing query segmentation by just relying on short text snippets.

5 Experiment Results

5.1 Performance on *searchspy-10* Query Log

Table 1 shows the P, R, and F measures for each statistic measure and each snippet source used in the experiments.

When Yahoo! Boss or Bing are used as snippet sources, F measures vary between 0.6 to 0.77; this is a reasonable performance for a technique that is using relatively little information (10 snippets of text as a maximum) which would make statistical algorithms to work in an unreliable way.

Quite surprisingly, using Google as the snippet source drops F measures to very low values (from 0.48 to 0.61). The main reason for such differences, beyond the quality of the returned results, is the amount of queries from the query log that Google couldn't find snippets for. About 1,100 against the less

Measure	Snippet Source	P	R	F
Mutual Information	Bing	0.6834	0.5481	0.6158
	Boss	0.7464	0.6210	0.6837
	Google	0.5374	0.4349	0.4862
SCP	Bing	0.6327	0.6216	0.6271
	Boss	0.6609	0.6481	0.6545
	Google	0.6145	0.6089	0.6117
Phi	Bing	0.6888	0.5367	0.6128
	Boss	0.7530	0.6061	0.6795
	Google	0.5411	0.4246	0.4829
Dice	Bing	0.6888	0.5354	0.6121
	Boss	0.7519	0.6057	0.6788
	Google	0.5405	0.4241	0.4823
Loglike	Bing	0.7053	0.6383	0.6718
	Boss	0.7372	0.6663	0.7017
	Google	0.5349	0.4715	0.5032
Entity Frequency (25)	Bing	0.7336	0.6446	0.6891
	Boss	0.8089	**0.7375**	**0.7732**
	Google	0.5873	0.5298	0.5585
Entity Frequency (100)	Bing	0.7530	0.5095	0.6312
	Boss	**0.8329**	0.5701	0.7015
	Google	0.6030	0.4084	0.5057
Average	Bing	0.7090	0.5753	0.6422
	Boss	**0.7717**	**0.6401**	**0.7059**
	Google	0.5734	0.4712	0.5223

Table 1. Performance using different statistic measures and snippet sources for queries in *searchspy-10* query log

than 500 for Bing and less than 300 for Yahoo!. However, the proposed system is higly dependant on the snippet source so we feel it was important to measure performance using the whole query log, 'failed' queries included.

Concerning the statistical methods, *Entity Frequency* –the simplest one– consistently achieves the best performance (according to F-measure); outperforming *Loglike*, the second best algorithm, by 10.19%.

5.2 Performance on *aol-bergsma-wang-2007* Query Log

In table 2 we can see the results of our method when running on *aol-bergsma-wang-2007* query log.

For this query log, F measures achieves better performance (between 0.07 for Boss and 0.23 for Google). The main reason for this better performance is that very few queries obtained no results in the snippet sources because [3] selected only those queries with at least one clicked result in the AOL query log.

Measure	Snippet Source	P	R	F
Mutual Information	Bing	0.7666	0.7240	0.7453
	Boss	0.7974	0.7557	0.7765
	Google	0.7833	0.7335	0.7584
SCP	Bing	0.7084	0.7507	0.7295
	Boss	0.7234	0.7752	0.7493
	Google	0.6953	0.7512	0.7233
Phi	Bing	0.8159	0.7362	0.7760
	Boss	0.8421	0.7655	0.8038
	Google	0.8269	0.7527	0.7898
Dice	Bing	0.8182	0.7423	0.7802
	Boss	0.8387	0.7644	0.8016
	Google	0.8256	0.7545	0.7901
Loglike	Bing	0.7842	0.8038	0.7940
	Boss	0.8012	**0.8264**	**0.8138**
	Google	0.8053	0.8238	0.8145
Entity Frequency (25)	Bing	0.7183	0.7895	0.7539
	Boss	0.7374	0.8162	0.7768
	Google	0.7150	0.7886	0.7518
Entity Frequency (100)	Bing	0.8427	0.5430	0.6929
	Boss	**0.8733**	0.5590	0.7161
	Google	0.8505	0.5534	0.7019
Average	Bing	0.7848	0.7149	0.7499
	Boss	**0.8069**	**0.7396**	**0.7732**
	Google	0.7901	0.7259	0.7580

Table 2. Performance using different statistic measures and snippet sources for queries in *aol-bergsma-wang-2007* query log

If those queries without results are removed the differences between snippet sources are not significant, but, again, using Yahoo! Boss achieves the better performance.

Concerning the algorihms, *Entity Frequency* behaves better with this query log than with the previous one but a more important boost is the one achieved by other statistic measures such as *Dice*, *Mutual Information* or *Loglike*. In fact, *Loglike* is the algorithm which achieves the highest values for R and F measures, although *Entity Frequency* still shows the highest value for Precision.

5.3 Bias Results

Table 3 shows the variation obtained in F measure for the best and worst cases as described in Section 4.3.

To compute the data in that table, queries with no results were removed because they were different for each search engine and, thus, they introduced an additional source of bias. By removing them, we can compare search engines

to each other and, thus, have a glimpse on the way they can (or cannot) be performing query segmentation by themselves.

Source	*searchspy-10*			*aol-bergsma-wang-2007*		
	Original	Quoted	Reversed	Original	Quoted	Reversed
Bing	0.799	0.889 (+11%)	0.796 (-0.37%)	0.771	0.831 (+7.81%)	0.779 (+1.04%)
Boss	0.808	0.872 (+7.98%)	0.792 (-1.98%)	0.787	0.840 (+6.71%)	0.783 (-0.5%)
Google	0.805	0.879 (+9.22%)	0.793 (-1.54%)	0.780	0.850 (+8.98%)	0.780 (-0.06%)

Table 3. Average F measures for best (quoted), base (original) and worst (reversed) scenarios

When we compare best case diferences against worst cases ones, we find that best cases are very different from the base case. So we can safely assert that (1) our technique is actually performing query segmentation –certainly by means of the snippets– and (2) it is not inadvertently taking advantage of any underlying query segmentation or entity detection phase performed by the search engine.

6 Conclussions

6.1 Summary

Query segmentation can be a useful tool to tackle with the currently emerging real-time scenario for IR systens. Accordingly, we have proposed the use of short snippets to obtain –by means of simple statistical methods– the most relevant MWUs. In addition to that, we performed several experiments on two previously segmented query logs: one by experts and the other by actual Web searchers. Preliminary results are really promising (F-measure is about 0.8) and, thus, further research in this line should be done.

6.2 Snippet Source Dependency

searchspy-10 query log reflects real users' behavior with regards to query segmentation. The main conclusion one can reach from the experiments conducted on this query log is that performance is highly dependent on the underlying snippet source. For example, preliminary work using Wikipedia search engine has shown it can be a very good source of snippets but only for certain kind of queries.

6.3 Statistic algorithm election

Two algorithms outperform the others in both experiments: Loglike and Entity Frequency: Loglike achieves the highest performance for all the experiments,

but its behavior is highly irregular and does not handle very well queries from 'actual' users. Entity Frequency provides much more consistent results for both query logs achieving similar performance. Besides, Entity Frequency different performance values between thresholds is also a variable to take into account.

7 Future Work

We have described a preliminary implementation which achieves good results; nevertheless, some questions remain open and they deserve further research.

- *Real Time Snippets*:
  More work must be done on integrating different snippet sources directly related to real-time Web.
- *Snippet Sources Integration*:
  The use of various complementary sources of snippets could improve the performance.

8 Acknowledgments

This work was partially financed by grant UNOV-09-RENOV-MB-2 from the University of Oviedo

References

[1] Steven M. Beitzel, Eric C. Jensen, David D. Lewis, Abdur Chowdhury, and Ophir Frieder. Automatic classification of Web queries using very large unlabeled query logs. *ACM Transactions on Information Systems (TOIS)*, 25(2), 2007.

[2] Michael Bendersky, W. Bruce Croft, and David a. Smith. Two-stage query segmentation for information retrieval. *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval - SIGIR '09*, page 810, 2009.

[3] Shane Bergsma and Qin Iris Wang. Learning Noun Phrase Query Segmentation. *Computational Linguistics*, (June):819–826, 2007.

[4] Joaquim Ferreira Da Silva and Gabriel Pereira Lopes. A Local Maxima method and a Fair Dispersion Normalization for extracting multi-word units from corpora. *Sixth Meetings on Mathematics of Language*, pages 369–381, 1999.

[5] Jianfeng Gao, Jian-Yun Nie, Guangyuan Wu, and Guihong Cao. Dependence language model for information retrieval. *Annual ACM Conference on Research and Development in Information Retrieval*, 2004.

[6] Bernard J. Jansen, Amanda Spink, Tefko Saracevic, and Judy Bateman. Real life information retrieval: a study of user queries on the Web. *SIGIR Forum*, 32(1):5–17, 1998.

[7] Bernard. J. Jansen, Amanda Spink, Tefko Saracevic, and Dietmar Wolfram. Searching the Web: The Public and Their Queries. *Journal of the American Society for Information Science and Technology*, 52(3):226–34, 2001.

[8] Bin Tan and Fuchun Peng. Unsupervised query segmentation using generative language models and wikipedia. *International World Wide Web Conference*, 2008.

[9] Cayley Torgeson, Abdur Chowdhury, and Greg Pass. A picture of search. page 1, Hong Kong, junio 2006. ACM.

