Informe Técnico − Technical Report
DPTOIA−IT−2002−006
mayo, 2002

Recuperación de información utilizando el modelo vectorial. Participación en el taller CLEF−2001

Ángel F. Zazo Rodríguez
Carlos G.−Figuerola Paniagua
José Luis Alonso Berrocal
Raquel Gómez Díaz

Departamento de Informática y Automática
Universidad de Salamanca

Revisado por:

Dr. Luis Antonio Miguel Quintales
Departamento de Informática y Automática
Universidad de Salamanca
lamq@usal.es

Dr. José Antonio Cordón García
Departamento de Biblioteconomía y Documentación
Universidad de Salamanca
jcordon@usal.es

Aprobado en el Consejo de Departamento de 20−05−2002

Información de los autores:

D. Ángel F. Zazo Rodríguez,
Área de Lenguajes y Sistema Informáticos. Departamento de Informática y Automática
Facultad de Traducción y Documentación − Universidad de Salamanca
C/ Francisco Vitoria, 6−16. 37008 − Salamanca
afzazo@usal.es

Dr. Carlos G.−Figuerola Paniagua
Área de Lenguajes y Sistema Informáticos. Departamento de Informática y Automática
Facultad de Traducción y Documentación − Universidad de Salamanca
C/ Francisco Vitoria, 6−16. 37008 − Salamanca
figue@usal.es

Dr. José Luis Alonso Berrocal
Área de Lenguajes y Sistema Informáticos. Departamento de Informática y Automática
Facultad de Traducción y Documentación − Universidad de Salamanca
C/ Francisco Vitoria, 6−16. 37008 − Salamanca
berrocal@usal.es

Dra. Raquel Gómez Díaz
Licenciada en Documentación. Doctora por la Universidad de Salamanca.
rgomez@usal.es

Este trabajo ha sido parcialmente financiado por el proyecto de investigación SA59/00B, sub−vencionado por la Junta de Castilla y León y el Fondo Social Europeo.

Resumen

En este documento se describe el proceso seguido para la construcción del sistema de recuperación de información que hemos utilizado en nuestra participación en CLEF−2001. El sistema utiliza el conocido modelo vectorial. Se analiza primeramente el problema de la recuperación de información, el modelo vectorial y la evaluación sobre una colección de pruebas. Seguidamente se describe el procesado léxico realizado sobre el contenido de documentos y consultas. Se finaliza con los resultados obtenidos en los experimentos y las conclusiones.

Abstract

This document describes the construction process of an information retrieval system (IRS) for our participation in CLEF−2001. We use the vector model. First, the document provides a brief description of the information retrieval problem, the vector model and the evaluation of the IRS. Next, the lexical analysis applied to documents and queries is described. Finally, we show the results and the conclusions.

Índice de contenido

1. Introducción

En este trabajo se describe la participación de nuestro equipo en el taller **Cross−Language Evaluation Forum** (CLEF), que precedió a la *Fifth European Conference on Digital Libraries* (ECDL−2001), llevada a cabo del 4 al 8 de septiembre de 2001 en Darmstadt (Alemania). Dichos eventos son actividades desarrolladas por el grupo de trabajo *DELOS Network of Excellence* del programa IST (*Information Society Technologies*) de la Comisión Europea (inicialmente fundado por el programa ESPRIT en 1996) para promover el desarrollo de tecnologías para la biblioteca digital. Uno de los fines primordiales de DELOS es proporcionar acceso a la información rompiendo las barreras idiomáticas o culturales, de ahí la necesidad de las actividades en recuperación de información plurilingüe.

CLEF se desarrolla en colaboración con las famosas TREC (*Text REtrieval Conference*) [NIST, 2002] para la evaluación de diferentes sistemas de recuperación de información plurilingüe. La principal meta de las TREC es establecer un foro para desarrollar la investigación en el campo de la Recuperación de Información (RI), proporcionando a los investigadores la infraestructura necesaria para la evaluación de sus metodologías de recuperación sobre grandes colecciones de documentos, con el objetivo último de conseguir mayor rapidez en la transferencia de resultados. Como el campo de la RI en muy amplio, en cada TREC se han ido añadiendo tareas a resolver: desde la recuperación monolingüe en varios idiomas, la recuperación multilingüe, el procesamiento del lenguaje natural, el tratamiento de enormes volúmenes de datos, los interfaces, la exploración Web, etc. Los participantes pueden elegir varios de los temas en que estén interesados.

En cada conferencia TREC se proporciona una colección de prueba conteniendo documentos y consultas, y se especifican las tareas para esa conferencia. Los participantes aplican sus sistemas de recuperación sobre la colección y mandan los resultados, que son evaluados por un comité supervisor. Finalmente se desarrolla un taller (*workshop*) en el que los participantes comparten sus experiencias. La colección de prueba y el software de evaluación están disponibles libremente para los investigadores, para que evalúen sus sistemas en cualquier momento.

Desde la primera conferencia TREC en 1992, se ha doblado la efectividad en los sistemas de RI, y se ha conseguido el doble objetivo de avanzar en los sistemas de recuperación y en la transferencia de resultados. Muchos de los motores de búsqueda hoy día tan utilizados fueron antes probados en estas conferencias. Ahora estamos por la novena conferencia TREC.

Aunque la lengua de trabajo inicial en las TREC fue el inglés, ya desde la TREC−3 se incluyó el español, pero en recuperación monolingüe. La recuperación multilingüe, es decir, resolver consultas en un idioma determinado frente a documentos escritos en otro idioma, no se estableció como tema de trabajo adicional a los existentes hasta la TREC−6 en 1997, bajo las siglas CLIR (*Cross−Languge Information Retrieval*) [Voorhees y Harman, 1997]. La evaluación de resultados de la última campaña de CLIR se produjo en el CLEF−2001.

En dicha campaña se puso a disposición de los investigadores una colección de documentos para cada una de las 6 lenguas (inglés, francés, alemán e italiano, ya existentes en CLEF−2000, más el holandés y el español en esta campaña [Peters, 2001]) con una batería de 50 preguntas (*topics*) en esos idiomas y también en otros tres idiomas europeos y tres asiáticos. Se incluyeron varias tareas de evaluación: recuperación multilingüe, recupeación bilingüe teniendo el inglés como una de las lenguas de documentos o consultas, recuperación monolingüe, recuperación de sistemas interactivos y recuperación mono y multilingüe en el campo temático de las ciencias sociales.

En nuestro caso, nos hemos centrado en la recuperación monolingüe de documentos en español.

1.1. La colección

La colección de documentos proviene de la agencia de noticias EFE, de todas las noticias del año 1994. Se trata de 215.718 documentos (534 MB de información, 1MB = 1000 KB), alma−cenadas en ficheros, uno por día del año 1994. A esta colección la hemos denominado EFE'94 en este informe. Cada archivo contiene varios documentos, y dentro de estos existen campos delimitados con etiquetas SGML, tal como se indica en un ejemplo en la figura 1. La codifica−ción de los documentos está descrito en SGML como "ISO Registration Number 100//CHAR−SET ECMA−94 Right Part of Latin Alphabet Nr. 1//ESC 2/13 4/1". Los campos que nos inte−resan son **TITLE** y **TEXT**, pues el resto contiene información como fecha, sección del perió−dico, etc. El número medio de palabras por documento, considerados dichos campos, es de 333.

```
<DOC>
<DOCNO>EFE19940101−00002</DOCNO>
<DOCID>EFE19940101−00002</DOCID>
<DATE>19940101</DATE>
<TIME>00.34</TIME>
<SCATE>VAR</SCATE>
<FICHEROS>94F.JPG</FICHEROS>
<DESTINO>ICX MUN EXG</DESTINO>
<CATEGORY>VARIOS</CATEGORY>
<CLAVE>DP2404</CLAVE>
<NUM>100</NUM>
<PRIORIDAD>U</PRIORIDAD>
<TITLE>   IBM−WATSON
           FALLECIO HIJO FUNDADOR EMPRESA DE COMPUTADORAS
</TITLE>
<TEXT>    Nueva York, 31 dic (EFE).− Thomas Watson junior, hijo del fundador
 de International Business Machines Corp. (IBM), falleció hoy,
 viernes, en un hospital del estado de Connecticut a los 79 años de
 edad, informó un portavoz de la empresa.
    Watson falleció en el hospital Greenwich a consecuencia de
 complicaciones tras sufrir un ataque cardíaco, añadió la fuente.
    El difunto heredó de su padre una empresa dedicada principalmente
 a la fabricación de máquinas de escribir y la transformó en una
 compañía líder e innovadora en el mercado de las computadoras. EFE
    PD/FMR
    01/01/00−34/94
</TEXT>
</DOC>
```

Figura 1. Un documento de la colección.

Junto con los documentos se acompaña la batería de 50 preguntas en español. Cada pre−gunta se divide en tres campos, **ES−title**, **ES−desc** (descripción) y **ES−narr** (narrativa), como se aprecia en la figura siguiente. Pueden lanzarse las consultas con uno o todos los campos.

```
<top>
<num> C042 </num>
<ES−title> Naciones Unidas y Estados Unidos invaden Haití </ES−title>
<ES−desc> Encontrar documentos sobre la invasión de Haití por los
soldados de la ONU y de los Estados Unidos. </ES−desc>
<ES−narr> Los documentos comentan tanto la discusión sobre la
decisión de la ONU de enviar las tropas americanas a Haití, como la
invasión misma. Se habla también de sus consecuencias directas. </ES−narr>
</top>
```

Figura 2. Una consulta de la colección.

1.2. Lenguaje de programación

En la primera participación en CLEF de nuestro grupo de trabajo, CLEF−2000 [Figuerola et al., 2001a] se utilizó un motor de recuperación de información para uso experimental y educativo [Figuerola et al., 2001b]. En esta participación hemos reconstruido el motor utilizando para ello el lenguaje de programación Perl [Wall, 2002].

El motivo ha sido la facilidad en el tratamiento de cadenas de texto que tiene este lenguaje, aunque en detrimento de la velocidad. Asimismo no se ha utilizado ningún gestor de bases de datos para el almacenamiento de los términos índice, sino que se han utilizado ficheros de texto plano. Ello nos ha permitido extender las posibilidades educativas de nuestro sistema. En varias de las subrutinas se utilizan expresiones regulares al estilo POSIX, por ello se requiere versión 5.6.0 o superior de Perl.

En este informe serán frecuentes las referencias al lenguaje Perl.

2. La Recuperación de Información

La Recuperación de Información tienen sus orígenes en las bibliotecas y centros de documentación en los que se requerían búsquedas bibliográficas de libros y artículos de revista. El objetivo principal de cualquier centro de documentación es satisfacer las necesidades reales y potenciales de información de todos los usuarios, proporcionándoles la información veraz, pertinente, justo a tiempo y al menor coste.

Debido a motivos históricos, los documentos en esos centros se representan utilizando un conjunto de términos índice o palabras clave. Existen fichas (manuales o electrónicas) en las que se rellenan los campos apropiados con esa información. En esos campos se incluyen datos como el título, autor, fecha de publicación, etc., del documento en cuestión. Pero también se incluyen otros términos que dan una indicación de su contenido, y que normalmente quedan reflejados en el campo *materia*. Uno o varios especialistas asignan la materia de acuerdo a criterios más o menos subjetivos.

Los usuarios que consultan el sistema de recuperación para buscar información deben traducir su necesidad informativa en una consulta adecuada al sistema de recuperación. Esto supone utilizar un conjunto de términos que expresen semánticamente su necesidad. En sistemas tradicionales también es habitual utilizar operadores booleanos para conectar varios criterios de búsqueda por campos diferentes.

El objetivo de la recuperación de información es, dada una necesidad de información y un conjunto de documentos, ordenar los documentos de más a menos relevantes para esa necesidad y presentarlos al usuario. Para ilustrar el problema nos ayudamos de la figura siguiente (adaptada de [Rijsbergen, 1979]).

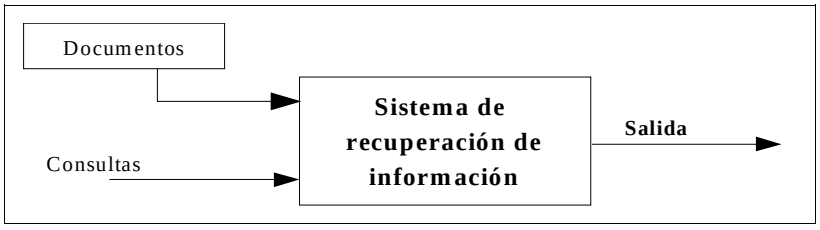


Figura 3. Sistema de recuperación de información.

En esa figura se ha supuesto que el sistema de recuperación de información es una caja negra que acepta documentos y consultas, y que obtiene una salida, que es el conjunto de documentos que satisfacen la consulta. El problema principal en este sistema es obtener representaciones homogéneas de documentos y consultas, y procesar convenientemente esas representaciones para obtener la salida. De otro lado, también es muy importante la evaluación de la salida, para determinar si ésta coincide con las necesidades informativas del usuario.

En el proceso de recuperación de información se suelen distinguir las siguientes etapas:

1. Obtener representación de los documentos. Generalmente los documentos se presentan utilizando un conjunto más o menos grande de términos índice. La elección de dichos términos es el proceso más complicado.
2. Identificar la necesidad informativa del usuario. Se trata de obtener la representación de esa necesidad, y plasmarla formalmente en una consulta acorde con el sistema de recuperación.
3. Búsqueda de documentos que satisfagan la consulta. Consiste en comparar las representaciones de documentos y la representación de la necesidad informativa para seleccionar los documentos pertinentes.
4. Obtención de resultados y presentación al usuario.
5. Evaluación de los resultados por parte del usuario.

Con el incremento del número de documentos en formato electrónico, se hace necesario contar con herramientas informáticas adecuadas para la recuperación de documentos. Las técnicas manuales han demostrado ser ineficaces, pues básicamente consisten en la elaboración manual de una descripción del contenido temático de cada uno de los documentos, siendo éste un trabajo costoso en tiempo, que además tiene abundante inconsistencia [Hooper, 1965; Stubbs et al., 2000].

Actualmente los ordenadores hacen posible representar un documento utilizando su contenido completo. Es lo que se denomina representación a texto completo del documento. Efectivamente es la forma más completa de representar un documento, pero implica un coste computacional muy alto para colecciones grandes de documentos.

En estos sistemas cada documento se puede representar por todas sus palabras, tanto nombres, como verbos, adjetivos, adverbios, etc. A pesar de ello, no todos los términos poseen la misma utilidad para describir el contenido de un documento. De echo, hay términos más importantes que otros, pero no es tarea fácil decidir la importancia de cada término. Por ejemplo, desde el punto de vista de la recuperación de información existen palabras casi vacías de contenido semántico, como los artículos, preposiciones o conjunciones, que parecen poco útiles en el proceso. Sin embargo, algunos estudios actuales en el campo de la Lingüística indican que el estudio de los artículos incorpora un nivel semántico adicional al nombre que acompañan, y por tanto quizás debieran tenerse en cuenta.

