Inteligencia Artificial. Revista Iberoamericana de Inteligencia Artificial
ISSN: 1137-3601
revista@aepia.org
Asociación Española para la Inteligencia Artificial
España

Figuerola, Carlos G.; Zazo, Ángel F.; Rodriguez Vazquez de Aldana, Emilio; Alonso Berrocal, José Luis
La Recuperación de Información en español y la normalizacion de términos
Inteligencia Artificial. Revista Iberoamericana de Inteligencia Artificial, vol. 8, núm. 22, primavera, 2004, pp. 135-145
Asociación Española para la Inteligencia Artificial
Valencia, España

ARTÍCULO

La Recuperación de Información en español y la normalización de términos

Carlos G. Figuerola, Ángel F. Zazo,
Emilio Rodríguez Vázquez de Aldana, José Luis Alonso Berrocal
Grupo REINA
Universidad de Salamanca
c/ Fco. de Vitoria, 6-16, 37008 Salamanca-España
http://reina.usal.es

La mayor parte de los Sistemas de Recuperación de Información utilizan, de una forma u otra, recuentos de frecuencias de las palabras que aparecen en los documentos. Tales recuentos conllevan la necesidad de normalizar dichos términos. Una simple normalización de caracteres (mayúsculas/minúsculas, acentos y otros diacríticos) parece insuficiente, ya que muchas palabras, por flexión morfológica o derivación, podrán ser agrupadas bajo una única forma, al tener contenidos semánticos muy cercanos. Se analizan diversos algoritmos de normalización y se muestran los experimentos llevados a cabo para evaluar su eficacia.

La Recuperación de Información en español y la normalización de términos

Carlos G. Figuerola, Ángel F. Zazo,
Emilio Rodríguez Vázquez de Aldana, José Luis Alonso Berrocal

Grupo REINA
Universidad de Salamanca
c/ Fco. de Vitoria, 6-16, 37008 Salamanca-España
http://reina.usal.es

Resumen

La mayor parte de los Sistemas de Recuperación de Información utilizan, de una forma u otra, recuentos de frecuencias de las palabras que aparecen en los documentos.Tales recuentos conllevan la necesidad de normalizar dichos términos. Una simple normalización de caracteres (mayúsculas/minúsculas, acentos y otros diacríticos) parece insuficiente, ya que muchas palabras, por flexión morfológica o derivación, podrían ser agrupadas bajo una única forma, al tener contenidos semánticos muy cercanos. Se analizan diversos algoritmos de normalización y se muestran los experimentos llevados a cabo para evaluar su eficacia.

Palabras clave: Recuperación de la Información, normalización, términos, n-gramas, s-stemmer, lematización flexiva, lematización derivativa

Abstract

Most of the Information Retrieval Systems uses counts of frequencies of the words that occur in documents. Such counts entail the need of normalizing these terms. A simple normalization of characters (upper/lowercase, accents and other diacritical ones) seems insufficient, since many words, by morphologic inflection or derivation, could be grouped under an only form, when having very near semantic mean. Several algorithms of normalization are analyzed and tested experimentally to evaluate their effectiveness.

Keywords: Information Retrieval, stemming, n-grams, inflectional stemming, derivational stemming.

1. Introducción

La mayor parte de los modelos y técnicas empleados en Recuperación de la Información utilizan en algún momento recuentos de frecuencias de los términos que aparecen en los documentos y en las consultas. Esto implica la necesidad de normalizar dichos términos, de manera que los recuentos puedan efectuarse de manera adecuada.

Dejando de lado la cuestión de las llamadas palabras vacías, a las que no cabe considerar como tales términos, tenemos el caso de las palabras derivadas del mismo lema, a las que cabe atribuir un contenido semántico muy próximo [17].

Las posibles variaciones de los derivados, junto con formas flexionadas, alteraciones en género y número, etc. hacen aconsejable un agrupamiento de tales variantes bajo un único término. Lo contrario produce una dispersión en el cálculo de frecuencias de tales términos, así como la dificultad de comparar consultas y documentos [30].

Esta operación es lo que en inglés se conoce como *stemming*. La eficacia del *stemming* ha sido objeto de discusión, entre otros por Harman [14], quien después de probar varios algoritmos (para el inglés) concluyó que ninguno de ellos aumentaba la efectividad en la recuperación.Trabajos posteriores [29] apuntaron en el sentido de que el *stemming* es eficaz en función de la complejidad morfológica de la lengua en que se opere, mientras que Krovetz [22] encuentra que el *stemming* mejora la exhaustividad, e incluso también la precisión cuando documentos y consultas son de corta longitud.