Las herramientas de búsqueda y recuperación de información, con finalidades académicas, utilizadas por el alumnado universitario

Jaume Sureda[1], Rubén Comas[2] y Mercè Morey[3]

Grupo de Investigación Educación y Ciudadanía. Departamento de Pedagogía Aplicada y Psicología de la Educación de la Universidad de las Islas Baleares.
Campus UIB, Edificio Beatriu de Pinós. Cra. De Valldemossa, km. 7,5. 07122, Palma de Mallorca, Islas Baleares
[1] sureda.negre@gmail.com
[2] rubencomas@uib.es
[3] merce.morey@uib.es

Resumen. Con la aparición y desarrollo de las TIC's, el ámbito educativo, y en particular la Universidad, está padeciendo cambios vertiginosos, no sólo en sus estrategias docentes sino en las dinámicas de aprendizaje del alumnado. Un ejemplo de ello es el manejo de la información por parte de los estudiantes pero, fundamentalmente, las herramientas de búsqueda y recuperación de información que utilizan dichos estudiantes. Actualmente, partiendo de la ingente cantidad de información a la que se tiene acceso, ¿cómo busca y recupera dicha información el alumnado universitario? Ésta es una de las cuestiones básicas a las que se ha intentado dar respuesta en un estudio más amplio, desarrollado por el grupo de investigación Educación y Ciudadanía de la Universitat de les Illes Balears, centrado en el análisis de los niveles de alfabetización informacional entre el alumnado de dicha universidad.

Palabras clave. Recuperación de información, educación superior, alfabetización informacional, alumnado universitario.

1 Introducción

Las estrategias de recuperación de información se han convertido en un elemento clave para el alumnado universitario, el cual, además de tener que enfrentarse en un futuro con un entorno laboral muy competitivo en el que la información (la gestión de la información) será crucial, tienen que ser capaces y competentes en su actividad académica, de conocer y manejar a la perfección las estrategias de búsqueda y recuperación de información más efectivas: comprender sus necesidades informacionales, saber localizar la información que precisan, evaluar la información localizada, emplear la información con arreglo a principios éticos y legales, y saber comunicar adecuadamente nueva información.

De esta manera, la sociedad de la información en la que estamos inmersos, así como el actual modelo de Educación Superior, exigen capacitar al alumnado universitario de una serie de competencias que van más allá de la memorización o

conocimiento de unos contenidos concretos. Por ello, desde finales de los años ’70 se viene desarrollando la llamada alfabetización informacional; un concepto que engloba dichas habilidades y que busca profundizar en las estrategias y herramientas de localización, análisis y comunicación de la información.

Aunque no entremos aquí en detalle acerca de las numerosas definiciones que se han elaborado sobre el término alfabetización informacional (por no ser éste el objeto de la presente comunicación) sí que haremos una breve revisión de algunas de las que han sido las más utilizadas, puesto que nos servirán para enmarcar el foco del estudio que presentamos: las técnicas y herramientas de recuperación de información que utiliza el alumnado universitario en sus actividades de índole académica, como elemento clave de la alfabetización informacional.

El concepto *information literacy* se utiliza por primera vez en 1974 refiriéndose al ámbito laboral y a la resolución de problemas (Zurkowkski, 1974 [1]):

> *Las personas entrenadas en el uso de los recursos de la información en su trabajo pueden denominarse alfabetizados informacionalmente. Ellos han aprendido técnicas y habilidades para utilizar una amplia gama de instrumentos de la información así como fuentres primarias para la resolución de problemas con dicha información.*

Dejando de lado el ámbito laboral y refiriéndonos al significado más amplio del término, una de las instituciones más reconocidas en el campo de la alfabetización informacional (la American Library Association) definía así, en 1989, a la persona que poseía las competencias informacionales necesarias:

> *Para ser competente respecto a la información, un individuo debe reconocer cuándo es ésta necesaria, y tener la capacidad para localizar, evaluar y usar de forma efectiva la información que requiere… La gente preparada en este aspecto es, finalmente, la que ha aprendido a aprender. Saben cómo aprender porque saben cómo se organiza la información, cómo encontrarla, y cómo usarla de forma que otros puedan aprender de ellos*. (ALA, 1989 [2])

De esta manera, la Association of College & Research Libraries (ACRL) es una de las instituciones más reconocidas en el campo de la alfabetización informacional y ligada a la ya citada American Library Association, propone una definición más adaptada a la enseñanza superior:

> *El desarrollo de personas que sean capaces de aprender a lo largo de toda su vida es primordial para la misión de las instituciones de educación superior. Asegurándose de que los individuos poseen las capacidades intelectuales del razonamiento y del pensamiento crítico, y ayudándoles a construir un marco para aprender a aprender, las instituciones universitarias ofrecen la base para un crecimiento continuo a lo largo de sus carreras, así como en sus funciones como ciudadanos y miembros de la comunidad bien informados.* (ACRL-ALA, 2000 [3])

De hecho, es la propia ACRL la que presenta una serie de normas (un total de 5), indicadores de rendimiento y resultados que sirven para valorar el nivel de alfabetización informacional en la Educación Superior (ACRL-ALA, 2000). Aunque no entremos en detalle en los indicadores y resultados apuntados, las 5 normas propuestas son:

- Norma 1. El estudiante que es competente en el acceso y uso de la información es capaz de determinar la naturaleza y nivel de la información que necesita.
- Norma 2. El estudiante competente en el acceso y uso de la información accede a la información requerida de manera eficaz y eficiente.
- Norma 3. El estudiante competente en acceso y uso de la información evalúa la información y sus fuentes de forma crítica e incorpora la información seleccionada a su propia base de conocimientos y a su sistema de valores.
- Norma 4. El estudiante competente en el acceso y uso de la información, a título individual o como miembro de un grupo, utiliza la información eficazmente para cumplir un propósito específico.
- Norma 5. El estudiante competente en el acceso y uso de la información comprende muchos de los problemas y cuestiones económicas, legales y sociales que rodean al uso de la información , y accede y utiliza la información de forma ética y legal.

En esta comunicación nos interesa destacar la Norma 2, centrada en las estrategias de recuperación de información como clave para el desarrollo global de las competencias en alfabetización informacional. Para dicha norma, se establecen a su vez 5 indicadores de rendimiento, y un total de 22 resultados. Aún así, no todos los indicadores ni resultados se refieren al sistema de recuperación de información utilizado, por lo que, a continuación se presentarán los datos de los indicadores y resultados que sí están directamente relacionados con la búsqueda y recuperación de información por parte del alumnado universitario. Así, los indicadores y resultados analizados son:

1. Indicador 1. Construye y pone en práctica estrategias de búsqueda diseñadas eficazmente.
    - Resultado 1.1. Identifica palabras clave, sinónimos y términos relacionados para la información que necesita.
    - Resultado 1.2. Construye una estrategia de búsqueda utilizando los comandos apropiados del sistema de recuperación de información elegido.
2. Indicador 2. Obtiene información en línea o en persona gracias a una gran variedad de métodos.
    - Resultado 2.1. Utiliza varios sistemas de búsqueda para recuperar la información en formatos diferentes.
3. Sabe refinar la estrategia de búsqueda si es necesario.
    - Resultado 3.1. Valora la cantidad, calidad y relevancia de los resultados de la búsqueda para poder determinar si habría que utilizar sistemas de recuperación de información o métodos de investigación alternativos.
    - Resultado 3.2. Identifica lagunas en la información recuperada y es capaz de determinar si habría que revisar la estrategia de búsqueda.

2 Metodología

En los meses de octubre y noviembre de 2009 se llevó a cabo parte del trabajo de campo para analizar y evaluar el nivel de competencias informacionales entre el alumnado universitario, concretamente entre los estudiantes matriculados en la Universidad de las Islas Baleares. Así, del total de 11.389 alumnos de dicha universidad, participaron en el estudio 1.025 sujetos, suponiendo de esta manera una muestra estadísticamente representativa por cursos y Facultades. Esta muestra representativa supone un error muestral del ±2,9% para un nivel de confianza del 95%, bajo la condición más desfavorable de p=q=0,05.

Dicho trabajo de campo consistió en la pasación de un cuestionario diseñado a tal fin *("Encuesta sobre las competencias en Alfabetización Informacional del alumnado de la Universidad de las Islas Baleares"*) en el que se incluían un total de 41 ítems para determinar el nivel de alfabetización informacional. Dichos ítems se construyeron en función de las 5 Normas apuntadas por la ACRL-ALA (2000) y que caracterizan a la persona competente en el acceso y uso de la información.

El instrumento elaborado para recoger toda esta información también se basó en otros cuestionarios utilizados anteriormente (Mittermeyer & Quirion, 2003 [4]; SCD-URFIST, 2008 [5]; etc.).

En esta comunicación exponemos los datos relativos a la búsqueda y, fundamentalmente, recuperación de información por parte del alumnado universitario representado en dicha investigación, por lo que sólo entraremos en el análisis de algunos de los ítems incluídos en la encuesta (los que evaluaban los indicadores y resultados pertenecientes a la Norma 2, ya comentada anteriormente), cuyos resultados se exponen a continuación en el siguiente apartado.

3 Resultados

En la Tabla 1 se señala la información relativa a la edad, género, curso, y área de conocimiento de la muestra estudiada.

Tabla 1. Porcentajes relativos a los datos generales de la población analizada.

Variables	Porcentajes
Edad	
Entre 17 y 20 años	35,4%
Entre 21 y 25 años	49,5%
Entre 26 y 30 años	8,4%
Más de 30 años	6,7%
Género	
Hombre	34,5%
Mujer	65,5%
Área de conocimiento	
CC. Sociales y Jurídicas	66,3%
Humanidades	4,5%
CC. Experimentales	16,1%

CC. de la Salud	13,1%
Curso[1]	
Primero	28,8%
Segundo	19,6%
Tercero	25,1%
Cuarto	21,3%
Quinto	5,1%

Una vez analizados los datos, se ha observado que existen diferencias estadísticamente significativas en relación al área de conocimiento a la que pertenece el alumnado encuestado, por lo que, además de destacar las cifras más interesantes obtenidas en el estudio, se hace, a su vez, una breve alusión a este grado de significación (que se completará con la debida nota a pie de página con el valor de la prueba de chi-cuadrado de Pearson –X^2– para la valoración de cada ítem analizado, así como la significación –p– y los grados de libertad –g.l.–).