Independientemente de ello, tampoco es útil para las tareas de recuperación de información aquellos términos que se repiten con mucha frecuencia en toda la colección de documentos, pues son términos poco discriminatorios en relación con una consulta dada. En conclusión, en estos sistemas no sólo se persigue encontrar aquellos términos que mejor representen a los documentos, sino además aquellos que permitan diferenciar unos respecto de otros.

Por otra parte, un tercer tipo de sistemas permiten realizar búsquedas conocidas como en texto libre. En éstos se realizan búsqueda de subcadenas, sobre el texto almacenado de todo el documento. Evidentemente, el coste de almacenamiento y computacional en la búsqueda es elevadísimo, y además, muchas veces se muestran incapaces de resolver dos problemas básicos: la sinonimia y la polisemia. Por ello, la investigación en recuperación de información busca diseñar sistemas que acepten consultas en lenguaje natural y proporcionen documentos adecuados a tales consultas, ordenados según algún criterio del sistema, de acuerdo a las características de los documentos y a las necesidades informativas expresadas por el usuario en su consulta [Belkin y Croft, 1987]. Uno de los modelos más conocido y difundido para el sistema de recuperación es el llamado modelo vectorial, que pasamos a describir.

3. El modelo vectorial

El modelo vectorial fue definido por Salton [Salton, 1968] hace ya bastantes años, y es ampliamente usado en operaciones de RI, así como también en operaciones de categorización automática, filtrado de información, etc. En el modelo vectorial se intenta recoger la relación de cada documento D_i, de una colección de N documentos, con el conjunto de las m características de la colección. Formalmente un documento puede considerarse como un vector que expresa la relación del documento con cada una de esas características.

$$D_i \rightarrow \vec{d}_i = (c_{i1}, c_{i2}, \ldots, c_{im}) \tag{1}$$

Es decir, ese vector identifica en qué grado el documento D_i satisface cada una de las m características. En ese vector, c_{ik} es un valor numérico que expresa en qué grado el documento D_i posee la característica k. El concepto 'característica' suele concretarse en la ocurrencia de determinadas palabras o términos en el documento, aunque nada impide tomar en consideración otros aspectos.

Si se consideran los términos como características definitorias del documento, el proceso que debe seguir el sistema pasa primero por seleccionar aquellos términos útiles que permitan discriminar unos documentos de otros. En este punto debemos señalar que no todas las palabras contribuyen con la misma importancia en la caracterización del documento. Desde el punto de vista de la recuperación de información existen palabras casi vacías de contenido semántico, como los artículos, preposiciones o conjunciones, que son poco útiles en el proceso.

Pero también son poco importantes aquellas palabras que por su frecuencia de aparición en toda la colección de documentos pierden su poder de discriminación. En RI todas ellas forman parte del conjunto de palabras vacías (stops words en inglés), que se eliminan en el proceso de indexación. Además de la eliminación de palabras vacías, en el proceso se pueden incluir aplicaciones léxicas como lematización (ver apartado 5.3) o extracción de raíces, etiquetado de términos, detección de unidades multipalabra, etc.

Una vez seleccionado el conjunto de términos caracterizadores de la colección de documentos, es necesario obtener el valor de cada elemento del vector del documento. El caso más simple es utilizar una aproximación binaria, de forma que si en el documento D_i aparece el término k, el valor c_{ik} sería 1, y en caso contrario sería 0.

No obstante, una palabra puede aparecer más de una vez en el mismo documento, y además, unas palabras pueden considerarse con más peso, esto es, más significativas que otras, de forma que el valor numérico de cada uno de los componentes del vector obedece normalmente a cálculos más sofisticados que la simple asignación binaria. De otro lado, también es importante normalizar los vectores para no privilegiar documentos largos frente a otros documentos menos extensos.

$$\vec{d}_i = \frac{1}{\sqrt{\sum_{j=1}^{m} w_{ij}^2}} (w_{i1}, w_{i2}, \ldots, w_{im}) \tag{2}$$

Se han propuesto diversos métodos para calcular el peso de cada término en el vector documento [Salton y McGill, 1983; Salton y Buckley, 1988; Harman, 1992a], pero en general, para estimarlos se parte de dos ideas en cierto sentido contrapuestas: si un término aparece mucho en un documento, es importante para caracterizar ese documento. Pero si aparece en muchos documentos de la colección, no es beneficioso para distinguir un documento de los demás, dado su escaso poder discriminatorio, resultando poco útil para la recuperación.

Para determinar la capacidad de representación de un término para un documento dado se computa el número de veces que aparece en dicho documento, obteniéndose la frecuencia del término en el documento, *tf* (*term frequency*).

Por otra parte, si la frecuencia de un término en toda la colección de documentos es extremadamente alta, se opta por eliminarlo del conjunto de términos de la colección (pertenece al conjunto de palabras vacías). Podría decirse que la capacidad de recuperación de un término es inversamente proporcional a su frecuencia en la colección de documentos. Esto es lo que se conoce como *idf* (*inverse document frequency*).

Así, para calcular el peso de cada elemento del vector que representa al documento se tiene en cuenta la frecuencia inversa del término en la colección, combinándola de alguna forma con la frecuencia del término dentro de cada documento. Normalmente se utiliza para ello el producto simple [Harman, 1992].

$$w_{ij} = tf_i \cdot idf_j \tag{3}$$

Salton y Buckley [Salton y Buckley, 1988] experimentaron con más de 200 sistemas de cálculo de pesos, pero uno de los más utilizados viene dado por la ecuación 4, que expresa el peso del término *j* en el documento *i*.

$$w_{ij} = tf_{ij} \cdot \log \frac{N}{df_j} \tag{4}$$

donde df_j es el número de documentos en que aparece el término *j*, y *N* el número de documentos de la colección.

El proceso realizado para los documentos también puede aplicarse a las consultas. Efectivamente, una consulta, *Q*, realizada en lenguaje natural está formada por términos, y, por tanto, puede verse como un documento más, seguramente bastante breve, aunque no siempre. Así pues, el mecanismo de obtención de pesos también se aplica a las consultas, para de esta manera poder disponer de representaciones homogéneas de consultas y documentos, que posibiliten obtener el grado de similitud entre ambas representaciones.

El vector representante de la consulta está formado por un vector de igual número de elementos que los vectores de los documentos. Cada elemento de ese vector expresa el grado en que cada uno de los términos de la colección representa las necesidades informativas de la persona que hace la consulta.

$$Q \rightarrow \vec{q} = \frac{1}{\sqrt{\sum_{j=1}^{m} p_j^2}} (p_1, p_2, \ldots, p_m) \tag{5}$$

La resolución de la consulta consiste en un proceso de establecer el grado de semejanza entre el vector consulta y el vector de cada uno de los documentos. Para una consulta determinada, cada documento arrojará un grado de similitud determinado; aquéllos cuyo grado de similitud sea más elevado se ajustarán mejor a las necesidades expresadas en la consulta, desde el punto de vista del sistema de recuperación de información. No obstante, es el usuario el que debe decidir la relevancia de los documentos recuperados, siendo ésta una característica totalmente subjetiva del mismo.

El modo más simple de calcular la similitud entre una consulta y un documento, utilizando el modelo vectorial, es realizar el producto escalar de los vectores que los representan (ecuación 6). En esa ecuación se incluye la normalización de los vectores, a fin de obviar distorsiones producidas por los diferentes tamaños de los documentos. El índice de similitud más utilizado es el coseno del ángulo formado por ambos vectores. Para una consulta *Q*, el índice de similitud con un documento D_i es:

$$SIMIL(Q, D_i) = \frac{\sum_{j=1}^{m} p_j d_{ij}}{\sqrt{\sum_{j=1}^{m} p_j^2 \cdot \sum_{j=1}^{m} d_{ij}^2}} \quad (6)$$

Hay otros métodos propuestos para calcular la similitud; un cuadro con los más importantes puede encontrarse en [Salton y McGill, 1983]. Los resultados de la computación del índice de similitud entre la consulta y todos los documentos permite ordenar los resultados en orden decreciente. De esta manera se le ofrecen al usuario primero los documentos que el sistema de recuperación considera más similares con la consulta, y que pueden coincidir, o no, con lo esperado por el usuario. La relevancia es la medida subjetiva que el usuario tiene para determinar si los resultados, y en qué grado, son adecuados a sus necesidades informativas.

4. Evaluación de la recuperación

La calidad de un sistema de recuperación de información está determinada por los resultados comparativos entre documentos recuperados y documentos relevantes para una consulta dada. Existen otras medidas de calidad referentes a velocidad de procesamiento y espacio de almacenamiento, pero en nuestro caso hemos obviado estas medidas, centrándonos en la evaluación de la recuperación. Debemos indicar, asimismo, que no analizamos la evaluación de sistemas interactivos, por el contrario, solamente se estudian los aspectos relacionados con sistemas que podemos denominar por lotes, es decir, sistemas en los que se plantea una consulta y se analizan los resultados de la misma, sin considerar las operaciones posteriores del usuario.

La evaluación de un sistema de recuperación se realiza utilizando una colección de pruebas (*test collection*) perfectamente caracterizada. Esta colección consiste en un conjunto de documentos y un conjunto de consultas. Para cada una de las consultas varios especialistas han seleccionado los documentos pertinentes de la colección. Esos documentos serán los documentos relevantes para la consulta dada. La evaluación de un sistema de recuperación consiste entonces en comparar, para cada una de las consultas de la colección de pruebas, los documentos que el sistema ha obtenido, esto es, los documentos recuperados, y los documentos marcados como relevantes para esa consulta.

El método más habitual para medir la calidad de un sistema es la utilización de **diagramas precisión−exhaustividad** (*precision−recall*; la traducción de *recall* no es uniforme en español y muchos autores no lo traducen. Nosotros utilizaremos *exhaustividad* porque es la palabra que creemos mejor se acerca a su significado). Tomemos una consulta concreta de la colección de pruebas y el conjunto de documentos relevantes para dicha consulta. Sea |*b*| número de documentos relevantes. Apliquemos esa consulta al sistema que se desea evaluar. Para esa consulta se ha recuperado un conjunto de documentos. Sea |*a*| el número de documentos recuperados. La figura 4 ilustra estos conjuntos.

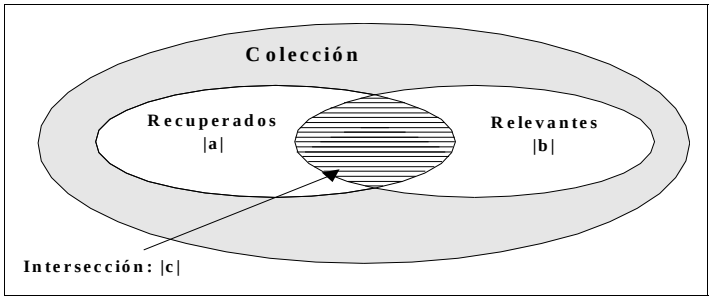


Figura 4. Precisión−exhaustividad para una consulta de ejemplo.

Las medidas de precisión y exhaustividad se definen como:

- Precisión: determina cuántos documentos recuperados son relevantes.

$$Precisión = \frac{|c|}{|a|}$$

- Exhaustividad: determina cuántos documentos relevantes se han recuperado.

$$Exhaustividad = \frac{|c|}{|b|}$$

Estas medidas de precisión y exhaustividad suponen que se han examinado todos los documentos recuperados. Sin embargo, al usuario normalmente no se le presentan todos los documentos a la vez, sino que el sistema se los presenta ordenados utilizando algún criterio interno. El usuario entonces empieza a examinar la lista de documentos ordenados, empezando por el primero. En este proceso las medidas de precisión y exhaustividad van variando con cada uno de los documentos examinados. Para la explicación se ha seguido más o menos fielmente [Baeza−Yates y Ribeiro−Neto, pp. 76 y ss.].

4.1. Diagramas precisión−exhaustividad no interpolada

Se trata de obtener un diagrama según se van revisando los documentos recuperados por el sistema que son relevantes. Para la explicación nos ayudaremos de un ejemplo. Supongamos que hemos seleccionado una consulta del conjunto de pruebas. Para esa consulta sabemos que hay, por ejemplo, 16 documentos relevantes. Lanzamos ahora al sistema que deseamos evaluar la consulta en cuestión, y se obtienen 20 documentos, ordenados de acuerdo al criterio del sistema de recuperación. Se indican en la tabla siguiente con un círculo los documentos recuperados que son relevantes.

1. •	5.	9.	13. •	17.
2.	6.	10.	14. •	18.
3. •	7. •	11. •	15.	19. •
4.	8. •	12.	16.	20.

Analicemos paso a paso la precisión y exhaustividad para cada documento recuperado que es relevante. Empecemos por el primer documento relevante que se ha recuperado. Es el primer documento, de modo que la precisión será 1 de 1 (1 relevante de 1 recuperado), es decir, del 100%. La exhaustividad es 1 documento relevante de un total de 16, es decir, el 6,25%.

El siguiente documento relevante ocupa la tercera posición, es decir, la precisión es del 66,67 % (2 relevantes de 3 recuperados), y la exhaustividad del 12,50% (2 relevantes de un total de 16) .

El siguiente documento relevante ocupa la séptima posición, la precisión es del 42,86 (3 relevantes de 7 recuperados) y la exhaustividad del 18,75% (3 relevantes de un total de 16). Para el resto de documento relevantes se ha obtenido la siguiente tabla. Por convenio se toma que la precisión se hace cero cuando ya no quedan documentos relevantes en los recuperados.

Exhaust. (%)	**Precisión** (%)
6,25	100,00
12,50	66,67
18,75	42,86
25,00	50,00
31,25	45,45
37,50	46,15
43,75	50,00
50,00	42,11
56,25	0
62,50	0
68,75	0
75,00	0
81,25	0
87,50	0
93,75	0
100,00	0

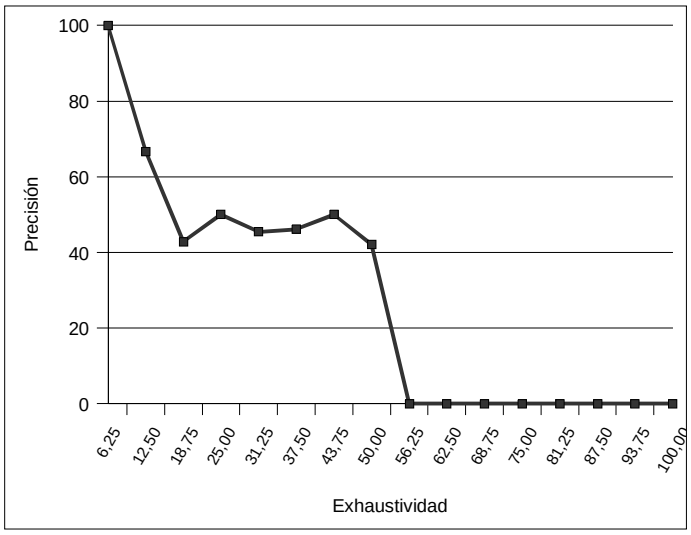


Figura 5. Diagrama precisión−exhaustividad.