El *stemming* parece, en consecuencia, fuertemente dependiente del idioma en que se encuentran documentos y consultas, de manera que resulta difícil aplicar algoritmos diseñados para un idioma a información en otra lengua diferente. No solamente los sufijos y raices son diferentes, sino que la forma en que aquéllos se adhieren a éstas es distinta. No obstante, se han propuesto sistemas elementales de *stemming* que son básicamente independientes del idioma. Éste es el caso de los n-gramas y, en buena medida, de los *s-stemmers*, aunque éstos requieren alguna pequeña adaptación.

Este trabajo explora las posibilidades y efecto de el *stemming* en la Recuperación de Información en español. Se han llevado a cabo diversos experimentos aplicando diferentes sistemas de *stemming* en español y evaluando sus resultados. Este trabajo está organizado de la siguiente manera: en la siguiente sección se comentan trabajos previos sobre *stemming*; en la sección 3 se describen los sistemas de *stemming* considerados, y en la sección 4 se describen los experimentos llevados a cabo para evaluar el alcance de cada uno de estos sistemas. Finalmente, en la sección 5 se ofrecen las conclusiones.

2. Trabajos previos

El *stemming* aplicada a la Recuperación de la Información se ha planteado de diversas maneras, desde un escueto stripping, hasta la aplicación de algoritmos bastante más sofisticados. Comienza a estudiarse en los años 60, con el fin de reducir los tamaños de los índices [4], y, además de una forma de normalizar los términos, puede verse también como una forma de expandir las consultas, añadiendo formas flexionadas o derivadas de las palabras a documentos y consultas.

Entre las aportaciones más conocidas encontramos el algoritmo propuesto por Lovin en 1968 [23], el cual, de alguna manera, está en la base de algoritmos y propuestas posteriores, como los de Dawson [11], Porter [30] y Paice [27]. Aunque buena parte de los trabajos están orientados a su uso con documentos en inglés, es posible encontrar propuestas y algoritmos para lenguas específicas; entre ellas el propio latín, a pesar de ser una lengua muerta [40], el malayo [2], el francés [38],[39], el árabe [1], neerlandés [21], [20], esloveno [29] o griego [19].

Por lo que se refiere al español, se aplicaron diversos mecanismos de *stemming* en operaciones de Recuperación de la Información en algunas de las conferencias TREC (Text REtrieval Conference) [15]. En general, estas aplicaciones consistieron en la utilización de los mismos algoritmos que para el inglés, aunque con sufijos y reglas para el español. Independientemente de los algoritmos aplicados, y de su adecuación al idioma español, el conocimiento lingüístico implementado (listas de sufijos, reglas de aplicación, etc.) era bastante pobre [12].

Desde el punto de vista del procesamiento de lenguaje, se han desarrollado en los últimos años varios lematizadores y analizadores morfológicos para el español; entre ellos, las herramientas COES [32], puestas a disposición del público por sus autores bajo licencia GNU [1]; el analizador morfosintáctico MACO+ [5] [2] o los lematizadores FLANOM [37] / FLAVER [36] [3]. Desconocemos, sin embargo, resultados experimentales de su aplicación a la Recuperación de Información.

De otro lado, se ha propuesto en diversas ocasiones el uso de n-gramas para obviar el problema planteado por formas flexionadas y derivadas de las palabras [31]. En trabajos previos, sin embar-

[1] http://www.datsi.fi.upm.es/~coes/
[2] http://nipadio.lsi.upc.es/cgi-bin/demo/demo.pl
[3] http://protos.dis.ulpgc.es/

go, pudimos comprobar la escasa efectividad de este mecanismo, siempre desde el punto de vista de la Recuperación de Información [13], así como la inadecuación del conocido algoritmo de Porter para idiomas como el español.

3. Algoritmos de *stemming*

3.1. s-stemmer

La idea básica es conseguir una reducción de plurales a singular, como forma de normalizar términos. En su forma original (para el inglés), el *s-stemmer* simplemente elimina las *s* finales de cada palabra. Para el español, puede enriquecerse teniendo en cuenta que los plurales de sustantivos y adjetivos terminados en consonante se consiguen con el sufijo *-es*. Remover directamente las terminaciones en *-es* puede producir inconsistencia con el caso de los plurales de palabras que directamente terminan en *-e* en su forma singular, por lo que, en nuestro caso, removemos también las *-e* finales.

La implementación en crudo de un *s-stemmer* es trivial, y ésta es una de sus ventajas. Sin embargo, el *s-stemmer* es incapaz de distinguir sustantivos y adjetivos de otras categorías gramaticales y se aplica indiscriminadamente a todas las palabras; tampoco contempla plurales irregulares. En compensación, al tratar todas las palabras de la misma forma, no introduce ruido adicional.