A continuación se exponen, en primer lugar, los datos referidos al nivel de utilización de Internet para la elaboración de trabajos académicos, de cara a introducir los ya mencionados resultados ligados a los distintos indicadores de rendimiento de la Norma 2 de la ACRL-ALA. Dichos resultados, como también se ha mencionado, han sido analizados a través de distintos ítems del cuestionario, en función de los datos obtenidos y explotados estadísticamente.

A nivel general, un 73,8% del alumnado afirma utilizar con mayor frecuencia Internet para realizar sus trabajos académicos en la Universidad de lo que lo utilizaba estando en Bachillerato; un 19,5% manifiesta realizar un uso de la Red comparable al que hacía en Bachiller, y sólo un 6,6% apunta que en la anterior etapa educativa utilizaba más Internet de lo que lo hace ahora en la Universidad. Si realizamos un análisis por ramas de conocimiento, se observa que existen diferencias estadísticamente significativas entre el uso de Internet en Bachillerato y en la Universidad por área de conocimiento, especialmente en los estudios incluídos en Ciencias Experimentales[2].

Para el Indicador de Rendimiento referido a la puesta en práctica de estrategias de búsqueda diseñadas eficazmente, se ha tomado como ejemplo el nivel de identificación y selección de palabras clave para la puesta en práctica de estrategias de búsqueda y recuperación de información, y se han utilizado los datos obtenidos de requerir al alumnado la selección de palabras clave para referirse a un tema concreto. En este ítem, sólo un 51,6% de los estudiantes escogieron las palabras clave correctas y evitaron términos innecesarios. Son reseñables, nuevamente, las diferencias de

[1] Debe aclararse que, debido a que algunos de los ítems que analizaremos a continuación, cuando se formulaban en el cuestionario, se referían a actuaciones llevadas a cabo durante el anterior curso académico, sólo se utilizará la información aportada por el alumnado que en el momento de la pasación de la encuesta estaba realizando segundo de carrera o un curso superior, dejando de lado las respuestas dadas por los estudiantes de primer curso. Así, en los ítems que se analicen y hagan referencia a una situación hipotética del anterior curso académico, se trabajará con un total de 730 sujetos en lugar de los 1.025 iniciales. De todas formas, dicho dato se aclarará en cada ocasión al presentar los resultados.

[2] X^2=39,425 // p=0,000 // g.l.=6

respuesta en función del área de conocimiento del alumnado, destacando los estudiantes pertenecientes a Ciencias Experimentales a la hora de definir las palabras clave (elemento fundamental a la hora de realizar cualquier búsqueda de información efectiva)[3].

Además, para profundizar aún más en el análisis de las estrategias de búsqueda (y recuperación) de información, también se han tomado como referencia los resultados de utilización de los comandos apropiados del sistema de recuperación elegido, basándonos, concretamente, en los operadores booleanos (imprescindibles a la hora de realizar una búsqueda en una base de datos, por ejemplo). La información analizada nos permite comprobar que, pese a que un 29% afirma conocer dichos operadores, y un 16,4% dice utilizarlos con frecuencia, sólo un 4,1% lo hace correctamente. Una vez más (y en este caso, de manera muy clara), se aprecian diferencias estadísticamente significativas en este ítem en función del área de conocimiento, destacando en el polo positivo, por encima del resto, el alumnado que está cursando estudios pertenecientes a Ciencias de la Salud[4].

Respecto al siguiente Indicador analizado (que hace referencia a la obtención de información en línea o físicamente a través del uso de gran variedad de métodos) los resultados esperados requieren, entre otros, de la utilización de varios sistemas de búsqueda para recuperar la información en formatos distintos. Los datos obtenidos a partir del cuestionario no reflejan esta variedad de métodos, a la vez que muestran una infrautilización de los servicios Bibliotecarios y denotan un posible exceso de uso de la Red ya que, de los alumnos matriculados en 2º, 3º, 4º y 5º, sólo un 29,7% puede ser considerado como usuario habitual[5] de las bibliotecas y únicamente un 14,8% utilizó habitualmente el catálogo online de la biblioteca durante el curso académico 2008-2009, mientras que el porcentaje que usuarios habituales de Internet ascendió al 71,6%. Dicho dato también mantiene las diferencias estadísticamente significativas cuando se refiere al área de conocimiento a la que pertenece el alumnado, resultando destacable la información aportada por el alumnado de Humanidades, que sobresale (muy por encima del resto) en el uso de las bibliotecas y del catálogo electrónico de éstas[6].

Otro resultado obtenido y analizado en relación a este mismo indicador de rendimiento es el que hace referencia a la utilización de servicios especializados para la recuperación de información. Para ello, se han tomado como ejemplo los datos obtenidos acerca del conocimiento que tiene el alumnado sobre bases de datos especializadas en su disciplina de estudio: sólo un 29,1% de los estudiantes afirma conocer dichas bases de datos. Si, a su vez, cruzamos este resultado por área de

[3] X^2=18,020 // p=0,035 // g.l.=9
[4] X^2=208,455 // p=0,000 // g.l.=9
[5] En los ítems (referidos a uso de las bibliotecas en general, del catálogo online de la biblioteca universitaria y de Internet en su sentido más amplio) en los que las posibilidades de respuesta iniciales eran "Ninguna vez", "Menos de 5 veces", "Entre 5 y 10 veces", "Entre 11 y 20 veces", "Entre 21 y 50 veces" y "Más de 50 veces", se ha llevado a cabo una recodificación de dichas posibilidades en "Usuario no habitual" (que incluye las tres primeras opciones) y "Usuario habitual" (que alude a la utilización del servicio o herramienta 11 o más veces)
[6] X^2=56,037 // p=0,000 // g.l.=3

conocimiento, de nuevo se aprecian diferencias estadísticamente significativas, especialmente en el caso del alumnado perteneciente a Ciencias de la salud[7].

Finalmente, los resultados en relación al último indicador analizado (depuración de la estrategia de búsqueda de información en caso necesario) y referidos a la valoración de la cantidad, calidad y relevancia de los resultados de la búsqueda para poder determinar si habría que utilizar sistemas de recuperación de información alternativos, se han basado en los datos obtenidos del ítem "*En una página web, a la hora de analizar la fiabilidad de la información que encuentras, ¿en cuáles de los siguientes indicadores te fijas principalmente?*". Dichos datos revelan que sólo un 38,9% del alumnado se fija en la autoría de la web como mejor indicador de fiabilidad; seguidamente, se apunta la fecha de actualización (en segunda posición, con un 23,4% de respuesta) y "*No me fijo en ningún indicador*" (con un 20,4% de estudiantes que han escogido esta opción como la tercera más referenciada). También en este caso, existen diferencias estadísticamente significativas entre los indicadores señalados por el alumnado en función del área de conocimiento, destacando el alto índice de estudiantes de Humanidades que señalan la autoría como principal indicador en el que se fijan a la hora de evaluar la fiabilidad de una página web (con un 66,7% de respuesta)[8].

4 Conclusiones

Los datos obtenidos, referentes a las estrategias de búsqueda y recuperación de información, con finalidades académicas, utilizadas por parte del alumnado universitario muestran que el colectivo de estudiantes universitarios no domina suficientemente ninguno de los aspectos analizados, pese a que se pueden encontrar distintos niveles de dominio, especialmente cuando se han cruzado los datos con el área de conocimiento a la que pertenecen los estudiantes.

Se constata sin embargo que los alumnos de carreras incluídas en el área de Ciencias de la Salud tienen un nivel competencial muy superior a los de otras áreas en lo que respecta al conocimiento de bases de datos especializadas. Muy probablemente es, dicho conocimiento, el que ha posibilitado que también sea este grupo el que muestra un mayor dominio de los operadores booleanos. También son los alumnos de Ciencias de la Salud los que afirman haber realizado un mayor uso de Internet cuando en el anterior curso académico se les requirió realizar un trabajo académico. Cabe pues destacar que el alumnado de Ciencias de la Salud es el que más utiliza la Red y el que posees mejores habilidades de recuperación de información.

El alumnado de Ciencias Experimentales (en las que se incluyen estudios como Biología, Física, Matemáticas o Química, entre otros) destaca en las estrategias de selección de conceptos clave a la hora de definir un tema (con un 60,5%), demostrando que no poseen las habilidades adecuadas para una correcta búsqueda y recuperación de información.

---

[7] X^2=69,287 // p=0,000 // g.l.=3

[8] X^2=27,606 // p=0,006 // g.l.=12

Más decepcionantes son los datos obtenidos del alumnado que cursa carreras adscritas a las Ciencias Sociales y Jurídicas: estos alumnos son los que, en el curso anterior, acudió menos a las bibliotecas para realizar búsquedas y recogida de información (sus estrategias de recuperación de información se centraron básicamente en la consulta a Internet), a la vez que muestran un mayor desconocimiento de las bases de datos especializadas.

Finalmente, el alumnado de Humanidades (el que está cursando carreras como Filosofía, Historia, Geografía o Filología) es el que presenta mayores tasas de frecuencia en cuanto al uso de la Biblioteca para realizar sus búsquedas y recogida de información (evidentemente, por la tradición formativa que han recibido). El alumnado de Humanidades aparece como el que mejor criterio tiene a la hora de analizar distintos indicadores de fiabilidad de una web.