A partir del diagrama de precisión−exhaustividad se obtiene la **precisión media no inter−polada**, sin más que calcular la media de la columna de precisión. En el ejemplo, su valor es 0,2770 (no se pone en forma porcentual).

4.2. Diagrama precisión−exhaustividad interpolada

Normalmente se toma interpolación de la precisión para 11 puntos estándar de exhaustividad en los niveles del 0%, 10%, 20%, ..., 100%. La precisión para cada uno de estos niveles se calcula como el máximo valor de precisión entre ese valor y el siguiente. Formalmente, podemos defi−nirlo de esta manera, sea n_j, con $j \in \{0, 1, 2, ..., 10\}$, el nivel estándar j−ésimo de exhaustivi−dad, entonces, el valor de precisión interpolada para ese nivel viene dado como:

$$P(n_j) = max_{n_j \leqslant n \leqslant n_{j+1}} P(n)$$

Puesto que el valor de precisión para cada nivel de exhaustividad interpolada se calcula sobre el valor máximo de precisión entre un nivel y el siguiente (por ejemplo, para calcular el valor interpolado de precisión para el nivel del 50% se observan los valores de precisión en el rango entre el 50% y el 60%), siempre se debe calcular primero el valor para el 100%.

Calcularemos el valor de precisión interpolada considerando el ejemplo anterior. Para el nivel del 100% de exhaustividad se tiene un valor de precisión del 0%. Para el nivel de exhaus−tividad del 90% se debe tomar el máximo entre el 90% y el 100% de exhaustividad. En nuestro caso son los valores del 93,75% (precisión de valor 0) y del 100% (precisión del valor 0). El valor máximo es 0, que es tomado para el nivel estándar 90% de exhaustividad.

Para los niveles de exhaustividad del 80%, 70% y 60% también se obtiene una precisión interpolada de 0. Calculemos ahora el valor para el nivel del 50%. Debemos tomar los valores entre el 50% y el 60%. En este caso tenemos los valores del 50% (42,11), 56,25% (0) y 60% (0). El máximo es 42,11. Para el nivel del 40% se toman los valores entre 40% y 50%. Tenemos el valor del 43,75 (50%) y el valor del 50% (42,11). El máximo es el 50. Para el nivel del 30%

se toman los valores entre el 30% y el 40%. Tenemos el valor del 31,25 (45,45), 37,50 (46,15) y del 40% (50). El máximo es 50. Para el nivel del 20% se toman los valores entre el 20% y el 30%. Tenemos el valor del 25,00 (50,00) y del 30% (50). El máximo es 50. Para el nivel del 10% se toman los valores entre el 10% y el 20%. Tenemos el valor del 12,50 (66,67), 18,75 (42,86) y del 20% (50). El máximo es 66,67. Para el nivel del 0% se toman los valores entre el 0% y el 10%. Tenemos el valor del 6,25 (100) y del 10% (66,67). El máximo es 100.

Obtenemos la siguiente tabla:

Exhaust (%)	**Precisión** (%)
0	100,00
10	66,67
20	50,00
30	50,00
40	50,00
50	42,11
60	0
70	0
80	0
90	0
100	0

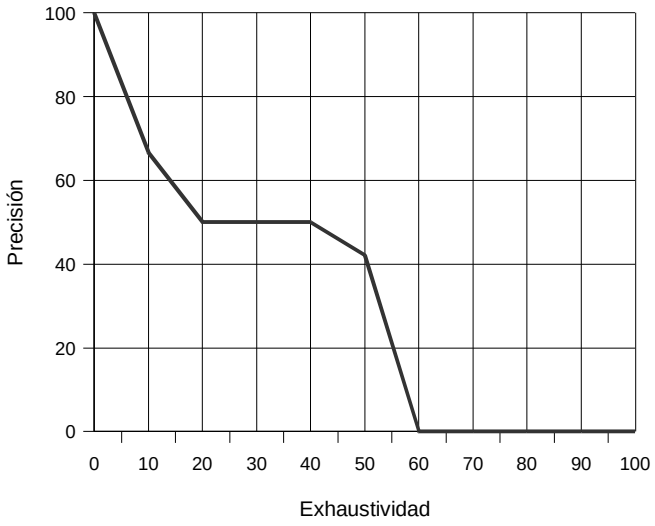


Figura 6. Diagrama precisión−exhaustividad interpolada.

4.3. Precisión a ciertos documentos vistos

Un enfoque adicional que permite ver la calidad del sistema de recuperación consiste en determinar la precisión a ciertos documentos vistos. Por ejemplo, se puede computar ese valor para 5, 10, 15, 20, 30, 50 y 100 documentos vistos. Esta medida permite ver la calidad del algoritmo de ordenación por relevancia del sistema de recuperación. En el ejemplo se tiene:

```
a   5 docs: = 0,4000 (2/5)
a  10 docs: = 0,4000 (4/10)
a  15 docs: = 0,4700 (7/15)
a  20 docs: = 0,4000 (8/20)
a  30 docs: = 0,2700 (8/30)
a  50 docs: = 0,1600 (8/50)
a 100 docs: = 0,0800 (8/100)
```

4.4. Valores individuales

La curvas de precisión−exhaustividad se pueden extender para todo el experimento sin más que calcular la media para todas las consultas de la colección. Ello permite comparar fácilmente las cualidades de dos algoritmos de recuperación.

Sin embargo, muchas veces interesa comparar la calidad de ambos sistemas considerando preguntas individuales. Se pretende entonces obtener valores que resuman el comportamiento de la curva precisión−exhaustividad para una consulta dada. Estos valores se estudian en los apartados siguientes.

4.4.1 Precisión media a ciertos documentos relevantes vistos

Se trata de obtener un valor medio según se van considerando los documentos relevantes, para el total de documentos vistos. Siguiendo el ejemplo de la figura 5, en los 20 documentos vistos se ha encontrado que la precisión, según van apareciendo nuevos documentos relevantes, es 1 − 0,6667 − 0,4286 − 0,5 − 0,4545 − 0,4615 − 0,5 − 0,4211. En este caso, la precisión media es (1 + 0,6667 + 0,4286 + 0,5 + 0,4545 + 0,4615 + 0,5 + 0,4211)/8 = 0,5541.

Esta medida favorece aquellos sistemas que obtienen antes los documentos relevantes. Pero no debe despistarnos, pues un valor alto puede enmascarar valores muy bajos de exhaustividad (por ejemplo, si solamente se recuperan dos documentos relevantes en las posiciones primera y segunda, el valor de esta medida sería alto, pero la exhaustividad sería pequeña).

4.4.2 R−Precisión

Se trata de obtener la precisión para una consulta una vez vistos una cantidad de documentos igual al número de documentos relevantes para esa consulta. En el ejemplo de la figura 5, el número de documentos relevantes para la consulta es 16. La precisión después de ver 16 documentos tiene un valor de 0,4375 (7/16).

Esta medida individual permite observar el comportamiento del algoritmo de recuperación para todas las consultas de la colección de prueba. También se puede obtener una R−Precisión media para toda la colección de consultas, pero puede ser una media poco caracterizadora del sistema.

4.4.3 Histograma de precisión

Se trata de obtener una comparativa entre dos algoritmos de recuperación utilizando la diferencia entre los valores correspondientes de R−Precisión para cada una de las consultas en ambos experimentos. Para cada consulta, *i*, se tiene un valor

$$RP_{A/B}(i) = RP_A(i) - RP_B(i)$$

siendo A y B los experimentos y *RP* la R−Precisión . Un valor positivo de esta medida indica una mejor calidad del sistema A para la consulta *i* en cuanto a R−Precisión. La figura 7 presenta un ejemplo.

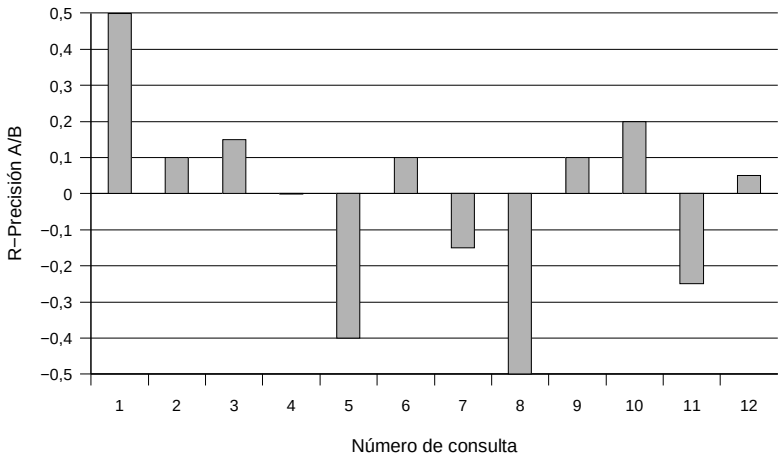


Figura 7. Histograma de precisión para dos experimentos.

4.5. Tablas resumen

Son tablas en las que aparecen los resultados, primero individuales y al final la media, para todas las consultas del experimento. Es habitual encontrarse datos como el número de documentos relevantes, número de documentos relevantes recuperados, valores interpolados de precisión, R−Precisión, precisión a cierta cantidad de documentos vistos, etc. Para nuestro experimento hemos obtenido una tabla de estas características, con sus valores medios finales. En la figura 8 puede verse un ejemplo para la consulta 42.

```
Queryid (Num):       42
Total number of documents over all queries
    Retrieved:     1000
    Relevant:       261
    Rel_ret:        241
Interpolated Recall − Precision Averages:
    at 0.00       1.0000
    at 0.10       1.0000
    at 0.20       0.9836
    at 0.30       0.7714
    at 0.40       0.7086
    at 0.50       0.6157
    at 0.60       0.5358
    at 0.70       0.4197
    at 0.80       0.3746
    at 0.90       0.2652
    at 1.00       0.0000
Average precision (non−interpolated) for all rel docs(averaged over queries)
                  0.6111
Precision:
  At    5 docs:   1.0000
  At   10 docs:   1.0000
  At   15 docs:   1.0000
  At   20 docs:   1.0000
  At   30 docs:   1.0000
  At  100 docs:   0.7700
  At  200 docs:   0.6350
  At  500 docs:   0.3940
  At 1000 docs:   0.2410
R−Precision (precision after R (= num_rel for a query) docs retrieved):
    Exact:        0.5632
```

Figura 8. Ejemplo de tabla resumen para una consulta.

Para terminar este apartado digamos que, aunque los valores de precisión−exhaustividad se consideren como los mejores valores para determinar las cualidades de un sistema, éstos se han aplicado para situaciones en las que se conoce la relevancia de la colección de pruebas, y en situaciones no interactivas, es decir, al sistema se le aplican las consultas de prueba y se analizan los resultados, sin considerar los efectos de la conducta del usuario. En [Baeza−Yates y Ribeiro−Neto, pp. 81 y ss.] pueden consultarse otras medidas para otras situaciones.

5. Preprocesado de texto

Una vez analizado el modelo vectorial y la evaluación de los sistemas de recuperación, es necesario detenerse un momento a estudiar el análisis de los elementos que componen cada vector que representa a documentos y consultas. Hemos dicho que cada elemento del vector expresa en qué grado el documento o consulta posee una característica de la colección. En general, el concepto característica se concreta en la aparición de determinadas palabras o términos en el documento. Ahora bien, el proceso pasa por seleccionar aquellos términos que mejor representen los documentos y que mejor discriminen unos documentos respecto de otros.

Como ya indicamos en un apartado anterior, no todas las palabras de un documento son igual de representativas del contenido del mismo. Por ejemplo, en lenguaje escrito las palabras que actúan gramaticalmente como nombres suelen tener más significado que el resto de palabras, aunque no siempre es así. Así pues, la fase inicial del proceso consiste en analizar el texto

del documento para determinar qué términos pueden utilizarse como términos índice. Esta fase se denomina habitualmente como *preprocesado de texto* de la colección de documentos, y suele llevar la sucesión de los siguientes pasos [Baeza−Yates y Ribeiro−Neto, 1999, cap. 7]:

1. Análisis léxico del texto con el objetivo de determinar el tratamiento que se realizará sobre números, guiones, signos de puntuación, tratamiento de mayúsculas y/o minúsculas, nombres propios, etc.
2. Eliminación de palabras vacías con el objetivo de reducir el número de términos con valores muy pocos discriminatorios para la recuperación. Es una forma de delimitar el número de términos índice como términos índice.
3. Aplicación de lematización sobre los términos resultantes para eliminar variaciones morfo−sintácticas y obtener términos lematizados.
4. Selección de los términos (o grupos de términos) que serán términos índice. Normalmente se realiza sobre la naturaleza sintáctica del término, pues, como ya se ha mencionado, los términos que actúan gramaticalmente como nombres suelen poseer un mayor contenido semántico que verbos, adjetivos o adverbios.
5. Construcción de tesauros, es decir, diccionarios de términos asociados, que permitan expandir la consulta con los términos relacionados.

5.1. Análisis léxico del texto

El preprocesado léxico inicial ha consistido en convertir la cadena de entrada, esto es, el texto de cada documento, en un conjunto de palabras o lemas [Fox, 1992], que puedan servir luego como términos índice.

Es importante determinar qué palabras o lemas van a incluirse en el conjunto de lemas a tratar. Parece claro que cualquier palabra formada por caracteres alfabéticos (formalmente `[:alpha:]` en notación POSIX) debiera incluirse como candidato a término índice. Pero en la colección nos encontramos con otros caracteres (números, signos de puntuación, etc.). Relacionado con ello, un aspecto que debe tenerse muy en cuenta es el idioma de los documentos: la colección en español contiene caracteres como la eñe, vocales acentuadas, símbolos especiales, etc, que forma parte de los términos junto a caracteres ASCII.

Además un aspecto previo en esta fase consiste en identificar cada palabra del texto. Parece claro que el separador por excelencia es el espacio, pero también son separadores los signos de puntuación. En este apartado veremos todas estas cosas.

5.1.1 Separación de palabras

Para separar los términos normalmente se utiliza el carácter espacio y los caracteres de puntuación, pero en Perl (herramienta de programación con el que se ha realizado el preprocesado del texto), se puede utilizar como separador de términos el patrón `\b`.