Son muchas las reflexiones que pueden extraerse de los breves datos expuestos en esta comunicación. Una que nos parece especialmente importante es señalar la necesidad de proponer respuestas que conduzcan a la mejora de la situación que se ha descrito y, entre ellas, consideramos que hay una a destacar: la puesta en marcha de actividades formativas (de mayor o menor duración) a lo largo de todos los niveles educativos, las cuales tengan en cuenta la estrecha colaboración entre el personal profesional bibliotecario y el personal docente de la institución educativa.

5 Referencias

1. Zurkowski, P.G.: The information service environment relationships and priorities. Related Paper nº 5. National Commission on Libraries and Information Science, Washington, D.C. (1974).
2. American Library Association (ALA): Presidential Committee on Information Literacy: Final Report. American Library Association, Chicago. Recuperado el 10 de abril de 2010 de http://www.ala.org/ala/mgrps/divs/acrl/publications/whitepapers/presidential.cfm (1989).
3. Association of College & Research Libraries - American Library Association (ACRL-ALA): Information Literacy Competency Standards for Higher Education. Recuperado el 10 de abril de 2010 de http://www.ala.org/ala/acrl/acrlstandards/informationliteracycompetency.htm (2000).
4. Mittermeyer, D. & Quirion, D.: Information Literacy: Study of Incoming First-Year Undergraduates in Quebec. CREPUQ, Québec. Recuperado el 10 de abril de 2010 de http://www.crepuq.qc.ca/documents/bibl/formation/studies_Ang.pdf (2003).
5. SCD-URFIST: Enquête sur les besoins de formation des doctorants à la maîtrise de l'information scientifique dans les Écoles doctorales de Bretagne. UEB, Rennes (2008).

Eficiencia y precisión de algoritmos de Filtrado Colaborativo: análisis y comparativa

Fidel Cacheda, Víctor Carneiro, Diego Fernández y Vreixo Formoso

Departamento de Tecnologías de la Información y las Comunicaciones
Universidade da Coruña
{fidel,viccar,dfernandezi,vformoso}@udc.es

Resumen Los algoritmos de Filtrado Colaborativo han alcanzado una gran popularidad en los últimos años, debido especialmente a su buen comportamiento en la recomendación de productos en contextos como el comercio electrónico. Capaces de extraer información útil a partir de la interacción de miles de usuarios con el sistema, la aplicación de estos algoritmos a la recuperación de información abre caminos realmente interesantes como búsquedas personalizadas o guiadas, recomendación de consultas... Sin embargo, las técnicas tradicionales, cuyo funcionamiento se basa en encontrar los usuarios (o elementos) más parecidos a aquél para el que tenemos que generar la recomendación, no se comportan especialmente bien en este contexto debido principalmente al enorme volumen de datos a manejar y a la baja densidad de la información disponible sobre cada usuario. En este trabajo realizamos una evaluación exhaustiva de un algoritmo propuesto previamente por nosotros que intenta reducir el efecto de dichos problemas y lo comparamos con las técnicas de filtrado colaborativo más populares empleando métricas ampliamente usadas. Los experimentos realizados confirman los buenos resultados obtenidos por nuestro algoritmo a pesar de su simplicidad, mejorando en varios órdenes de magnitud la eficiencia computacional de las técnicas actuales, al tiempo que mantiene una precisión equivalente a la de éstas.

1. Introducción

Internet se ha destapado como una gran biblioteca de información que crece sin cesar. Los sistemas de recuperación de información intentan proporcionar acceso a dicha información del modo más eficiente y eficaz posible. Sin embargo, no se trata únicamente de proporcionar los elementos más relevantes, sino de mostrar aquéllos que son más adecuados para el usuario en cuestión atendiendo a sus gustos e intereses: ésta es la finalidad de los sistemas recomendadores. Para ello toda la información disponible sobre un determinado usuario se une formando lo que se denomina "perfil de usuario", y es a partir de estos perfiles como el sistema podrá predecir la relevancia de un determinado elemento o bien generar recomendaciones personalizadas. Las tareas que este tipo de algoritmos realizan son principalmente dos: "anotación en contexto" [15], consistente en informar de cuán útil puede ser un elemento para un usuario determinado; y "búsqueda

de buenos elementos", consistente en recomendar una lista de elementos presumiblemente relevantes para el usuario.

Las técnicas de Filtrado Colaborativo (CF) -en pleno auge- usan los perfiles de otros usuarios para elaborar una recomendación a un usuario concreto. En este trabajo realizaremos un análisis de diferentes algoritmos de CF existentes, poniendo especial atención en el algoritmo *Tendencies-Based* [3]. Se demostrará que éste presenta una precisión muy semejante a los mejores algoritmos mostrados, siendo además muy reseñable su simplicidad y su eficiencia computacional, lo cual lo hace muy apropiado para sistemas online y para aquéllos que involucren una gran cantidad de usuarios y/o elementos.

El resto del artículo se estructura de la siguiente forma. Primeramente se muestra el estado del arte en donde nos centraremos en describir los sistemas recomendadores -especialmente las técnicas de filtrado colaborativo-, así como las técnicas de evaluación más destacadas. A continuación, se explicarán los experimentos realizados y los resultados. Finalmente, presentaremos las conclusiones.

2. Estado del arte

En el mundo del filtrado de información ha sido el filtrado por contenido el primero en aparecer. En este caso el perfil del usuario está formado por el contenido de los elementos que éste considera más interesantes. Se trata de un sistema muy eficiente cuando se trabaja con documentos de texto. Entre sus limitaciones [15] podemos destacar el hecho de que es ineficiente con información multimedia, debido a que el análisis del contenido por parte de una máquina suele diferir de la valoración realizada por el usuario. Si bien se ha intentado solucionar por medio de contenido multimedia anotado, la realidad es que no lo resuelve totalmente, al requerir la participación humana en la anotación. Un defecto todavía más importante es que no son capaces de distinguir la calidad de un elemento. Si, por ejemplo, dos artículos emplean exactamente las mismas palabras, un sistema basado en contenido no podrá discernir entre el que está bien escrito y el que no lo está, ya que se trata de términos subjetivos que son ajenos al sistema en cuestión. Por último, otro problema que se les puede achacar es la imposibilidad de poder encontrar elementos interesantes para el usuario que no estén relacionados con aquéllos que ha buscado previamente.

Existen, por otro lado, sistemas de filtrado de información capaces de solventar estos problemas: los sistemas de filtrado colaborativo [15], basados en las valoraciones hechas por usuarios con gustos e intereses semejantes. El sistema almacenará una matriz de valoraciones en la que estarán presentes todos los usuarios y elementos, así como las valoraciones que el sistema conoce. A partir de las valoraciones de usuarios semejantes se realizará la recomendación. Según sea el proceso utilizado para ello, los algoritmos se pueden clasificar en: sistemas basados en memoria y sistemas basados en modelo.

Un sistema basado en memoria emplea en su totalidad la matriz de valoraciones para realizar una recomendación. Se trata de buscar usuarios [15] y/o elementos [13] que se asemejen lo máximo posible al usuario/elemento activo (es

por ello que también se conocen como algoritmos basados en vecinos). A pesar de su sencillez y de la buena precisión de sus resultados en términos generales suelen presentar importantes problemas de escalabilidad, puesto que, como ya se ha dicho, trabajan con la matriz de valoraciones en su totalidad. También son mucho más sensibles a los problemas comunes de los sistemas recomendadores: baja densidad de los datos [13], *cold-start* [14], y *shilling* [9].

Los sistemas basados en modelo, por su parte, construyen un modelo con la matriz de valoraciones, y obtienen posteriormente la predicción a partir de él. Los parámetros del modelo se estiman haciendo uso de los datos originales. En la literatura existen diferentes aproximaciones: basados en métodos de álgebra lineal, clustering, grafos, técnicas relacionadas con la inteligencia artificial, etc. Este tipo de sistemas se ven afectados en menor medida por los problemas mencionados anteriormente y, además, son más rápidos a la hora de realizar la recomendación. Sin embargo, presentan otra serie de problemas: parámetros demasiado difíciles de estimar, modelos que no se adaptan a los datos, consumo excesivo de tiempo para la construcción del modelo, etc. Se han propuesto diversas soluciones a dichos problemas intentando hacer más rápidos y/o sencillos los algoritmos [8,4], o combinando técnicas basadas en modelo y en memoria [7].

3. Experimentos

En este trabajo hemos evaluado los algoritmos de filtrado colaborativo más populares. Entre los algoritmos basados en memoria, las variantes basadas en usuarios [15,2], y basadas en elementos [13], así como una técnica más actual conocida como *Similarity Fusion* [17]. Entre los algorimos basados en modelo, hemos estudiado los algoritmos *Regression-based* [16], *Cluster-Based Smoothing* [18], SVD [12], así como modelos sencillos como el *Slope One* [8]. Hemos puesto especial interés en las técnicas basadas en SVD regularizado, hoy en día extremadamente populares y consideradas el algoritmo a batir en el ámbito de CF. En concreto hemos evaluado el algoritmo RSVD original [4], las variantes RSVD2 y NSVD2[10] y la variante SVD++ [7]. Finalmente, hemos analizado un modelo que integra la factorización empleando SVD con las técnicas basadas en vecinos [7], así como un modelo mixto que combina un modelo probabilístico con técnicas basadas en memoria: *Personality Diagnosis*[11]. Todos ellos han sido comparados con el algoritmo *Tendencies-based*[3]. Dicho algoritmo, en lugar de buscar relaciones entre usuarios y/o elementos, se centra en encontrar las diferencias entre ellos, diferencias que el algoritmo modela de forma sencilla: las tendencias. Dichas tendencias, que representan la diferencia entre la valoración de un par usuario-elemento respecto a la media de ambos, se emplean posteriormente para generar las recomendaciones. El algoritmo recomendará aquellos elementos con tendencia positiva, es decir, elementos valorados por encima de la media de cada usuario.