Si se utiliza el espacio y los caracteres de puntuación, hay que reconocer cada uno de ellos como separadores y obrar en consecuencia. Por ejemplo, si los separadores están en el conjunto `[.,;:!¡¿?()/'"{}]` más el espacio, un texto como "`¡Qué! No. Adiós, María.`" quedaría separado en las siguientes palabras "`Qué No Adiós María`". Ahora bien, muchas veces interesa conservar los separadores (son importantes para determinar dónde empieza y acaba una oración, por ejemplo), y quizás la salida del preprocesado sería conveniente que fuera "`¡ Qué ! No . Adiós , María .`". Después simplemente separamos por espacios y obtenemos todos los elementos (términos y signos de puntuación). El único inconveniente es especificar el conjunto de signos de puntuación, pero no hay mayor problema.

Otra alternativa es utilizar el patrón \b. Pero aquí el problema lo tenemos con obtener todos los caracteres que se incluyen en el patrón \w (carácter alfanumérico más el signo '_'). Es realmente un problema de internacionalización, es decir, considerar la situación local del idioma. Por ejemplo, si no se utiliza especificación del idioma, por defecto se utilizará el ISO C (en este caso el patrón \w básicamente lo constituyen los caracteres alfanuméricos del ASCII de 7 bits), de modo que el ejemplo "¡Qué! NO. Adiós, María." quedaría como "¡ Qu é ! No . Adi ó s , Mar í a .". Si por el contrario, se utiliza el idioma es_ES (para español), se tendrá "¡ Qué ! No . Adiós , María .", pues en el patrón \w se incluyen todos los caracteres alfanuméricos del español.

En nuestro programa hemos utilizado tanto una aproximación como la otra, eso sí, teniendo en cuenta que la colección de documentos y consultas está en español.

5.1.2 Tratamiento de acentos

El tratamiento de vocales acentuadas es también importante. La pregunta es: ¿incluimos acentos como aparecen, o los sustituimos por la vocal no acentuada? Por ejemplo, el buscador Google las separa, y no es lo mismo buscar por 'ángel' que por 'angel'. Otros sistemas de búsqueda menos potentes sustituyen las vocales acentuadas. En la política del analizador debe quedar claro qué se hace. Además en el preprocesado hay varias etapas como la eliminación de palabras vacías o la lematización, que pueden o no tener en cuenta las vocales acentuadas. Asimismo, es importante también determinar si en el almacenamiento de términos índice se utilizan o no las palabras con vocales acentuadas.

En general, las vocales acentuadas incluyen una carga semántica importante a la palabra, pero en recuperación de información no suelen considerarse; el motivo no es otro que el alto grado de errores ortográficos que se cometen con los acentos, por lo que, generalmente, en el preproceso léxico se convierten a vocales no acentuadas aquellas que lo estén en el texto.

En nuestro programa hemos permitido la posibilidad de considerar vocales acentuadas o transformarlas a vocales sin acentos. Para nuestros experimentos hemos optado siempre por convertir a formas no acentuadas. En caso de desear convertir a formas no acentuadas se utilizaría la siguiente transformación:

```
$texto =~ tr[óíáéúüÓÍÁÉÚÜàèìòùÀÈÌÒÙâêîôûÂÊÎÔÛäëïöÄËÏÖåÅãÃõÕ]
           [oiaeuuOIAEUUaeiouAEIOUaeiouAEIOUaeioAEIOaAaAoO]
```

5.1.3 Internacionalización. Caracteres ISO−8859−1 (Latin−1)

En las especificaciones de los documentos SGML (se encuentra en el fichero **efe.dtd** de la colección EFE'94) se tienen las líneas que se indican en la figura 9). Los caracteres del texto son los 128 caracteres del ASCII (ISO 646−1983 IRV) y los 128 segundos caracteres del ISO−8859−1 (ECMA−94 Right Part of Latin Alphabet Nr. 1). La colección está en español.

En los programas de Perl hemos incluido las sentencias apropiadas para manejar la codificación correcta. El procesamiento de datos obliga a tener en cuenta los caracteres del patrón \w, también las funciones para pasar a minúsculas, búsquedas de datos alfanuméricos, ordenaciones, etc. Los programas principales de Perl tienen las siguientes sentencias:

```
use POSIX qw(locale_h);
setlocale(LC_COLLATE, "es_ES.ISO-8859-1");
setlocale(LC_CTYPE, "es_ES.ISO-8859-1");
setlocale(LC_NUMERIC, "C");
use locale;
```

Para las subrutinas basta que contengan la sentencia: `use locale;`

NOTA: el sistema operativo debe soportar la utilización de *locales*.

```
SGML "ISO 8879:1986"
                                        CHARSET
BASESET  "ISO 646-1983//CHARSET
International Reference Version (IRV)//ESC 2/5 4/0"
DESCSET    0  9 UNUSED
           9  2  9
          11  2 UNUSED
          13  1 13
          14 18 UNUSED
          32 95 32
         127  1 UNUSED
BASESET   "ISO Registration Number 100//CHARSET ECMA-94 Right Part of Latin
Alphabet Nr. 1//ESC 2/13 4/1"
DESCSET  128 32 UNUSED
         160 95 32
         255  1 UNUSED
```

Figura 9. Codificación de los documentos.

En la tabla siguiente se estudia la aparición de los caracteres en la colección de documentos, y se indica la acción que se sugiere para el preprocesado léxico de cada uno de ellos.

Dec	Exa	Car	Comentario
0 – 31	0 – 1F		Caracteres de control que no aparecen en la colección.
32	20		Espacio
33	21	!	(cierre de exclamación). Se da en frases exclamativas y también para realizar cuadros clasificatorios. Tener en cuenta a la hora de determinar dónde empieza y acaba una frase para la obtención de nombres propios. No considerar al final del proceso.
34	22	"	(comillas doble). Aparece en textuales (dijo que "...") y para indicar nombres propios, apodos y voces extranjeras (Miguel Porland "Michel", mercados "swap"). Tener en cuenta a la hora de determinar dónde empieza y acaba una frase para la obtención de nombres propios.
35	23	#	(símbolo numérico) Aparece en 8 ocasiones en la colección, 3 son errores. Este término es susceptible de ser por sí mismo término índice, al ser utilizado a menudo (aunque en textos en inglés) como sustituto de la palabra 'número'. En nuestra colección apenas aparece, y no le hemos asignado significado especial.
36	24	$	Aparece en unas 40 ocasiones, normalmente para indicar dólares. También aparece como error tipográfico y en el fichero 19941225 (que posee errores). Este término es susceptible de ser por sí mismo término índice, al ser utilizado a menudo (normalmente en documentos económicos o financieros) como sustituto de la palabra 'dólar'. En nuestra colección apenas aparece, y no le hemos asignado significado especial.
37	25	%	Aparece en porcentajes 2721 veces en la colección, junto a un número (96,8%) o separado de él (96,8 %). Este término es susceptible de ser por sí mismo término índice, pero en nuestro caso no le hemos asignado significado especial. Puede aparecer también en rutas de Internet (http://www/%7Ecarlos/pp.htm), o como forma hexadecimal (%7F) pero aquí no lo tenemos en cuenta.
38	26	&	Aparece en 5 ocasiones como error tipográfico (también en el fichero 19941225). Este término es susceptible de ser por sí mismo término índice, pero no le hemos asignado significado especial.

Dec	Exa	Car	Comentario
39	27	’	(comilla simple) Aparece junto a números (Mundial’93, sub’21), limitando nombres y apodos (’El País’, ’Chendo’, ’bakalao’), en literales (dijo que ’...’), formando parte de nombres propios (L’Enfantillage, d’Orsay, D’Aubuisson, N’sang, Bar’am, Aujourd’hui), indicando segundos (30’), en otros idiomas (inglés: I’m, What’s, Madigan’s, Maxim’s; francés: E’lui, l’infante), para separar decimales (37’5) y en siglas en plural (PC’s, ONG’s) Podría considerarse la situación especial dentro de nombres propios y siglas (N’sang, PC’s) o como separador decimal en números (37’5), y actuar en consecuencia, pero nosotros no lo tendremos en cuenta. Es decir, en el proceso se eliminará o se sustituirá por un espacio. Solamente lo tendremos en cuenta a la hora de determinar dónde empieza y acaba una frase para la obtención de nombres propios.
40 41	28 29	( )	(apertura y cierre de paréntesis). Pudieran servir como limitadores de frases para marcar nombres propios, pero no lo hemos tenido en cuenta.
42	2A	*	(asterisco). No aparece en la colección, salvo en la etiqueta <TIME>*****</TIME> en 10 documentos. No tienen significado especial diferente al de signo de puntuación.
43	2B	+	(suma) Aparece en Canal+, sumas, y también en vez de ’ y " en literales (fue un +ajuste de cuentas+) y nombres propios (+Chendo+). No tendría significado especial salvo en el último caso. Así pues, es importante a la hora de delimitar frases en el procesado de nombres propios. Lamentablemente hay nombres propios que incluyen el carácter, como Canal+, C++, etc., pero en nuestro caso no lo incluimos como carácter válido para formar términos índice. Puede aparecer también en CGI de Internet (http://www/ccc.cgi?uno+tres), pero aquí no lo tenemos en cuenta.
44	2C	,	(coma) No tiene significado especial.
45	2D	−	(guión) Aparece en inicio de noticia (Malabo, 31 dic (EFE).−), epígrafe de fechas históricas (1833.− Los Reyes), referencia textual (−afirmo Obiang−), epígrafe general (− Hombre (23)), separador (−−−−−−−−−−−−−), separador de fechas y años (01/01/02−07/94, 1978−1981), resultados encuentros deportivos (0−2), nombres compuestos (Kai−chek, Qianhai−Tíbet, París−Dakar−París, Panamá−Cuba−Perú, Mazar−E−Sharif), palabras compuestas (semi−cantón, centro−derecha, ex−ministro, ex−coronel, Nacional−VI), encuentros deportivos (Milán−Sampdoria), siglas compuestas (ITAR−TASS, PSE−EE, MS−DOS), deporte nacionalidad−equipo (ESP−ciclismo, ESP−Kelme), nombre −sigla (Israel−OLP), como parte de nombre propio (cadena France−2, Túpolev−154), carreteras (AS−29, N−620), siglas con números (TVE−1, AX−25, START−1), matrículas (SA−0992−G), y en algunos casos más. En la mayoría de casos el criterio debería ser ignorar el guión, que al final conlleva eliminarlos o sustituirlo por un espacio. La situación menos favorable se da cuando intervienen números y letras (France−2, N−620), pues frecuentemente los encontramos separados (France 2, N 620) y juntos (France2, N620). El criterio que se ha tomado es separarlos también (por ejemplo, TVE−1 => TVE − 1). Nota: también se emplea el guión para separar sílabas al final de línea en textos, en vez del guión de separación (orden 173 en Latín−1). En nuestra colección de documentos no se da esta situación, de modo que no la tenemos en cuenta.
46	2E	.	(punto) Aparece como separador de oraciones, y como tal, debe tenerse en cuenta en la detección de nombres propios. También aparece en siglas (EE.UU.). El tratamiento de nombres propios y siglas se analiza más adelante. También aparece en informática en nombres de fichero (command.com), o en direcciones de Internet (193.146.221.123, alcotan.usal.es) pero aquí no lo tenemos en cuenta.

Dec	Exa	Car	Comentario
47	2F	/	(barra inclinada) Aparece como separador de fechas, cursos (1999/2000), años (94/95), leyes (3/1993), fracciones (1/2), versos, resultados (88/92). Algunas veces separa términos similares o contrapuestos (si/no, no sabe/no contesta). A veces forma parte de un nombre (AS/400, PS/2) o sirve de separador de siglas (TCP/IP, PSOE/PNV/EE). También se utiliza como separador de rutas en Internet (http://milano.usal.es/afzazo/index.htm), pero aquí no lo tenemos en cuenta. En general no hay que tener un trato especial con este carácter, salvo en casos my excepcionales de nombres propios. En nuestro caso no hemos considerado esta situación, de modo que no se considera en el proceso.
48 – 57	30 – 39	0 – 9	Dígitos numéricos.
58	3A	:	(dos puntos) No tiene un significado especial, salvo como separador de frases para la detección de nombres propios. Puede aparecer también en rutas de archivo (C:\datos) o en rutas de Internet (http://www.usal.es), pero aquí no lo tenemos en cuenta.
59	3B	;	(punto y coma) No tiene un significado especial, salvo como separador de frases para la detección de nombres propios.
60	3C	<	(menor que) Salvo en las etiquetas SGML, solo aparecen en errores en dos líneas de toda la colección. No posee significado especial.
61	3D	=	(igual) Aparece en cuentas y como separador. No posee significado especial.
62	3E	>	(mayor que) Salvo en las etiquetas SGML, solo aparecen en errores en dos líneas de toda la colección. No posee significado especial.
63	3F	?	(interrogación) No tiene un significado especial, salvo como separador de frases para la detección de nombres propios. Puede aparecer también en CGI de Internet (http://www/ccc.cgi?uno+tres), pero aquí no lo tenemos en cuenta.
64	40	@	(arroba) En la colección solamente aparece en errores en 4 líneas. Nosotros no lo tendremos en cuenta. Este término aparece en direcciones de correo electrónico, pero no lo hemos consi−derado.
65 – 90	41 – 5A	A – Z	Letras ASCII mayúsculas (A−Z)
91	5B	[	(corchete) Aparece en errores en dos líneas. No posee significado especial.
92	5C	\	(barra atrás) Aparece unas 100 veces como separador. No lo hemos considerado. Puede aparecer también en rutas de archivo (C:\datos), pero aquí no lo tenemos en cuenta.
93	5D	]	(corchete) Aparece en errores en dos líneas. No posee significado especial.
94	5E	^	(acento diacrítico) No aparece. No posee significado especial.
95	5F	_	Aparece unas 70 veces, la mitad como error en vez de '−', el resto sin significado. No posee significado especial, pero es el carácter elegido como marcador de nom−bres propios y siglas en el proceso (por ejemplo, _ABC_, _Juan_de_la_Cruz_), pues está incluido en el patrón `\w` de Perl. Debido a que aparece muy pocas veces y normalmente sustituyendo al guión, hemos decidido sustituir este carácter por el guión al inicio del proceso.
96	60	‘	Sólo aparece en errores en cuatro líneas. No posee significado especial.