Como conjunto de datos en que basar la evaluación, hemos utilizado el ofrecido por Netflix [1], el mayor *dataset* disponible actualmente para la evaluación de algoritmos de CF. Debemos destacar, sin embargo, que, dado que muchos de

los algoritmos evaluados son sumamente ineficientes, hemos optado por elegir aleatoriamente un 30 % de los usuarios y elementos disponibles, quedándonos con un *dataset* más reducido, de cerca de 10 millones de valoraciones, sobre el que llevamos a cabo los experimetos. Además, los experimentos también han sido reproducidos con el conjunto de datos Movielens [5], dataset de pequeño tamaño pero empleado en gran número de trabajos.

Para realizar los experimentos, hemos dividido aleatoriamente el conjunto de datos en dos grupos, un subconjunto de entrenamiento y otro de evaluación, empleando para ello dos técnicas. En la primera de ellas, se escoge, aleatoriamente, un 10 % de las valoraciones existentes como conjunto de evaluación, y, de entre las restantes, un determinado porcentaje (entre 10 % y 90 %) como conjunto de entrenamiento. De esta forma, podemos evaluar la evolución de los algoritmos desde condiciones de densidad relativamente elevadas hasta condiciones de baja densidad. Precisamente, y teniendo en cuenta que en RI las condiciones de baja densidad van a ser las más frecuentes, hemos empleado un segundo método especialmente diseñado para evaluar el comportamiento de los algoritmos en estas condiciones extremas, *Given-N*[2], consistente en seleccionar como conjunto de entrenamiento un número reducido de valoraciones por usuario (hemos probado con 1, 2, 3, 4, 5, 7, 10, 12, 15 y 20).

En cuanto a qué aspectos de un algoritmo considerar, hemos realizado dos tipos de experimentos, uno destinado a evaluar la calidad de las predicciones de cada algoritmo, y otro a evaluar la calidad de las recomendaciones. Como cabría esperar, hemos empleado distintas métricas para cada tipo de experimento. Para evaluar la calidad de las predicciones, hemos empleado las métricas de precisión en la predicción tradicionales, tales como MAE (Mean Absolute Error) o RMSE (Root Mean Squared Error)[6]. En la interpretación de ambas métricas se ha tenido en cuenta el *coverage*, es decir, el porcentaje de elementos para los cuales el algoritmo ha sido capaz de obtener una predicción. Por otra parte, la calidad de las recomendaciones ha sido evaluada empleando las métricas tradicionales de precisión en la clasificación, tales como *precision and recall*, la medida F1 [12], las curvas *ROC*[6] o la métrica *Half-Life Utility*[2].

4. Resultados

En primer lugar, hemos estudiado la evolución de la precisión según la densidad de la matriz de valoraciones, comprobando la influencia del tipo de algoritmo, tal como se observa en la Figura 1. Los algoritmos basados en memoria presentan muy buenos resultados en condiciones de densidad elevada, pero empeoran significativamente a medida que ésta disminuye. Por otra parte, los algoritmos basados en modelo presentan una evolución más suave, manteniendo unos buenos resultados en condiciones menos propicias. De hecho, cuando el conjunto de entrenamiento contempla un 80 % de los datos disponibles, no existe una diferencia estadísticamente significativa entre los seis mejores algoritmos, entre los que se encuentran variantes basadas en memoria (UB) y modelo (RSVD2, SVD++, RSVD, SO y TB), en ambos *datasets*. Sin embargo, con un conjunto de

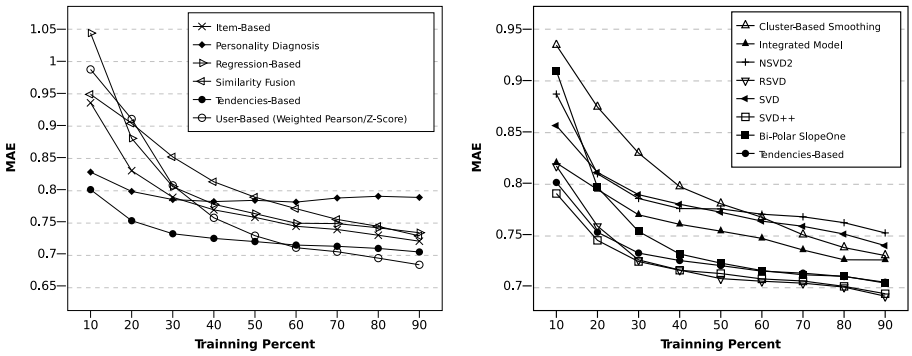


Figura 1. Evolución de la precisión (MAE) según la densidad, en el *dataset* Movielens.

entrenamiento que contemple sólo un 10 % de los datos, los algoritmos basados en memoria obtienen resultados significativamente peores.

Una conclusión que podemos extraer de este comportamiento es que cuando la información disponible es suficiente (condiciones de alta densidad), todos los algoritmos se comportan de forma parecida, obteniendo buenos resultados. A medida que la información disponible disminuye (baja la densidad) los resultados empeoran en todos los algoritmos, sólo que lo hacen a un ritmo diferente. En nuestra opinión, estas diferencias son el resultado de la capacidad del algoritmo de extraer conocimiento útil a partir de la información disponible. Los algoritmos basados en memoria sólo utilizan el parecido en usuarios o elementos, por lo que su capacidad de extraer conocimiento es muy limitada, lo cual provoca que sus resultados empeoren significativamente al disminuir la información. De hecho, el algoritmo *Similarity Fusion*, que utiliza tanto la relación entre usuarios como la relación entre elementos, presenta una curva más suave que los algoritmos UB (que sólo contempla la relación entre usuarios) e IB (que sólo tiene en cuenta la relación entre elementos). Por otra parte, los algoritmos basados en modelo son menos sensibles a los cambios de densidad, gracias a que extraen mucha más información a partir de los datos disponibles. Durante la fase de construcción del modelo, los algoritmos afinan los parámetros para ajustarse correctamente a los datos. Si el modelo es capaz de representar fielmente la realidad, no es necesaria una gran cantidad de información para realizar el ajuste. Por supuesto, cuando el modelo no representa adecuadamente los datos, los resultados obtenidos van a ser inadecuados. De hecho, como cabría suponer, la precisión de este tipo de algoritmos depende de lo bien que el modelo representa la realidad.

Esto puede comprobarse fácilmente al comparar los resultados de dos modelos diferentes: SVD regularizado frente a los basados en cluster. Los primeros obtienen resultados especialmente buenos, tanto en las métricas de precisión en la clasificación (tal y como se observa en la Figura 2) como en las métricas de precisión en la predicción. Esto se debe por una parte a que la idea de características de los elementos y el grado de afinidad de cada usuario a dichas características,

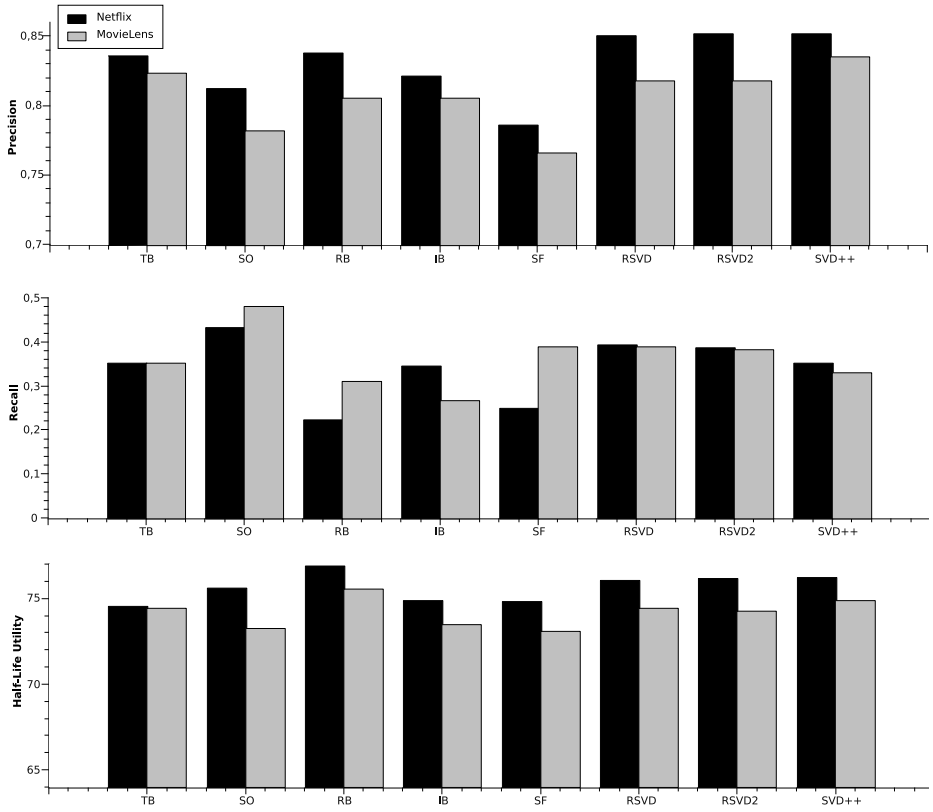


Figura 2. Resultados según las métricas de precisión en la clasificación con un conjunto de entrenamiento del 50 %.

en que se basa este modelo [4], se ajusta perfectamente a la realidad. Además, el proceso de entrenamiento que estos algoritmos emplean se demuestra muy eficaz a la hora de seleccionar las características más relevantes. En el lado opuesto están los algoritmos de clustering, que han obtenido muy malos resultados.

Por otra parte, también hemos observado que en condiciones de alta densidad los algoritmos basados en modelo presentan una precisión ligeramente inferior a técnicas basadas en memoria como UB. Esto es debido en primer lugar a los posibles errores de ajuste del modelo a la realidad, pero también a la pérdida de información que se lleva a cabo, de forma intencionada, durante la fase de construcción del modelo, de cara a tener algoritmos eficientes computacionalmente; es decir, durante la fase de entrenamiento se deben seleccionar aquellas características más determinantes y descartar el resto.

Finalmente, la eficiencia computacional de un algorimo es un factor determinante a la hora de aplicarlo a un caso real. Por ello, hemos realizado un estudio teórico de la complejidad computacional de cada algoritmo, en la que el algoritmo TB destaca como el más eficiente, con una complejidad de $O(mn)$ en el

entrenamiento, y de $O(1)$ en la predicción, superando con creces al resto de alternativas. Además, los resultados teóricos se confirman en los tiempos alcanzados por los algoritmos. El tiempo de entrenamiento del algoritmo TB, 13,2 ms en el *dataset* Movielens, es varios órdenes de magnitud mejor que los de algoritmos con una precisión similar, tales como las técnicas RSVD (5,433 ms), RSVD2 (13,198 ms) o SVD++ (71,025 ms). A su vez, su tiempo de predicción (12 ms), es similar a los de otras aproximaciones basadas en modelo, y muy inferior a los de algoritmos basados en memoria como el UB. Con el *dataset* Netflix se obtienen también buenos resultados, con un tiempo de entrenamiento de tan sólo 4 segundos, frente a los más de 10 minutos de RSVD, los casi 15 minutos de RSVD2 o las más de 2 horas y media de SVD++[1].

5. Conclusiones

A lo largo de este trabajo se han realizado diversos experimentos en distintas condiciones para comparar algoritmos de filtrado colaborativo. Se ha empleado una metodología bien definida y las métricas más ampliamente usadas con el fin de evaluar los resultados. Todos los resultados obtenidos conducen a la conclusión de que el algoritmo *Tendencies-Based* en el dominio con el que tratamos (la recomendación de películas) alcanza la precisión de los métodos más modernos empleados, y su eficiencia computacional es muy superior. Hemos comprobado que los algoritmos basados en memoria alcanzan una precisión muy alta con una matriz de datos relativamente densa, pero a medida que la densidad disminuye los resultados empeoran considerablemente; por su parte, los algoritmos basados en modelo llegan a resultados muy similares a los primeros con una matriz densa y el descenso de la calidad de la precisión obtenida no es tan acusado. Estos resultados se pueden explicar por el hecho de que no todos los algoritmos tienen las mismas capacidades para extraer información (relaciones entre usuarios, entre elementos, etc.) a partir de la matriz de datos original.

En cuanto a los trabajos futuros, sería interesante repetir el estudio realizado en dominios de distinta índole. El hecho de haber obtenido tan buenos resultados con el algoritmo *Tendencies-Based* abre las puertas para su empleo en otros campos de la Recuperación de Información.

Referencias

1. Bennett, J., Lanning, S.: The netflix prixe. In: KDDCup '07: Proceedings of KDD Cup and Workshop. p. 4. ACM, San Jose, California, USA (2007)
2. Breese, J.S., Heckerman, D., Kadie, C.: Empirical analysis of predictive algorithms for collaborative filtering. In: Proceedings of the Fourteenth Annual Conference on Uncertainty in Artificial Intelligence. pp. 43–52 (1998)

[1] Los tiempos han sido medidos en un PC con procesador Quad Core Xeon a 2,66GHz, 2x6 MB cache, 1333FSB y 8GB de RAM. Los algoritmos han sido implementados en Java.

3. Cacheda, F., Carneiro, V., Fernández, D., Formoso, V.: Comparison of collaborative filtering algorithms: limitations of current techniques and proposals for scalable, high-performance recommender systems. ACM Transactions on the Web (To Appear)
4. Funk, S.: Netflix Update: Try This at Home. http://sifter.org/ simon/journal/20061211.html (2006)
5. Herlocker, J.L., Konstan, J.A., Borchers, A., Riedl, J.: An algorithmic framework for performing collaborative filtering. In: SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval. pp. 230–237. ACM, New York, NY, USA (1999)
6. Herlocker, J.L., Konstan, J.A., Terveen, L.G., Riedl, J.T.: Evaluating collaborative filtering recommender systems. ACM Trans. Inf. Syst. 22(1), 5–53 (2004)
7. Koren, Y.: Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: KDD '08: Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 426–434. ACM, New York, NY, USA (2008)
8. Lemire, D., Maclachlan, A.: Slope one predictors for online rating-based collaborative filtering. In: Proceedings of SIAM Data Mining (SDM'05) (2005)
9. Mobasher, B., Burke, R., Bhaumik, R., Williams, C.: Toward trustworthy recommender systems: An analysis of attack models and algorithm robustness. ACM Trans. Interet Technol. 7(4), 23 (2007)
10. Paterek, A.: Improving regularized singular value decomposition for collaborative filtering. In: Proc. KDD Cup Workshop at SIGKDD'07, 13th ACM Int. Conf. on Knowledge Discovery and Data Mining. pp. 39–42 (2007)
11. Pennock, D., Horvitz, E., Lawrence, S., Giles, C.L.: Collaborative filtering by personality diagnosis: A hybrid memory- and model-based approach. In: Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence, UAI 2000. pp. 473–480. Stanford, CA (2000)
12. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Application of dimensionality reduction in recommender systems–a case study. In ACM WebKDD Workshop (2000)
13. Sarwar, B., Karypis, G., Konstan, J., Reidl, J.: Item-based collaborative filtering recommendation algorithms. In: WWW '01: Proceedings of the 10th international conference on World Wide Web. pp. 285–295. ACM, New York, NY, USA (2001)
14. Schein, A.I., Popescul, A., Ungar, L.H., Pennock, D.M.: Methods and metrics for cold-start recommendations. In: SIGIR '02: Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval. pp. 253–260. ACM, New York, NY, USA (2002)
15. Shardanand, U., Maes, P.: Social information filtering: algorithms for automating "word of mouth". In: CHI '95: Proceedings of the SIGCHI conference on Human factors in computing systems. pp. 210–217. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA (1995)
16. Vucetic, S., Obradovic, Z.: A regression-based approach for scaling-up personalized recommender systems in e-commerce. In ACM WebKDD Workshop (2000)
17. Wang, J., de Vries, A.P., Reinders, M.J.T.: Unifying user-based and item-based collaborative filtering approaches by similarity fusion. In: SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval. pp. 501–508. ACM, New York, NY, USA (2006)
18. Xue, G.R., Lin, C., Yang, Q., Xi, W., Zeng, H.J., Yu, Y., Chen, Z.: Scalable collaborative filtering using cluster-based smoothing. In: SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval. pp. 114–121. ACM, New York, NY, USA (2005)

Mejorando la búsqueda directa en texto comprimido *

Antonio Fariña, Susana Ladra, José R. Paramá, Ángeles S. Places, Ángel Yáñez

Laboratorio de Bases de Datos, Univ. de A Coruña, España. Campus de Elviña s/n, A Coruña, España
{fari,sladra,parama,asplaces,infaym00}@udc.es

Resumen La aparición de las técnicas de compresión estadísticas basadas en palabras y orientadas a byte supuso un gran avance para la búsqueda secuencial de patrones en un texto. Estas técnicas, asociadas a un modelador semiestático, consiguen comprimir los textos alrededor de un 31-33 % del tamaño original, pero lo más importante, consiguen buscar secuencialmente en la versión comprimida hasta 8 veces más rápido que en la versión original.
Este avance abrió las puertas a las bases de datos de texto comprimidas, donde el texto es almacenado e indexado en su versión comprimida.
En este trabajo mostramos cómo la adición de una mínima información de filtrado permite acelerar las búsquedas sobre texto comprimido, a costa de un ligero empeoramiento de la tasa de compresión. Esta estrategia ha sido aplicada a la técnica End-Tagged Dense Code, reduciendo sensiblemente (hasta 10 veces) los tiempos de búsqueda y empeorando únicamente su compresión en torno a un 1-3 % (en textos grandes).

1. Introducción

Desde la década de los noventa, una nueva generación de técnicas de compresión específicas para texto permiten buscar un patrón dentro de la versión comprimida hasta 8 veces más rápido que sobre el texto original [8]. El objetivo es mantener el texto siempre comprimido, pero para que esto sea factible se deben cumplir dos premisas: *i)* debe ser posible descomprimir a partir de cualquier punto del texto, *ii)* debe ser posible empezar a buscar desde cualquier punto. La primera condición es necesaria para poder mostrar el texto alrededor de una ocurrencia de un patrón buscado. La segunda permite la utilización de índices invertidos a bloques, que dado un patrón, nos indica los bloques donde hay ocurrencias que tendrían que ser buscadas secuencialmente.

La primera de estas técnicas se denominó *Tagged Huffman* (TH) [8]. TH hace uso de una técnica de codificación estadística clásica denominada *codificación Huffman* [5]. Huffman, como cualquier técnica estadística, asigna códigos a los símbolos originales, estos códigos son secuencias de bits más cortas para

* Parcialmente financiado por: el "Ministerio de Ciencia e Innovación"(MICINN ref. TIN2009-14560-C03-02)

los símbolos originales más frecuentes y más largas para los símbolos menos frecuentes. Para obtener la frecuencia de cada símbolo, TH utiliza un modelador semiestático que realiza una pasada previa sobre el texto para recopilar los diferentes símbolos existentes y sus frecuencias. En una segunda pasada, cada símbolo original es reemplazado por el código que la codificación Huffman le asignó después de la primera pasada.