Dec	Exa	Car	Comentario
97 – 122	61 – 7A	a – z	Letras ASCCI minúsculas (a−z)
123	7B	{	No aparece en la colección. No posee significado especial.
124	7C	|	Sólo aparece en errores en dos líneas. No posee significado especial.
125	7D	}	No aparece en la colección. No posee significado especial.
126	7E	~	No aparece en la colección. No posee significado especial. Puede aparecer también en rutas de Internet (http://www/~carlos/pp.htm), pero aquí no lo tenemos en cuenta.
127 – 159	7F – 9F		Caracteres de control que no aparecen en la colección.
160	A0		(no−break space). No aparece en la colección. No tener en cuenta.
161	A1	¡	(apertura exclamación). Se da en frases exclamativas. Tener en cuenta a la hora de determinar dónde empieza y acaba una frase para la obtención de nombres propios.
162	A2	¢	(centimo) No aparece. Este término es susceptible de ser por sí mismo término índice, pero no lo hemos considerado.
163	A3	£	(libra) No aparece. Este término es susceptible de ser por sí mismo término índice, al ser utilizado como sustituto de la palabra 'libra', pero no lo hemos considerado.
164	A4		(moneda) No aparece en la colección. Este término es susceptible de ser por sí mismo término índice, al ser el símbolo de moneda, pero no lo hemos considerado.
165	A5	¥	(Yen). No aparece en la colección. Este término es susceptible de ser por sí mismo término índice, al ser utilizado como sustituto de la palabra 'yen', pero no lo hemos considerado.
166	A6		(barra partida). Aparece en un error. No posee significado especial.
167	A7	§	(sección) Aparece en 3 líneas (ampliaci§n, poblaci§n, y 1 línea mal). No se consi−dera en el proceso. Puede tener el significado de sección, pero nosotros no lo hemos considerado.
168	A8		(diéresis). No aparece. No posee significado especial.
169	A9	©	(copyright) Aparece en errores en 1 línea. Puede tener significado por sí mismo, pero no lo hemos considerado.
170	AA	ª	(ordinal femenino) No aparece en la colección. Pudiera aparecer en ordinales, (1ª, 2ª...), o en abreviaturas (Mª, Hª). Si se utiliza *setlocale* con es_ES en Perl, este carácter está incluido en \w.
171	AB	«	(doble flecha izquierda). No aparece. Es un símbolo utilizado igual que la comilla, y por tanto, debiera tenerse presente como separador de frases para detectar nombres propios.
172	AC	¬	(negación) Aparece en errores en 1 línea. Puede tener significado por sí mismo, pero no lo hemos considerado.
173	AD	-	(guión blando, separador de sílabas). No aparece en la colección. Si apareciera, debería realizarse un tratamiento adecuado para unir las sílabas que separase.
174	AE	®	(marca registrada) No aparece. Puede tener significado por sí mismo, pero no lo hemos considerado.
175	AF	¯	(MACRON) No aparece. No tiene significado especial.

Dec	Exa	Car	Comentario
176	B0	°	(grado) Aparece en errores en una línea. Pude tener significado por sí mismo, pero no lo hemos considerado.
177	B1	±	(más−menos). No aparece. No tiene significado especial.
178	B2	²	(superíndice 2). No aparece. No tiene significado especial.
179	B3	³	(superíndice 3). No aparece. No tiene significado especial
180	B4		(acento agudo). No aparece. No posee significado especial.
181	B5	µ	(micro) Aparece en errores en 1 línea. Pude tener significado por sí mismo, pero no lo hemos considerado.
182	B6	¶	(PILCROW − calderón) No aparece. Puede tener significado por sí mismo, pero no lo hemos considerado.
183	B7	·	(punto centrado) No aparece. No posee significado especial.
184	B8		(cedilla). No aparece. No posee significado especial.
185	B9	¹	(superíndice 1). No aparece. No tiene significado especial.
186	BA	º	(ordinal masculino) Aparece en una línea (2º). Si se utiliza *setlocale* con es_ES en Perl, este carácter está incluido en \w. Nosotros no lo consideramos.
187	BB	»	(doble flecha derecha). No aparece. Es un símbolo utilizado igual que la comilla, y por tanto, debiera tenerse presente como separador de frases para detectar nombres propios.
188 189 190	BC BD BE		(fracción 1/4, 1/2 y 1/4). No aparecen. Pueden tener significado por sí mismos, pero no los hemos considerado.
191	BF	¿	(apertura interrogación). Se da en frases interrogativas. Tener en cuenta a la hora de determinar dónde empieza y acaba una frase para la obtención de nombres propios.
192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214	C0 C1 C2 C3 C4 C5 C6 C7 C8 C9 CA CB CC CD CE CF D0 D1 D2 D3 D4 D5 D6	À Á Â Ã Ä Å Æ Ç È É Ê Ë Ì Í Î Ï Ð Ñ Ò Ó Ô Õ Ö	(acento grave, agudo, circunflejo, tilde, diéresis, círculo, ligadura AE, C cedilla, ETH (Ð), eñe). No aparecen o lo hacen en errores en muy pocas líneas. Ç aparece en 4 ocasiones en BARÇA. La eñe aparece habitualmente. Los acentos agudos aparecen normalmente. Todos estos caracteres están en el patrón \w de Perl si se utiliza *setlocale* con es_ES.
215	D7	×	(aspa). No aparece. No tiene significado especial.

Dec	Exa	Car	Comentario
216	D8	Ø	(O barra) (acento grave, agudo, circunflejo, diéresis, THRON (Þ), S SHARP (ß), ligadura ae (æ), c cedilla, eth (ð), eñe).
217	D9	Ù	
218	DA	Ú	No aparecen o aparecen en errores en 1 ó dos líneas. Ü aparece normalmente, así como la eñe, y los acentos agudos.
219	DB	Û	
220	DC	Ü	Todos estos caracteres están en el patrón \w de Perl si se utiliza *setlocale* con es_ES.
221	DD	Ý	
222	DE	Þ	
223	DF	ß	
224	E0	à	
225	E1	á	
226	E2	â	
227	E3	ã	
228	E4	ä	
229	E5	å	
230	E6	æ	
231	E7	ç	
232	E8	è	
233	E9	é	
234	EA	ê	
235	EB	ë	
236	EC	ì	
237	ED	í	
238	EE	î	
239	EF	ï	
240	F0	ð	
241	F1	ñ	
242	F2	ò	
243	F3	ó	
244	F4	ô	
245	F5	õ	
246	F6	ö	
247	F7	÷	No aparece. No tiene significado especial.
248	F8	ø	(o barra, acento grave, agudo, circunflejo, diéresis, thron (þ)).
249	F9	ù	
250	FA	ú	No aparecen o aparecen en errores en 1 ó dos líneas. ü aparece normalmente y también los acentos agudos.
251	FB	û	
252	FC	ü	Todos estos caracteres están en el patrón \w de Perl si se utiliza *setlocale* con es_ES.
253	FD	ý	
254	FE	Þ	
255	FF	ÿ	

Tabla 1. Análisis de los caracteres de la colección.

5.1.4 Tratamiento de números

Generalmente los números no se incluyen como términos índice, salvo que la colección de documentos sea legislativa (leyes, normas), de patentes, o en general, si la información numérica es de importancia. Sin embargo pudieran ser apropiados algunas veces cuando nos referimos a años, aunque para ello suele ser habitual incluir un campo explícito en la base de datos. No obstante hemos comprobado que casi todos los motores de búsqueda en Internet los indexan sin mayor problema.

Si se decide indexar los números, el problema es que el volumen de términos índice se dispara (por ejemplo, si se incluye cualquier número de 6 cifras, potencialmente el número de tér-

minos se incrementa en 1 millón). Hemos comprobado experimentalmente que el número de términos de la colección EFE'94 es de unos 350.000 términos (sin incluir números y antes de realizar lematización), y si incluyéramos números, se incrementarían en un 300%. Sin embargo, los números que se escriben en lenguaje natural suelen llevar separador de miles (por ejemplo, 1.000.000), y podrían tomarse por separado (el '1', el '000' y el '000'). Así, el número de términos correspondiente a datos numéricos se incrementaría solamente en varios miles más.

Sabemos que esto puede introducir cierto ruido en la recuperación, pero al menos se incluye cierta información numérica en la caracterización de la colección. Hemos comprobado que la colección posee 3464 números diferentes, lo cual supone muy poco gasto computacional adicional.

El software desarrollado permite especificar si se desea o no incluir los números como términos índice. En caso de no querer incluir número como términos índice, es decir, si se desea eliminarlos, se deben eliminar aquellos número que puedan contener puntos, comas o comillas (1.000.000, 123,45, 3'75).

Un problema que se tiene con los números es cuando aparecen junto a caracteres alfabéticos en el mismo término (por ejemplo, en sub−21, EEUU'94, B12, PS/2, SA−0992−G, 27−E, 180−R). Existen distintos criterios a la hora de manejar estas situaciones. En el análisis de los caracteres de la colección ya se han comentado las situaciones y las acciones, que en resumen consisten en sustituir el carácter intermedio por un espacio.

5.1.5 Detección de nombres propios

Si importantes para la recuperación de información son los términos que actúan gramaticalmente como nombres, más importantes son las palabras que actúan como nombres propios. Muchas veces todo un documento puede representarse por los nombres propios que contiene, y a menudo se les asigna mayor peso.

En el preprocesado léxico hemos incluido una subrutina que marca los nombres propios en la cadena de texto pasada como argumento, y devuelve el número de nombres propios marcados. La subrutina supone que el texto de entrada está en minúscula, salvo nombres propios o inicio de oración. La subrutina busca las palabras que empiezan con mayúscula, marcándolas como posibles nombres propios. Un nombre propio puede ser simple, puede estar formado por una sucesión de nombres propios simples, o puede formarse con la inclusión de '*de/del*', '*de la/las/los*' e '*y*' entre los nombres propios.

- simples: Madrid, España
- compuestos: José María López, Juan de Austria, Mar del Río, Cereceda de la Sierra, Palacio de los Castro, Juan de las Indias, Construcciones y Contratas

Para ampliar la generalidad de la subrutina, hemos permitido detectar nombres propios separados por la conjunción '*y*', aunque sabemos que apenas se utilizará esta característica, pues se cometen innumerables errores (por ejemplo, en expresiones como 'el acuerdo entre España y Francia...', que tome 'España_y_Francia' como nombre propio).

Para marcar un nombre propio se utilizarán caracteres que se pasan a la subrutina como argumento. Por ejemplo, si los marcadores son '`{}`' el nombre propio se incluirá entre llaves. Un aspecto importante a tener en cuenta: si el nombre propio es compuesto, siempre se intercalará el carácter '_' como separador.

- Madrid, España => `{Madrid}` , `{España}`
- José María López => `{José_María_López}`
- Mar del Río => `{Mar_del_Río}`
- Cereceda de la Sierra => `{Cereceda_de_la_Sierra}`
- Carlos García y López => `{Carlos_García_y_López}`

El criterio para determinar si una palabra es nombre propio es el siguiente: cualquier palabra que empiece por mayúsculas es candidata a ser nombre propio. Esto es cierto si la palabra no está al principio de una oración, que es donde se puede dar la ambigüedad.

El primer paso entonces es dividir el texto de entrada en oraciones. El separador de oraciones por excelencia es el punto, pero es habitual encontrarse con otros separadores, como `;:!¡?¿'"`, o incluso otros (ver el análisis realizado sobre los caracteres de la colección). La subrutina posee un parámetro en el que se indican los separadores admitidos, aunque por defecto siempre se considera el punto.

Una vez dividido el texto en oraciones, es necesario reconocer los nombres propios. Pero existen nombres propios que contienen caracteres especiales como el guión o la comilla (Alí−Ben−Ami, Bar'am). El caso del guión, aunque es muy frecuente, plantea menos problemas, ya que generalmente une palabras que son nombres propios (p.ej. rally París−Dakar). La subrutina simplemente separará las palabras (`rally París − Dakar`).

El caso de la comilla es menos frecuente, pero más problemático, pues aparece en situaciones como (L'Enfantillage, d'Orsay, D'Aubuisson, N'sang, Bar'am, Aujourd'hui), y en palabras de otros idiomas (I'm, Madigan's, Maxim's, What's), francés (E'lui), etc. Existen varios criterios para solucionar esta situación (por ejemplo, separar por la comilla y considerar cada lado si es nombre propio o no y obrar en consecuencia). Sin embargo, en esta versión, el criterio que se considera es simplemente sustituir la comilla por un espacio en blanco (p.ej. d'Orsay => `d Orsay`).

Una vez ha quedado claro que se hace con el guión y la comilla, el paso siguiente es implementar un algoritmo para extraer los nombres propios. Los nombres propios empiezan en mayúscula (en Perl se utiliza la búsqueda al estilo POSIX, p.ej. `[[:upper:]]`, de ahí la necesidad de disponer de Perl v.5.6.0 o superior), y su detección en mitad de la oración no plantea problema. La ambigüedad puede darse si la palabra con mayúscula inicial es la primera palabra de la oración, que puede ser nombre propio o simplemente inicio de oración. Esto se complica porque se deben recoger también nombres compuestos. Veamos algunos ejemplos:

- Los nombres propios.... => '`Los`' no es nombre propio
- La Universidad de... => '`La`' no es parte del nombre propio
- La Coruña tiene... => '`La`' es parte del nombre propio
- Salamanca es la ciudad... => '`Salamanca`' es nombre propio
- Tenemos ahora más... => '`Tenemos`' no es nombre propio
- Oliver Stone presentó... => '`Oliver`' es parte del nombre propio

Para resolver la ambigüedad se necesitan dos diccionarios, uno de nombres propios y otro de palabras de inicio de oración que no pueden ser nombres propios (por supuesto incluirá las palabras vacías). Estos dos diccionarios se darán como argumentos a la subrutina.

Los dos diccionarios deben estar ordenados alfabéticamente (se utiliza la búsqueda binaria). La subrutina considera que los términos del diccionario de nombres no propios están en minúscula. El diccionario de nombres propios contiene tanto nombres propios simples como compuestos; si el nombre propio es compuesto, el carácter de unión será '_'. Tanto unos como otros, aparecerán con mayúscula inicial, por ejemplo: `La_Coruña`.

El proceso que sigue el sistema cuando se detecta un posible nombre propio, por ejemplo, en '`Los hombres de...`', es el siguiente: el algoritmo detectará '`Los`' como posible candidato a nombre propio. Para comprobar si es o no nombre propio se realizan estos dos pasos:

- Chequeo del diccionario de nombres propios. En este caso '`Los`' no está en el diccionario, pero puede ser un nombre propio no recogido en el diccionario, de modo que no podemos asegurar si es o no nombre propio.

- Chequeo del diccionario de nombres no propios. Se pasa a minúscula el término y se comprueba si está en el diccionario. En este caso '`los`' está (es palabra vacía), lo cual indica que no puede ser nombre propio, de manera que no se marca como tal.

Otro ejemplo, cuando se encuentre el texto '`La Coruña tiene...`', el algoritmo detectará como posible candidato a nombre propio '`La_Coruña`'. Se pasa entonces a comprobar si está en el diccionario de nombres propios, en el que efectivamente se encuentra, de modo que '`La_Coruña`' se marca como nombre propio, '`{La_Coruña} tiene...`'.