La siguiente característica destacable del TH es que el alfabeto de entrada son palabras, en lugar de los tradicionales caracteres. Para textos suficientemente grandes, los compresores basados en Huffman y orientados a palabras logran comprimir el texto al 25 % de su tamaño original [6], en lugar del 60 % que conseguía la versión orientada a caracteres.

Finalmente, TH aportó dos elementos novedosos. El primero es que el alfabeto de salida son bytes en lugar de bits. Tradicionalmente los códigos que se le asignan a los símbolos originales solían ser secuencias de bits, pero en TH cada código es una secuencia de uno o más bytes. Esto supuso una pérdida de compresión del 11 %, pero la ventaja es que la manipulación de bytes es mucho más rápida y sencilla que la de bits, lo que supone una importante mejora en todos los tiempos. El otro elemento novedoso introducido por TH fue la inclusión de un bit de marca en cada uno de los bytes que forman un código. Este bit se establece a 1 cuando se trata del primer byte de un código y a 0 en los restantes bytes (si existen). Este bit proporciona sincronización a la hora de buscar y descomprimir; al poder localizar el comienzo de cada código es posible empezar a buscar o descomprimir en cualquier punto del texto.

Los códigos densos [3] supusieron una mejora con respecto a TH tanto en ratios de compresión como en la velocidad de compresión y descompresión. El primer código denso fue el End-Tagged Dense Code (ETDC), que como TH, es una codificación estadística orientada a bytes con un bit de marca en cada byte. ETDC comprime alrededor de 2-3 puntos porcentuales más que TH, además de ser más rápido en la compresión, descompresión y búsqueda.

En este trabajo mostraremos una variante del ETDC denominada End-Tagged Dense Code con Filtrado (ETDC^f), que añadiendo una pequeña cantidad de información en el preludio del fichero comprimido, permite acelerar las búsquedas hasta 10 veces con respecto a búsquedas en ETDC. Evidentemente, esta información añadida supone una pérdida de compresión, pero ésta es de tan sólo del 1 %-3 % en textos grandes.

2. End Tagged Dense Code

Para obtener el código de cada palabra original, ETDC usa un modelo semiestático del texto que es simplemente el vocabulario (la lista de palabras originales presentes en el texto) ordenado por frecuencia de aparición. El proceso de codificación es como sigue:

- Los códigos de 1 byte del 128 al 255 son asignados a las 128 palabras más frecuentes del vocabulario.

- Las palabras desde la posición[1] 128 a la posición $128+128^2-1$ del vocabulario son codificadas con códigos de 2 bytes. El primer byte de cada código tiene un valor del rango $[0, 127]$ y el segundo byte toma valores en el rango $[128, 255]$.
- Las palabras entre las posiciones $128 + 128^2$ y $128 + 128^2 + 128^3 - 1$ reciben códigos de tres bytes, etc.

3. End-Tagged Dense Code con Filtrado (ETDC^f)

ETDC^f es una variante del ETDC que almacena la primera y la última aparición de cada palabra en el texto para acelerar su búsqueda. Además, también almacena la distancia mínima existente entre apariciones de una misma palabra, de modo que el buscador pueda efectuar saltos cada vez que encuentre el patrón que está siendo buscado.

3.1. Funcionamiento y Estructuras utilizadas por ETDC^f

El ETDC^f es una técnica de compresión *semiestática* y por lo tanto, realiza *dos pasadas* sobre el texto a comprimir. La primera recaba las palabras existentes en el texto y su frecuencia de aparición, y la segunda sustituye cada palabra original w_i por su correspondiente código C_{w_i}.

La principal diferencia existente en el proceso de compresión de ETDC^f frente al ETDC estriba en que el compresor debe ser capaz de obtener la nueva información de filtrado. Durante la segunda pasada de la compresión, a medida que se van emitiendo los códigos correspondientes a cada palabra, el ETDC^f va obteniendo la información de filtrado. De este modo, para cada palabra w_i anota tres valores: *i)* la posición de *inicio* de la primera aparición del código C_{w_i}; *ii)* la posición dónde aparece C_{w_i} por *última* vez; y *iii)* el menor *salto* (medido en número de bytes) existente entre dos apariciones consecutivas de C_{w_i} en el texto comprimido. De esta forma, el fichero comprimido obtenido por el ETDC^f contiene el vocabulario ordenado por frecuencia y la representación comprimida del texto (al igual que en ETDC), y adicionalmente, tres arrays: *inicio*, $final$ y *salto*. El proceso de descompresión es totalmente análogo al del ETDC.

3.2. Búsquedas directas sobre texto comprimido con ETDC^f

La búsqueda directa sobre texto comprimido se realiza comprimiendo el patrón y buscando esa versión del patrón directamente en el texto comprimido. Al igual que en ETDC, el algoritmo más adecuado para está búsqueda es Horspool [4,9]. Esta es una simplificación del algoritmo Boyer-Moore [2] , que resulta especialmente adecuada para la búsqueda sobre texto comprimido, dónde los patrones son cortos (1-4 bytes) y la distribución de frecuencias de los bytes del fichero comprimido es relativamente uniforme [9].

[1] Las posiciones de palabras comienzan en cero.

Búsquedas de palabras simples. Una vez obtenida la versión comprimida del patrón P, se realiza una primera *fase de preprocesado* del patrón en la que se precalcula una tabla de saltos (d) que será utilizada en la segunda fase o *fase de búsqueda*. En esta segunda fase, el patrón P se va desplazando a lo largo del texto comprimido T. Cada vez que se avanza el patrón, se compara $P_1, \cdots, P_m$ "hacia atrás" con el texto $T_i, \cdots, T_{i+m-1}$. Si se da una coincidencia, se notifica una aparición del patrón en la posición i. El proceso de búsqueda continúa avanzando el patrón sobre el texto un número de bytes que depende únicamente de la última posición del texto que está alineada con el patrón en el momento actual; en concreto, se desplaza el patrón $d[T_{i+m-1}]$ bytes.

La Figura 1 compara el algoritmo de búsqueda sobre ETDC con el usado en ETDC^f. Como se puede ver, las diferencias entre ambos se centran en tres aspectos:

- El proceso de búsqueda no comienza al principio del texto comprimido, sino en la primera ocurrencia del patrón buscado (línea 5).
- Análogamente no es necesario procesar todo el texto comprimido al saber de antemano la posición de la última aparición del cada palabra en el texto (línea 5).
- Tras cada emparejamiento del patrón con el texto, no se saltan únicamente $d[T_{i+m-1}]$ bytes, sino que podemos avanzar **salto**$(w_i) + m - 1$ bytes, sin miedo a perder ninguna aparición del patrón.

```
Búsqueda_ETDC (w_i, T)
(1) P ←encodeETDC(w_i)
(2) m ← |P|, n ← |T|
(3) for b ← 0 to 255 do d[b] ← m
(4) for j ← 1 to m − 1 do d[P[j]] ← m − j
(5) i ← 0
(6) while i ≤ n − m
(7)     j ← m − 1
(8)     while j ≥ 0 and T[i + j] = P[j] do
(9)         j ← j − 1
(10)    if j < 0 and (i = 0 or T[i − 1] ≥ 128)
(11)        output i
(12)    i ← i + d[T[i + m − 1]]
```

```
Búsqueda_ETDC^f (w_i, T)
(1) P ←encodeETDC(w_i)
(2) m ← |P|, n ← |T|
(3) for b ← 0 to 255 do d[b] ← m
(4) for j ← 1 to m − 1 do d[P[j]] ← m − j
(5) i ←inicio(w_i), n ←min(final(w_i), n)
(6) while i ≤ n − m
(7)     j ← m − 1
(8)     while j ≥ 0 and T[i + j] = P[j] do
(9)         j ← j − 1
(10)    if j < 0 and (i = 0 or T[i − 1] ≥ 128)
(11)        output i
(12)        i ← i+salto(w_i) +m − 1
(13)    else i ← i + d[T[i + m − 1]]
```

Figura 1. Búsqueda de un patrón P sobre el texto comprimido T.

Búsqueda de frases

Las estructuras utilizadas por ETDC^f no sólo sirven para mejorar la búsqueda de palabras, sino que permiten también agilizar la búsqueda de frases sobre texto comprimido con ETDC. De nuevo, el proceso de búsqueda comienza obteniendo la versión comprimida del patrón concatenando los códigos de las palabras del patrón. Dado que para cada palabra w_i que forma la frase disponemos de los

valores $\langle inicio_i, salto_i, final_i \rangle$, la búsqueda puede beneficiarse de los siguientes factores:

- Si una palabra w_i del patrón no aparece hasta la posición x entonces la frase no puede aparecer hasta la posición $x - m$. Por lo tanto, el proceso de búsqueda con Horspool no necesita comenzar antes de la posición $i \leftarrow \text{máx}(inicio(w_1), \cdots, inicio(w_m)) - m$.
- La posición de la última aparición de la frase siempre será igual o inferior a la menor de las últimas apariciones de todas las palabras que forman la frase. Por ello, la búsqueda no necesita procesar más allá de la posición $i \leftarrow \text{mín}(final(w_1), \cdots, final(w_m)) + m$.
- Un ocurrencia de una frase implica la aparición de todas sus palabras, por lo tanto, el salto que se puede realizar es: $\text{máx}(salto(w_1), \cdots, salto(w_m))$.

4. Resultados experimentales

Hemos realizado experimentos sobre un conjunto de textos TREC. De TREC-2 se han utilizado "AP Newswire 1988" y "Ziff Data 1989-1990", y de TREC-4 "Congressional Record 1993", "Financial Times 1991, 1992, 1993 y 1994". A su vez *FTALL* agrega los corpus provenientes del Financial Times. Hemos creado un corpus aún mayor (*ALL*: de aproximadamente 1Gb) agrupando los corpus anteriores y el corpus Calgary[2].