Veamos otro ejemplo. En el texto '`La Universidad de Jaén posee...`', el algoritmo detectará como posible nombre propio '`La_Universidad_de_Jaén`'. Primero se debe comprobar que está en el diccionario de nombres propios, pero no aparece (seguramente sí estará '`Universidad_de_Jaén`', pero no con el artículo), entonces debemos comprobar que la palabra inicial '`la`' (que es la ambigua en la detección al comienzo de la frase) está en el diccionario de no propios (en minúscula), como efectivamente es, de modo que '`La`' se quita del nombre y se marca el resto como propio: '`La {Universidad_de_Jaén} posee...`'.

Frecuentemente se dará el caso en que la primera palabra de la oración no esté en ninguno de los dos diccionarios. En tal situación, debemos suponer siempre que se trata de un nombre propio no recogido en el diccionario. Existe la posibilidad de ir ampliando el diccionario de nombres propios en este caso.

Vemos que es muy importante la construcción de los dos diccionarios. En particular, en el diccionario de no propios deben estar incluidas las palabras vacías de la colección (*stop words*) (ver el apartado 5.2). En la práctica, es importante un diccionario de nombres propios que empiecen por palabra vacía (`La_Coruña`, `El_Mundo`, `Cabo_Verde`, `Los_Ángeles`).

Nota: no se pueden detectar nombres propios en el campo **TITLE** de la colección de documentos porque su contenido aparece en mayúsculas.

5.1.6 Detección de siglas

La siglas pueden considerarse un caso particular de nombre propio, en el que todos los caracteres aparecen en mayúscula. En el preprocesado léxico hemos incluido una subrutina que marca las siglas en la cadena de texto pasada como argumento. La subrutina devuelve el número de siglas marcadas. Los caracteres empleados como marcadores se pasan como argumento a la subrutina. Por ejemplo, si los marcadores son '`{}`' la sigla quedaría marcada con llaves (`{ONU}`).

Se considera que una sigla está formada por las iniciales de las palabras que representa, separadas por punto, o todas juntas sin punto (excepto cuando se trate de un único carácter en mayúsculas, que no se considera sigla si no lleva punto). La subrutina marcará la siglas como se indica:

```
A.                  => {A}
A.B.        AB      => {AB}         {AB}
A.B.C.      ABC     => {ABC}        {ABC}
A.B.C.D.    ABCD    => {ABCD}       {ABCD}
AA.BB.      AABB    => {AABB}       {AABB}
AA.BB.CC.   AABBCC  => {AABBCC}     {AABBCC}
```

Cuando la sigla no contenga puntos, pero acabe en punto, se tratará de fin de oración, de modo que se conservará éste.

```
AB.         =>  {AB}.
ABC.        =>  {ABC}.
AABB.       =>  {AABB}.
AABBCC.     =>  {AABBCC}.
```

Algunas veces aparecen siglas junto a términos que no lo son. En este caso se ha optado por marcar la sigla, conservando el carácter.

```
F.C.Barcelona  => {FC}. Barcelona
FC.Barcelona   => {FC}. Barcelona
```

En otras ocasiones aparecen varias mayúsculas separadas por punto u otro carácter que no forman sigla. En este caso, se marcan las siglas que corresponda y se mantiene el carácter.

```
ABC.BBB        => {ABC}.{BBB}
ABC-BBB        => {ABC}-{BBB}
```

Hemos detectado varias situaciones en las que las siglas están construidas incorrectamente. Se ha incluido como opción en la subrutina tratar este tipo de siglas. Se explican las situaciones encontradas.

a) Siglas en plural. Muchas veces las siglas van seguidas de una *s* para indicar el plural, con o sin comilla. En este caso se ha acordado marcar la sigla y perder el resto.

```
ONGs       => {ONG}
PC's       => {PC}
```

b) Siglas sin punto final. No es un caso frecuente, pero puede darse. En este caso se ha acordado marcar la sigla completa:

```
A.B        => {AB}
A.B.C      => {ABC}
AA.BB      => {AABB}
AA.BB.CC   => {AABBCC}
```

c) Es frecuente encontrarse con siglas en plural separadas por el carácter espacio. En esta versión se ha acordado marcar la sigla unión.

```
CC OO      => {CCOO}
EE UU      => {EEUU}
CC  AA     => {CCAA}
```

Nota: la subrutina puede erróneamente marcar siglas cuando encuentra 2 ó más mayúsculas juntas separadas de algún otro carácter alfanumérico por un guión, apóstrofe, etc. (por ejemplo, TVE-1 => `{TVE}-1`). Por ello, se recomienda resolver antes la situación de las palabras con guión, comillas, etc.

5.1.7 Almacenamiento en mayúsculas y/o minúsculas

En los sistemas de recuperación de información lo normal es que los términos índice se convier-tan todos a mayúsculas o minúsculas, y así se opere con ellos. No obstante, en situaciones donde interese reconocer nombres propios, siglas o acrónimos es importante marcarlos de alguna manera.

Por ejemplo, en la frase "*Tengo una casa cerca de la Casa Blanca, donde consulto los proyectos de la empresa CASA*", después del preprocesado léxico tendré tres términos:

- `casa`, como nombre común,
- `Casa_Blanca`, como nombre propio, y
- `CASA`, como sigla.

Nosotros hemos tomado el criterio de convertir todos los términos índice en minúscula, salvo aquellos expresamente marcados como siglas o nombres propios. Pero aún queda por decidir cómo se manejan éstos. Por ejemplo, si tenemos el nombre propio `Luis`, ¿cómo se maneja el término índice: `Luis`, `_luis_`, `luis` o de varias formas a la vez?

La respuesta depende del criterio considerado al analizar la consulta:

a) Si la consulta no distingue mayúsculas/minúsculas (es '*case insensitive*'), igual dará que en dicha consulta aparezca *Luis* o *luis*, y por tanto, el término índice que se almacenará para el documento será solamente `luis`.

b) Si la consulta es '*case sensitive*', no es igual que en la misma aparezca *Luis* o *luis*, pues al tratar la consulta, el primero sería nombre propio, y el segundo no. Así pues, se tendría que almacenar el nombre propio como `Luis` o `_luis_` para encontrar los documentos con el nombre propio.

Creemos que la situación correcta es la segunda, ya que en un sistema de recuperación de información que utilice el modelo vectorial las consultas se realizan utilizando lenguaje natural, esto es, con mayúsculas y minúsculas según corresponda.

Un problema que no queda resuelto es la situación de nombres propios compuestos. Por ejemplo, dado un nombre propio como `Villaseco de los Gamitos`, ¿deberían tomarse como término índice `Villaseco_de_los_Gamitos`, o también los nombres simples que componen el complejo (`Villaseco` y `Gamitos`)? Veamos la posibilidades:

a) Si la filosofía es muy restrictiva, se almacenaría solamente el nombre completo `Villaseco_de_los_Gamitos` como término índice, de modo que una consulta sobre `Villaseco` no encontraría documentos sobre `Villaseco de los Gamitos`. La situación se complica en la recuperación si consideramos que en el lenguaje periodístico es habitual referirse a las personas por sus apellidos ('*Aznar*' por '*José María Aznar*').

b) Otra posibilidad consiste en almacenar como términos índice solo los nombres simples (`Villaseco` y `Gamitos`). Una pregunta sobre `Villaseco de los Gamitos` debiera encontrar documentos de `Villaseco de los Gamitos` y también documentos que contuvieran `Villaseco` y `Gamitos`.

c) La tercera posibilidad que nos queda es almacenar tanto el nombre compuesto como los simples. Así, se almacenarían `Villaseco_de_los_Gamitos`, `Villaseco` y `Gamitos` como términos índice. El problema viene ahora al considerar el peso de los nombres que forman parte del nombre compuesto. Si todos se consideran con valor unitario, se habría multiplicado por tres el peso del nombre propio. En este caso, una solución de equidad es dividir la unidad a partes iguales (1/n). Otro criterio es valorar el nombre compuesto con 0,5 y repartir el 0,5 restante entre los nombres simples. Otra solución es asignar 0,5 al compuesto y 0,5 a cada uno de los restantes.

En nuestros experimentos para el taller CLEF−2001 hemos optado por la opción b).

5.1.8 Caracteres de puntuación

Generalmente no se utilizan como parte de términos índice, pero algunas veces deben tenerse en cuenta, como en el caso del guión (ex−alcalde, ruso−japonesa, Bosnia−Herzegovina, Kai−Chek, MS−DOS, TVE−1, N−630, PSE−PSOE), la comilla (Muldial'93, sub'21, D'Aubuisson), la barra inclinada (PS/2), el carácter se subrayado (FS_34), el punto (command.com), etc.

Nosotros no los hemos tenido en cuenta, y al final del preprocesado léxico se eliminan. Para cada carácter ya se han indicado los criterios que se tendrán en cuenta en su tratamiento.

5.2. Eliminación de palabras vacías

Con el objetivo de reducir el número de términos índice no se incluyen palabras que, por su poca capacidad semántica o por su alta frecuencia, son poco significativas en el proceso de recuperación de información. Este conjunto de palabras se compone de preposiciones, artículos, adverbios, conjunciones, posesivos, demostrativos, pronombres, verbos (ser, estar) y algunos nombres, y se denomina conjunto de palabras vacías (*stop words* en inglés)

Con la eliminación de palabras vacías se pretende reducir el ruido que pudieran introducir en la recuperación. Es una forma de delimitar el número de términos que servirán como términos índice. La mayoría de motores de búsqueda en Internet eliminan las palabras vacías de contenido.

Para la eliminación de palabras vacías se ha separado el texto en palabras y luego se han buscado en la lista de palabras vacías, utilizando para ello el algoritmo de la búsqueda binaria. En nuestro caso hemos utilizado una lista de palabras vacías bastante grande, de 505 palabras. Todas ellas están en minúscula y sin acentos.

5.3. Lematización

Se denomina lematización al proceso mediante el cual se buscan variaciones morfológicas de los términos, con el objetivo de extraer la raíz común a ellos. La forma canónica de la raíz se denomina *lema* (aunque el término no está aceptado por la Real Academia de la Lengua), y representa las variaciones de los términos que de ella derivan. La lematización se realiza porque semánticamente los términos que derivan de una raíz están muy próximos entre sí.

Para la lematización hemos utilizado un lematizador desarrollado por nuestro grupo de investigación [Figuerola et al., 2000; Gómez, 2001]. El lematizador detecta variantes tanto flexivas como derivativas de una palabra. Ello implica algoritmos más o menos complejos, desde los que juegan con las variantes de género y número para la lematización flexiva, a los que combinan ésto con una serie de sufijos y prefijos, para la lematización derivativa.

El objetivo de la lematizacion en recuperación de información es reducir el número de términos índice semánticamente muy parecidos. Con ello se pretende aumentar las tasas de exhaustividad del sistema de recuperación, y a la vez reducir el espacio de almacenamiento y aumentar la velocidad del proceso.

Es obvio que las palabras que pertenecen a una misma familia están relacionadas semánticamente, pero estas relaciones no se ponen de manifiesto de manera automática en los sistemas convencionales de RI. Cuando se utiliza el modelo vectorial como algoritmo del sistema de recuperación se trabaja con la frecuencia de aparición de términos en el documento. Los términos se tratan como meros conjuntos de caracteres, y por tanto, un término en singular es distinto al término en plural, lo cual puede provocar perder documentos relevantes. Con ello los métodos basados en frecuencias arrastran un error inherente a su propia construcción. Con la lematización se pretende reducir ese error, de modo que varias palabras con un origen común se puedan reducir a un único término, basándose en que todas guardan relación semántica.

Para un análisis mayor de la lematización le remitimos a las referencias indicadas.

Nuestro lematizador detecta variantes tanto flexivas como derivativas de una palabra. Los experimentos realizados para evaluar nuestro sistema han sido de tres clases, consistentes en ver el comportamiento del sistema sin aplicar lematización, y aplicando luego la flexiva y después la derivativa. Podemos adelantar que los resultados mejores se observaron con la lematización flexiva.

La forma de implementar el proceso lematizador en el sistema es el siguiente. Se obtienen las palabras de toda la colección de documentos, y se pasan *off−line* al lematizador flexivo y derivativo, obteniendo una tabla en cada caso con la palabra origen y lematizada. Esa tabla se utiliza en el programa en Perl implementada como un *hash*. (palabra no lematizada => palabra lematizada). A la hora de aplicar la lematización en el sistema, para cada palabra se obtiene su lematizada y se sigue el procesamiento léxico.

Un tema que parece claro es que los términos marcados como nombres propios o siglas no deben ser lematizados en el proceso. Un apunte más en relación con la eliminación de palabras vacías: hay que aplicar el algoritmo de eliminaron antes y después de la lematización, pues al lematizar algunas palabras pueden reducirse a palabras vacías.

5.4. Tesauro

Un tesauro es un diccionario de términos controlados con relaciones explícitas entre los términos. Por ejemplo, un tesauro geográfico contiene nombres de poblaciones y accidentes geográficos, así como las relaciones entre ellos (por ejemplo, Salamanca estará relacionado con España). Otros tesauros están construidos como diccionarios de sinónimos y/o antónimos.

Normalmente los tesauros se utilizan en recuperación de información para ampliar las consultas, de modo que, además de buscar por los términos que aparecen en la misma, se buscan por aquellos términos relacionados en el tesauro. Normalmente esto produce mejores valores de exhaustividad.

En nuestro sistema de recuperación no se han utilizado tesauros. Su uso se deja para versiones posteriores.

Para finalizar el apartado correspondiente al procesado léxico de texto, debemos indicar que éste se lleva a cabo utilizando una subrutina, que se aplica tanto al texto de los documentos como al de las consultas. Esa subrutina encuentra los términos índice y su frecuencia para cada documento o consulta.

6. Almacenamiento, indexado y búsqueda de información

Después de todo el procesado léxico del texto de los documentos y consultas es necesario almacenar la información en ficheros de bases de datos, indexarlos para aumentar la velocidad e implementar un algoritmo que comparare documentos y consultas para extraer los relacionados. Nuestras bases de datos son ficheros en texto plano.

Para almacenar la información de los documentos hemos utilizado un programa diferente al utilizado para las consultas, aunque con una filosofía de funcionamiento similar. Ambos programas utilizan la subrutina que obtiene los términos índice y su frecuencia.

Para obtener los resultados de similitud entre consultas y documentos se ha utilizado un programa que utiliza el fichero de pesos de las consultas, va tomando consulta por consulta, extrayendo los términos de cada una de ellas, y buscando esos términos en el fichero inverso de documentos. Todos los programas utilizan el mismo archivo de configuración para mantener la uniformidad.

El fichero de configuración posee los siguientes grupos de parámetros:

- Parámetros relativos a la colección de documentos. Aquí se indican las rutas de los ficheros en que se encuentran los documentos, qué identificadores al estilo SGML, se utilizan para marcar cada documento, o cada campo del documento, así como sus ponderaciones, y aspectos similares.
- Parámetros relativos a la colección de consultas. Son los parámetros homólogos para la colección de consultas.
- Parámetros relacionados con el proceso léxico del texto. Aquí se indican los diccionarios de palabras vacías, de nombres propios, de lemas, etc., así como aspectos relacionados con el tratamiento de acentos, siglas, nombres propios, etc.
- Parámetros de salida. Aquí quedan indicados el nombre de los ficheros de salida del proceso, el valor umbral para *idf* y el número de decimales de los valores numéricos de tipo real.
- Parámetros para ficheros inversos. Estos parámetros son los nombres de ficheros donde se almacenan las claves hexadecimales, el módulo, el fichero inverso para los pesos y el fichero índice asociado.
- Parámetros para el análisis de los resultados. Estos se utilizan para obtener los resultados de precisión−exhaustividad por consulta y medias. Sirven también para obtener la gráfica de resultados.

6.1. Documentos

El objetivo del programa que trata los documentos es obtener los siguientes ficheros:

- Fichero de frecuencias. Ese fichero almacena el identificador del documento, junto con los términos y la frecuencia de cada término. El formato es el indicado. Debemos resaltar que, para ahorrar espacio, solo se almacenan los términos que aparecen en el documento, pues el resto de términos tendrá frecuencia cero.

```
clave_doc1;térm1,frec11;term2,frec12;....
clave_doc2;térm1,frec21;term2,frec22;....
```

- Fichero para *df* (recuerde el modelo vectorial). El fichero recoge, para cada término, el número de documentos en que aparece (*df*). El formato es el siguiente:

```
term1;df1
term2;df2
```

- Fichero para *idf*. Aplicando la ecuación 4 se obtiene el valor de *idf* de cada término, [idf = log(N/df)]. El valor numérico se almacena con el número de decimales indicado en el fichero de configuración. El fichero para *idf* tiene el siguiente formato:

```
term1;idf1
term2;idf2
```

- Fichero de pesos. A partir del fichero de frecuencias y del fichero para *idf* se obtiene el fichero de pesos aplicando la ecuación 4. El formato es similar al indicado para el fichero de frecuencias, y el número de decimales para el peso es el dado en un parámetro del fichero de configuración.

```
clave_doc1;térm1,peso11;term2,peso12;....
clave_doc2;térm1,peso21;term2,peso22;....
```

- Fichero de claves hexadecimales. Para reducir el tamaño de los archivos en el proceso de similitud con ficheros inversos, se almacenarán las claves en formato hexadecimal. Para ello se obtendrá un archivo con el siguiente formato:

```
clave_hex1;clave_doc1
clave_hex2;clave_doc2
```

- Fichero inverso de términos para el fichero de pesos. A partir del fichero de pesos se obtiene el fichero inverso que reducirá el tiempo de procesamiento. El formato del fichero es:

```
term1;clave_hex,peso;clave_hex,peso;...
term2;clave_hex,peso;clave_hex,peso;...
...
```

- Fichero índice para el fichero inverso. Este fichero almacena la estructura del fichero inverso, para realizar búsquedas muy rápidas. El formato del fichero es:

```
term1;offset;long
term2;offset;long
...
```

- Fichero de módulos. También para agilizar el procesamiento de la similitud entre documentos y consultas, nos hemos decantado por tener un fichero aparte con el módulo de los vectores de pesos de los documentos. El formato es el siguiente:

```
clave_hex1;módulo1
clave_hex2;módulo2
...
```

6.2. Consultas

Para las consultas el proceso que se sigue es similar al de documentos, salvo que ahora no es necesario obtener ficheros inversos. Se han tomado las consultas en un proceso por lotes, con los mismos criterios léxicos que los documentos.

El objetivo del programa es obtener el fichero de frecuencias y el fichero de pesos para la colección de consultas. El primero de ellos almacena el identificador de cada consulta, junto con los términos y la frecuencia de cada término. El formato es similar al correspondiente para los documentos. El fichero de pesos es similar al de frecuencia, pero almacena los pesos.

El proceso que sigue el programa es idéntico al correspondiente para los documentos, salvo que no se obtienen ficheros inversos. Para obtener el fichero de pesos para las consultas se utiliza el fichero para *idf* obtenido en el procesamiento de los documentos.

6.3. Similitud

El programa calcula la similitud entre los vectores de la consultas y los documentos a partir de los ficheros indicados en el archivo de configuración. El objetivo es obtener un fichero con el resultado del experimento.

Los archivos implicados en el proceso son los siguientes:

- Fichero de pesos para las consultas
- Fichero inverso para los pesos de los documentos
- Fichero índice para el fichero inverso
- Ficheros de claves y de módulos

El objetivo del programa es obtener un fichero de salida, con el formato de la colección CLEF. El valor de la similitud se almacena con 6 decimales, tal como se indica a continuación.

```
xx Q0 xxxxxxxxxxx-xxxxx x x.xxxxxx xxxxxxx
1  2  3                 4 5        6

1: número de consulta (C067 => 67), en orden creciente
2: Q0 (invariable)
3: DOCNO: EFE1994xxxx-xxxxx
4: Ranking: desde 0 hasta NDPC-1, en orden creciente
5: Valor similitud, en orden decreciente
6: Identificación del experimento: p.ej. usal01
```

7. Resultados del experimento

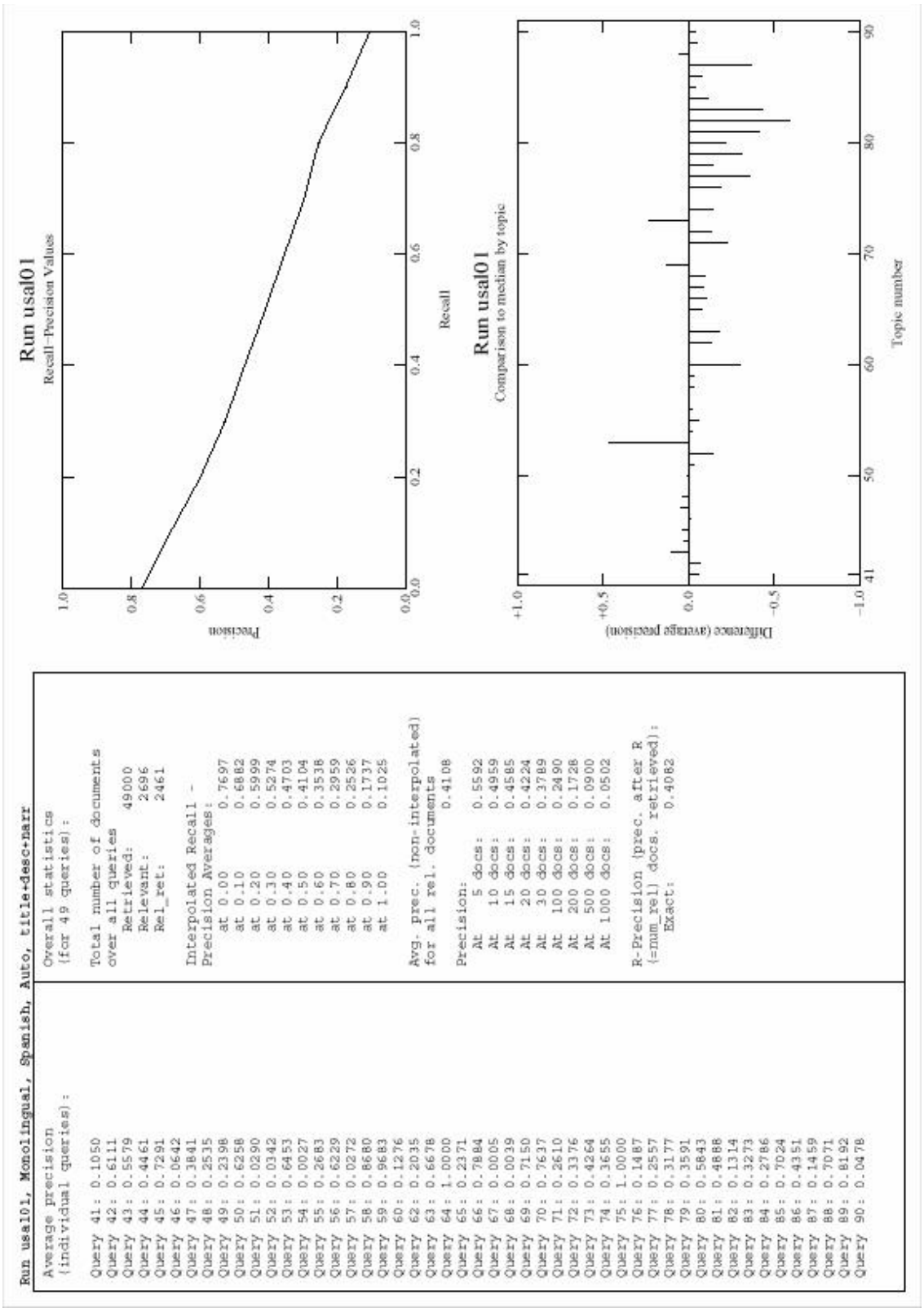
Hemos llevado a cabo tres experimentos, con el objetivo principal de obtener resultados sobre la calidad del algoritmo de lematización. Para los tres experimentos los criterios de procesado léxico han sido los siguientes:

- El contenido de los documentos se ha tomado de los campos **TITLE** y **TEXT**. El campo **TITLE** de los documentos se ha convertido a minúsculas, ya que originalmente está en mayúsculas, que se marcarían erróneamente como nombres propios.
- Se han tomado los tres campos de las consultas (**ES-title**, **ES-desc** y **ES-narr**).
- Al inicio del proceso se ha eliminado el carácter '_' del texto de los documentos y consultas, ya que ha sido el elegido como marcador de siglas y nombres propios.
- Consideración de números como términos índice, tal como se ha indicado en el apartado 5.1.4.
- Sustitución de guiones y comillas junto a números por un espacio.
- Conversión de acentos a formas no acentuadas.
- Marcado de siglas. El carácter de marcado ha sido '_' (_BBC_).
- Marcado de nombres propios. El carácter de marcado ha sido '_' (_Luis_, _Jose_Maria_, _Villaseco_de_los_Gamitos_). Se ha optado por almacenar solamente los nombres propios simples, y reducirlos también a minúsculas (luis, villaseco, gamitos). Lo mismo para las siglas (bbc). Los separadores de oraciones considerados han sido: .:;¡!¿?. No se ha considerado ponderación especial para nombres propios y siglas.
- Reducción a minúsculas de términos índice.
- Eliminación de palabras vacías (conjunto de 505 palabras).
- Diccionario de nombre propios.

En el primer experimento no se ha realizado lematización. El segundo se ha realizado lematización flexiva, y derivativa para el tercero.

Debemos mencionar aquí un aspecto importantísimo a la hora de obtener los resultados de precisión-exhaustividad. Los resultados están condicionados, en el sentido de que nuestros resultados han colaborado a determinar los documentos relevantes considerados para cada consulta. El proceso para determinar los documentos relevantes se ha llevado a cabo con los resultados de los experimentos de otros participantes en CLEF-2001 (para la colección en Español, han sido 12 grupos de investigación). Es decir, son resultados parciales, que pueden enmascarar errores.

7.1. Experimento 1. Sin lematizar



Run usa101, Monolingual, Spanish, Auto, title+desc+narr

Average precision
(individual queries):

Query 41 : 0.1050
Query 42 : 0.6111
Query 43 : 0.5579
Query 44 : 0.4461
Query 45 : 0.7291
Query 46 : 0.0642
Query 47 : 0.3841
Query 48 : 0.2535
Query 49 : 0.2398
Query 50 : 0.6258
Query 51 : 0.0290
Query 52 : 0.0342
Query 53 : 0.6453
Query 54 : 0.0027
Query 55 : 0.2683
Query 56 : 0.6229
Query 57 : 0.0272
Query 58 : 0.8680
Query 59 : 0.9683
Query 60 : 0.1276
Query 62 : 0.2035
Query 63 : 0.6678
Query 64 : 1.0000
Query 65 : 0.2371
Query 66 : 0.7884
Query 67 : 0.0005
Query 68 : 0.0039
Query 69 : 0.7150
Query 70 : 0.7637
Query 71 : 0.2610
Query 72 : 0.3376
Query 73 : 0.4264
Query 74 : 0.3655
Query 75 : 1.0000
Query 76 : 0.1487
Query 77 : 0.2557
Query 78 : 0.3177
Query 79 : 0.3591
Query 80 : 0.5843
Query 81 : 0.4888
Query 82 : 0.1314
Query 83 : 0.3273
Query 84 : 0.2786
Query 85 : 0.7024
Query 86 : 0.4351
Query 87 : 0.1459
Query 88 : 0.7071
Query 89 : 0.8192
Query 90 : 0.0478

Overall statistics
(for 49 queries):

Total number of documents
over all queries
Retrieved: 49000
Relevant: 2696
Rel_ret: 2461