Se utilizó una Intel®Pentium®-IV@ 3.00 GHz (16Kb L1 + 1024Kb L2 cache), con 4 GB dual-channel DDR-400Mhz RAM, con Sistema Operation Debian GNU/Linux (kernel 2.4.27). El compilador usado fue *gcc* versión 3.3.5 y opciones de optimización *-O3*. Los resultados que muestran tiempos miden tiempo de CPU en modo usuario (*CPU user-time*) en milisegundos.

Presentamos una comparativa del ETDC^f frente a ETDC y al TH. Los resultados se centran en la tasa de compresión y velocidad de búsqueda. No incluimos en esta comparación los tiempos de compresión (que ahora son aproximadamente un 1-5 % más lentos que en ETDC), ni los tiempos de descompresión del ETDC^f por ser idénticos a los obtenidos por ETDC.

4.1. Tasa de compresión

La Tabla 1 muestra los resultados. En textos pequeños el peso de los tres arrays incluidos en ETDC^f tiene una influencia considerable en la tasa de compresión. Sin embargo, en colecciones de textos de mayor tamaño, la pérdida de compresión con respecto al ETDC está sólo en torno a 1 punto porcentual.

4.2. Tiempo de búsqueda

Búsqueda de palabras simples. Siguiendo las ideas propuestas en [8], hemos extraído 1.000 palabras al azar del vocabulario obtenido en cada texto, asumiendo probabilidad uniforme y descartado las palabras que ocurren 1 única vez. En

[2] `http://corpus.canterbury.ac.nz/.`

Corpus	Tamaño (bytes)	TH	ETDC	ETDC^f
FT91	14.749.355	37,93 %	35,53 %	41,69 %
CR	51.085.545	34,28 %	31,94 %	34,71 %
FT92	175.449.615	35,39 %	32,82 %	34,76 %
ZIFF	185.220.211	36,31 %	33,77 %	35,31 %
FT93	197.586.334	35,49 %	32,87 %	34,64 %
FT94	203.783.923	35,45 %	32,83 %	34,57 %
AP	250.634.187	35,45 %	32,90 %	33,97 %
FTALL	591.569.227	35,13 %	32,53 %	33,70 %
ALL	1.080.720.303	36,38 %	33,66 %	34,64 %

Tabla 1. Comparación en tasa de compresión de TH, ETDC y ETDC^f.

la Tabla 2 se muestran los tiempos medios de búsqueda de los 1.000 patrones. Se puede apreciar que las búsquedas sobre texto comprimido con ETDC^f son obviamente las más rápidas, siendo en determinados casos hasta 3 veces más rápidas que en el caso de usar ETDC.

Corpus	TH	ETDC	ETDC^f	Corpus	TH	ETDC	ETDC^f
FT91	14,9	9,6	3,7	FT94	100,1	119,7	46,3
CR	32,0	30,8	11,9	AP	123,5	150,5	55,7
FT92	88,7	103,1	39,4	FTALL	266,0	343,6	138,7
ZIFF	96,4	113,6	41,8	ALL	483,4	646,7	187,2
FT93	96,6	116,6	46,2				

Tabla 2. Comparación de tiempos medios de búsqueda (en ms.) de palabras simples.

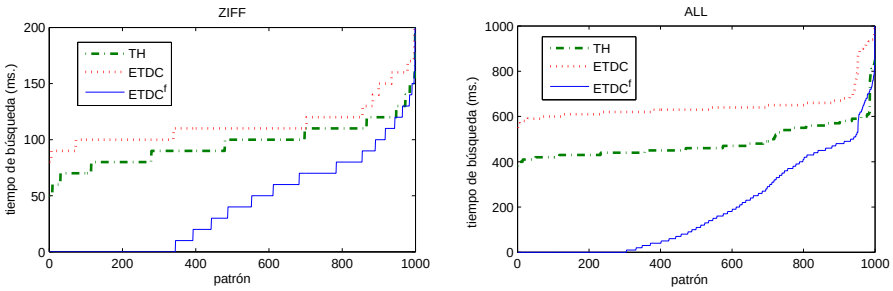


Figura 2. Tiempo de búsqueda para cada uno de los 1.000 patrones buscados.

En la Figura 2 se muestra el tiempo necesario para localizar cada uno de los 1.000 patrones buscados tanto en el corpus ZIFF, como en el ALL. Los patrones mostrados en dichas gráficas se han ordenado en función de su tiempo de búsqueda para mejorar la legibilidad. Obsérvese que el ETDC^f es capaz de encontrar todas las apariciones de aproximadamente el 30-40 % de los patrones

de forma casi instantánea. Dichos patrones son los que poseen una frecuencia baja (y normalmente un gran salto entre sus apariciones), o simplemente aquellos que se encuentran localizados en una porción pequeña del texto. El único caso desfavorable al ETDC^f se da cuando dos apariciones del patrón están muy próximas en el texto (aunque el resto de las apariciones estén muy separadas), ya que esto limitará el salto a realizar tras una aparición del patrón, perjudicando el tiempo de búsqueda.

Si nos ceñimos a palabras en el caso más favorable para ETDC^f, por ejemplo, palabras con entre 2 y 10 apariciones, las búsquedas con ETDC^f llegan a mejorar hasta en 10 veces los tiempos del ETDC.

Búsqueda de frases. Se han extraído del texto conjuntos de 1.000 frases obtenidas a partir de 1.000 posiciones de la colección elegidas al azar. En concreto hemos creado conjuntos formados por frases de 2, 5, 7 y 9 palabras. En este caso, nos hemos centrado en la comparativa entre ETDC y ETDC^f.

Como se puede observar, ETDC^f también resulta de interés cuando se buscan frases. En frases con pocas palabras la mejora está en torno al 10-25 % frente al ETDC. Cuando se buscan frases más largas (y que posiblemente aparecerán un reducido número de veces) los tiempos de búsquedas mejoran notablemente (sobre un 25-40 %).

Texto	Pal. Frase	ETDC	ETDC^f	Texto	Pal. Frase	ETDC	ETDC^f
FT91	2	8,42	5,72	AP	2	144,35	108,20
	5	4,00	2,25		5	67,18	49,82
	7	3,16	1,82		7	51,50	38,83
	9	2,73	1,66		9	43,40	32,31
ZIFF	2	110,31	82,07	ALL	2	638,07	466,44
	5	52,06	38,19		5	290,37	200,03
	7	39,76	29,63		7	225,04	151,94
	9	34,00	24,60		9	188,61	120,00

Tabla 3. Comparación del tiempo de búsqueda de frases (en ms.).

5. Conclusiones y trabajo futuro

En este trabajo se ha mostrado ETDC^f una simple modificación sobre ETDC que permite mejorar el rendimiento de las búsquedas sobre texto comprimido. Dicha técnica consiste en añadir al vocabulario utilizado por el compresor información de filtrado que permite reducir la cantidad de texto comprimido que debe ser procesado.

En general, ETDC^f presenta excelentes tiempos de búsqueda de palabras simples, y pierde en torno a 1-1,5 puntos porcentuales en tasa de compresión frente a la versión del ETDC original. Además se ha visto que esta técnica también permite mejorar las búsquedas de frases (secuencia de palabras). Para

ello nos hemos centrado en el hecho de que una frase no puede existir sin todas las palabras que la construyen y a que ETDC^f dispone de información relativa a en qué posiciones pueden aparecer una palabra dada. De este modo es posible acortar aún más la cantidad de texto comprimido que hay que procesar durante la búsqueda (desde la aparición más tardía de todas las palabras de la frase hasta la más temprana posición de la última ocurrencia de dichas palabras) y agrandar el salto que se puede realizar tras un emparejamiento.

Como trabajo futuro, en primer lugar pretendemos aplicar esta estrategia de filtrado a otras técnicas autosincronizadas como el SCDC o el TH. Seguidamente, también puede resultar de interés su aplicación junto a técnicas como el Huffman basado en palabras y orientado a bit [7], ya que permitiría obtener tasas de compresión por debajo del 30 %. Adicionalmente, aunque en principio no parece factible el realizar saltos tras una aparición del patrón buscado, ya que el buscador perdería el sincronismo en el texto comprimido, estamos interesados en estudiar la aplicabilidad de las ideas mostradas en [1]. En este caso el buscador podría avanzar, y posteriormente debería re-sincronizarse para poder continuar la búsqueda.

Referencias

1. M. T. Biskup and W. Plandowski. Guaranteed synchronization of huffman codes with known position of decoder. In *Proc. Data Compression Conference 2009*, pages 33–42. IEEE Computer Society, 2009.
2. R. S. Boyer and J. S. Moore. A fast string searching algorithm. *Communications of the ACM*, 20(10):762–772, 1977.
3. N. Brisaboa, A. Fariña, G. Navarro, and Paramá J. Lightweight natural language text compression. *Information Retrieval*, 10(1):1–33, 2007.
4. R. N. Horspool. Practical fast searching in strings. *Software Practice and Experience*, 10(6):501–506, 1980.
5. D. A. Huffman. A method for the construction of minimum-redundancy codes. *Proceedings of the IRE*, 40(9):1098–1101, 1952.
6. A. Moffat. Word-based text compression. *Softw. Pract. Exper.*, 19(2):185–198, 1989.
7. A. Moffat and A. Turpin. Fast file search using text compression. In *Proc. 20th Australian Computer Science Conference*, pages 1–8, 1997.
8. E. Moura, G. Navarro, N. Z., and R. Baeza-Yates. Fast and flexible word searching on compressed text. *ACM TOIS*, 18(2):113–139, 2000.
9. G. Navarro and M. Raffinot. *Flexible Pattern Matching in Strings – Practical on-line search algorithms for texts and biological sequences*. Cambridge University Press, 2002.

Índice de Autores