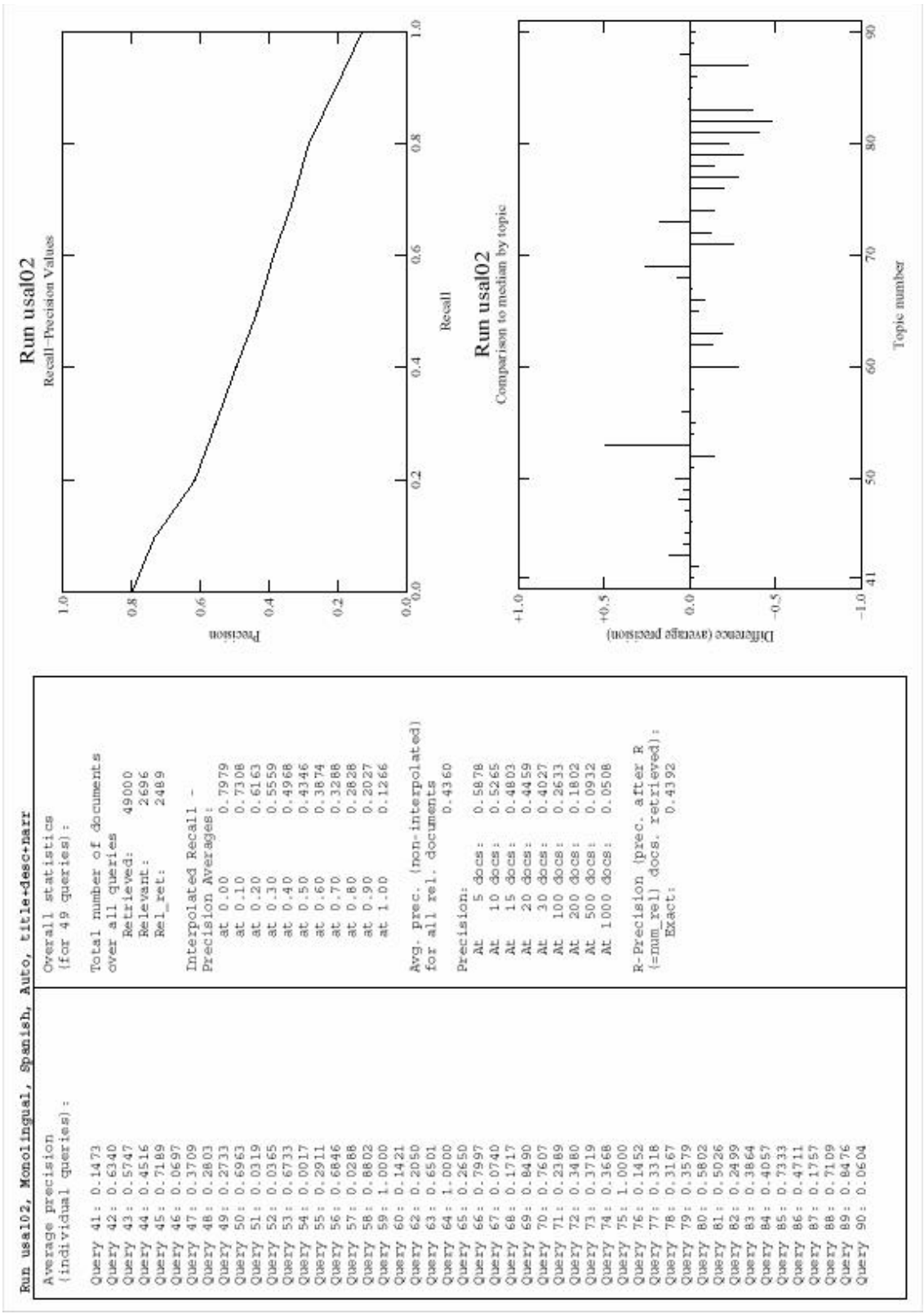
Interpolated Recall -
Precision Averages:
at 0.00 0.7697
at 0.10 0.6882
at 0.20 0.5999
at 0.30 0.5274
at 0.40 0.4703
at 0.50 0.4104
at 0.60 0.3538
at 0.70 0.2959
at 0.80 0.2526
at 0.90 0.1737
at 1.00 0.1025

Avg. prec. (non-interpolated)
for all rel. documents
0.4108

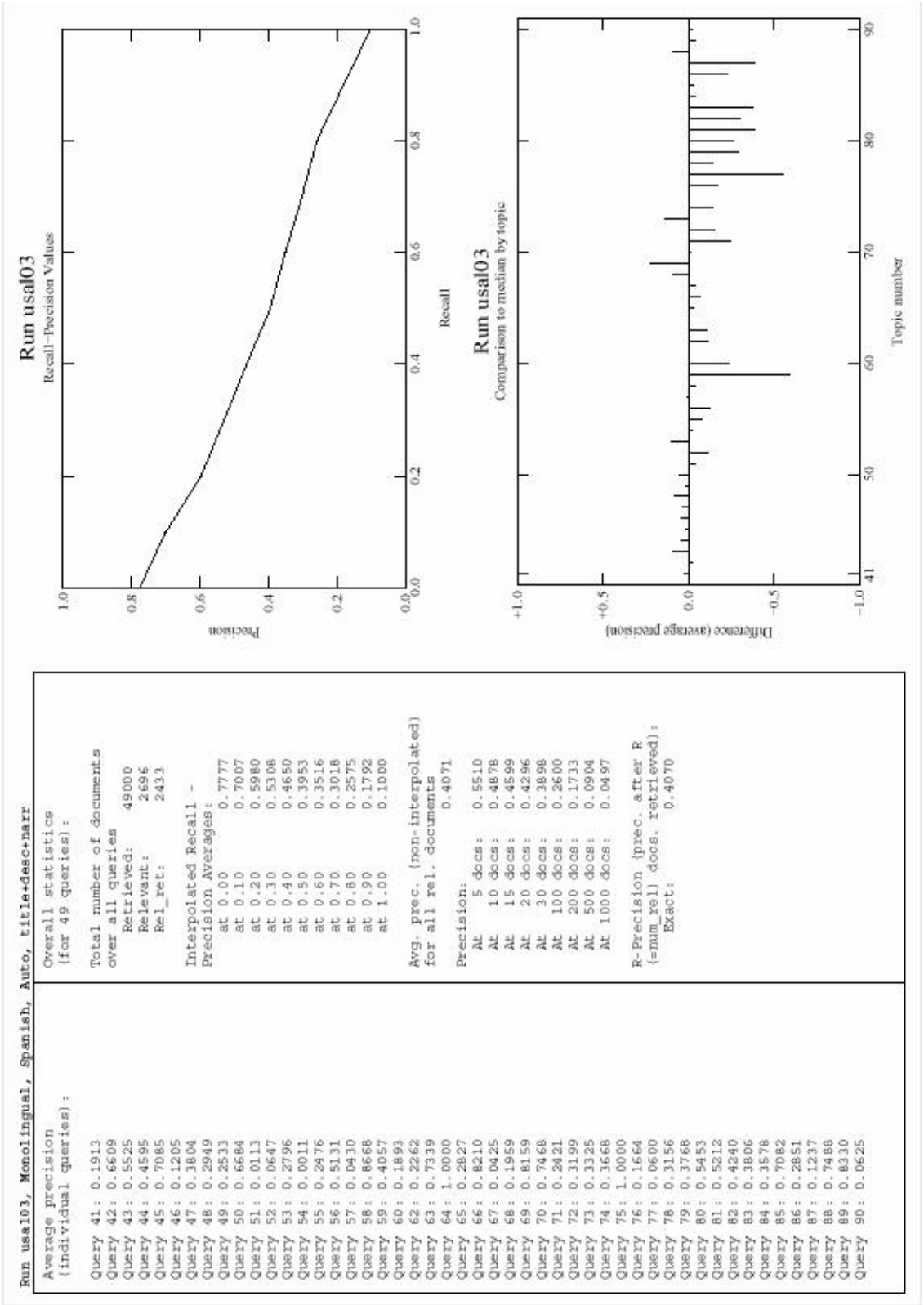
Precision:
At 5 docs: 0.5592
At 10 docs: 0.4959
At 15 docs: 0.4585
At 20 docs: 0.4224
At 30 docs: 0.3789
At 100 docs: 0.2490
At 200 docs: 0.1728
At 500 docs: 0.0900
At 1000 docs: 0.0502

R-Precision (prec. after R
(=num_rel) docs. retrieved):
Exact: 0.4082

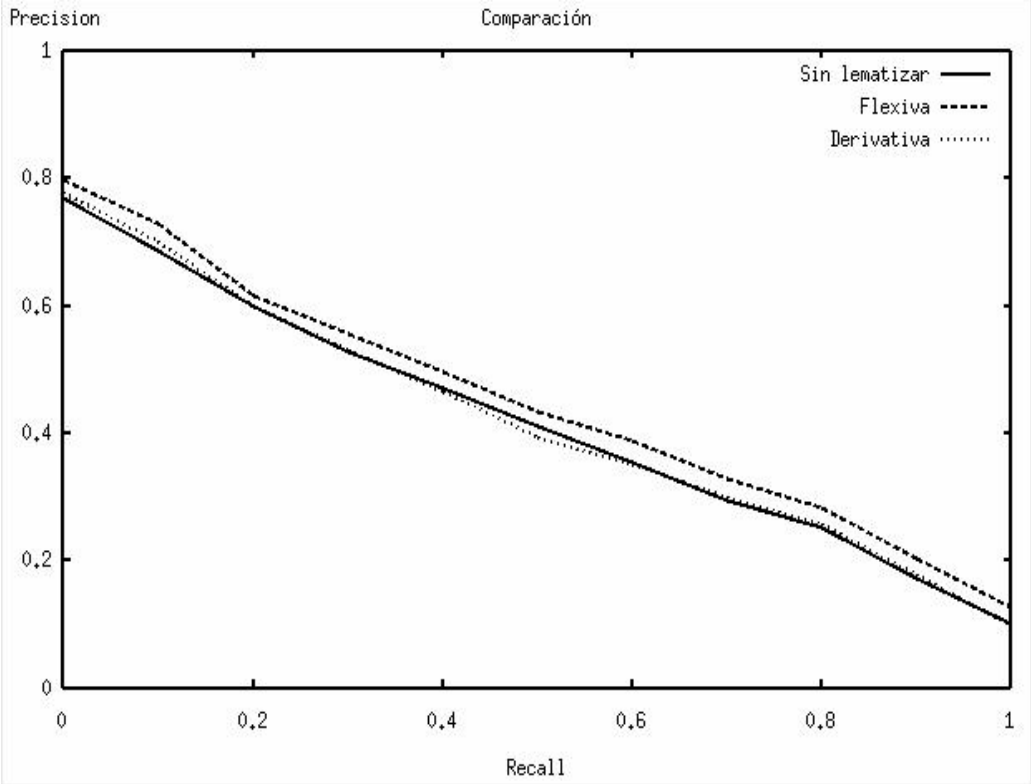
7.2. Experimento 2. Lematización flexiva



7.3. Experimento 3. Lematización derivativa



7.4. Comparación de resultados



Podemos observar que los mejores resultados se obtienen con la lematización flexiva. Con la derivativa los resultados son similares a cuando no se realiza lematización, sin embargo el coste computacional es mucho mayor que para la lematización flexiva. Podemos afirmar entonces que para la recuperación de información, y considerando todas las limitaciones expuestas en este documento, parece más idóneo realizar una lematización flexiva.

No olvidemos que los resultados han intervenido en la selección de documentos relevantes, y por tanto hay cierto grado de parcialidad.

8. Referencias

[Baeza−Yates y Ribeiro−Neto, 1999] Baeza−Yates, Ricardo, y Ribeiro−Neto, Berthier. *Modern information retrieval*. Harlow [England], etc: Addison−Wesley, 1999.

[Belkin y Croft, 1987] Belkin, N.J.; Croft, W.B. Retrieval techniques. *Annual Review of Information Science and Technology*, 22, p. 109−145. (1987)

[Buckley et al., 1994] Buckley, C.; Allan, J.; Salton, G. Automatic routing and ad−hoc retrieval using SMART: TREC 2. // Donna Harman, editor, *Proceedings of the Second Text Retrieval Conference TREC−2*. NIST Special Publication , 500−215. (1994).

[Figuerola et al., 2000] Figuerola, C.G.; Gómez Díaz, R.; López de San Román, E. Stemming and n−grams in Spanish: an evaluation of their impact on information retrieval. En: *Journal of Information Science*, 26 (6) 2000, pp. 461−467.

[Figuerola et al., 2001a] Figuerola, C.G.; Alonso Berrocal, J.L.; Zazo Rodríguez, A.F.; Gómez Díaz, R. A simple approach to the Spanish−English Bilingual retrieval task. En: Peters, C. (Ed.). *Cross Language Information Retrival and Evaluation*, pp. 224−229. Springer−Verlag: Berlin, N.Y., [etc.], 2001.

[Figuerola et al., 2001b] Figuerola, C.G.; Alonso Berrocal, J.L.; Zazo Rodríguez, A.F. Diseño de un motor de recuperación de información para uso experimental y educativo. En: *BID. Textos universitaris de bibliotecomia i documentació*, V. 11, pp. 201−209, 2001.

[Fox−1992] Christopher Fox. Lexical Analysis and Stoplists. En W.B. Frakes y R. Baeza−Yates. *Information retrieval, data structures and algorithms*. New Jersey, London, etc.: Prentice−Hall, 1992.

[Gómez, 2001] Gómez Díaz, Raquel. *Estudio de la incidencia del conocimiento lingüístico en los sistemas de recuperación de información para el español*. Tesis doctoral. Salamanca: Ediciones Universidad de Salamanca, 2001 (colección Vitor).

[Harman, 1992] Harman, D. Ranking Algorithms. //Frakes, W.B.; Baeza−Yates, R. *Information retrieval: Data Structures and Algorithms*. Prentice−Hall, Englewood Cliffs (NJ), pp. 363−392. (1992).

[Harter y Hert, 1977] Harter, S.P.; Hert, C.A. Evaluation of information retrieval systems. // *Annual Review of Information Science and Technology*, 32, p. 3−94. (1977).

[Hooper, 1965] Hooper, R. S. *Indexer consistency tests−origin, measurements, results and uti−lization*. Bethesda, MD. (1965).

[NIST, 2002] NIST (National Institute of Standards and Technology), Information Technology Laboratory's (ITL) Retrieval Group of the Information Access Division (IAD) and the Information Technology Office of the Defense Advanced Research Projects Agency (DARPA) and the Advanced Research and Development Agency (ARDA): Text REtrie−val Conference (TREC). [http://trec.nist.gov, consulta: abril 2002].

[Peters, 2001] Peters, Carol. CLEF−2001: Workshop of the Cross−Language Evaluation Forum. En: ERCIM News No.47, October 2001, http://www.ercim.org/publication/Ercim_News/enw47/clef2001.html

[Rijsbergen, 1979] Rijsbergen, Keith V. *Information Retrieval*. 2nd print. London: Butther−worths, 1979. [También en línea: http://www.dcs.gla.ac.uk/Keith/ (consultado en abril de 2002].

[Salton, 1968] Salton, G. *Automatic Information Organization and Retrieval*. McGraw−Hill, N.Y. (1968).

[Salton y Buckley, 1988] Salton, G.; Buckley, C. Term−Witghting Approaches in Automatic Text Retrieval. En: *Information Processing and Management*, 24(5), 513−523. (1988)

[Salton y Buckley, 1990] Salton, G.; Buckley, C. Improving retrieval performance by relevance feedback. *Journal of the American Society for Information Science*, 41 (4), 288−297. (1990).

[Salton y McGill, 1983] Salton, G.; McGill, M.J. *Introduction to Modern Information Retrie−val*. McGraw−Hill, New York. (1983)

[Stubbs et al., 2000] Stubbs, E. A.; Mangiaterra, N.E; Martinez, A. M. Internal quality audit of indexing: a new application of interindexer consytency. En: *Cataloguing & Classifica−tion Quaterly*, 28(4), 53−70. (2000).

[Voorhees y Harman, 1997] Ellen M. Voorhees, Donna Harman. Overview of the Six Text REtrieval Conference (TREC−6). En: E.M. Voorhees and D. K. Harman (editors), *Pro−ceedings of the Six Text REtrieval Conference (TREC−6)*, pages 1−24, November 1997. NIST Special Publication 500−240.

[Wall, 2002] Wall, Larry. *Larry Wall's Very Own Perl Page*.
[En línea: http://www.wall.org/~larry/perl.html (consultado: abril 2002)].