3.2. n-gramas

La descomposición de texto en n-gramas tiene múltiples aplicaciones; uno de los usos propuestos es atemperar la dispersión producida por flexiones y derivaciones de un mismo lema [31]. La idea básica es que dos palabras con la misma raíz, pero distinto sufijo (cuya aposición puede, además, modificar parte de la raíz), al ser descompuestas en n-gramas, producirán una serie de ellos iguales (los correspondientes a la parte de la raíz), más otros diferentes (los correspondientes a la parte del sufijo). En Sistemas de Recuperación basados en n-gramas, palabras pertenecientes a la misma familia deberían producir una cierta similitud.

Los n-gramas han sido utilizados con cierta frecuencia en Recuperación de Información, como muestran los trabajos de [16], [10], [8], [7] o [6].

3.3. Lematización flexiva

A diferencia de las técnicas anteriores, es posible aplicar conocimiento lingüístico más complejo para tratar de reducir todas las palabras de una misma familia a un mismo lema. En este sentido, parece que la distancia semántica entre un lema y sus posibles variantes morfológicas o flexiones es escasa, por lo que fundir tales términos bajo una única forma podría aumentar la eficiencia en la recuperación.

Determinar el lema original que, flexionado, ha dado lugar a un término que podamos hallar en un documento o consulta implica efectuar un análisis morfológico de dicho término. La base de nuestro analizador morfológico consiste en un traductor de estados finitos (FST, *Finite-State Transducer*) que intenta representar las modificaciones experimentadas por un lema cuando se flexiona, añadiéndole un determinado sufijo. Así, hay una instancia de ese FST para cada sufijo contemplado; cada uno de éstos conlleva una serie de reglas que expresan cómo se incorpora ese sufijo a un lema. Dado que, para un mismo sufijo, puede existir multitud de variantes y de excepciones, en ocasiones el FST resultante puede ser bastante complejo.

Así, para lematizar una palabra, se busca el sufijo más largo que coincida con el fin de esa palabra y se forma el FST correspondiente con las reglas de ese sufijo. La red de ese FST se recorre con la palabra a lematizar, y las cadenas obtenidas en el nodo terminal del FST se contrastan con un diccionario de lemas. Si se encuentra en ese diccionario, la cadena obtenida se da como lema correcto. Sin embargo, a un mismo término pueden aplicársele en ocasiones varios sufijos, lo que obliga a probar todos ellos.

Si, agotadas las posibilidades, ninguna de las cadenas terminales obtenidas por el FST se encuentra en el diccionario de lemas, se deduce que, o bien la palabra puede considerarse normalizada en sí misma, o bien se trata de un caso no previsto por el lematizador.

Esto último puede significar lo siguiente:

- a) la palabra tiene un sufijo no recogido en la lista de sufijos del lematizador
- b) el sufijo se añade de manera no prevista por las reglas incorporadas en la base de

conocimiento

- c) el lema no está recogido en el diccionario de lemas.

Esto permite someter al lematizador a un proceso de entrenamiento de manera que, manualmente, se examinan los resultados de lematizar palabras de un corpus y se corrige la base de conocimiento del lematizador cuando ello es necesario.

Cada sufijo contemplado lleva información acerca del tipo de flexión que produce. Sin embargo, un mismo sufijo puede corresponder a diversas flexiones, por lo que, en muchos casos, obtendremos varios posibles lemas para un mismo término. De otro lado, hay sufijos que pueden solaparse entre sí; esto produce también varios lemas para un mismo término. Así pues, es preciso efectuar una desambiguación morfosintáctica para elegir el lema correcto.

La función de un desambiguador morfosintáctico o *Part Of Speech Tagger* radica en asignar, a cada palabra de una oración, una única etiqueta morfosintáctica entre un conjunto previamente definido, teniendo en cuenta el contexto de cada una de aquéllas. Para la resolución de este problema, se han propuesto tres tipos de aproximaciones: la linguística o basada en un conjunto de reglas explícitas, la estadística, basada en los Modelos de Markov, y una solución híbrida, que combina ciertas características de ambas [26].

Aunque la precisión conseguida en el etiquetado se sitúa entre el 95 % y el 97 % para los modelos estadísticos y algo superior para los modelos lingüísticos [25] -p.e. para el Tagger ENGCG se ha estimado una corrección en el etiquetado del 99.5 %, aunque dejando un 3.2 % de palabras sin desambiguar [41]-, hemos optado por construir un desambiguador basado en los modelos estadísticos, pues el coste de desarrollo es claramente inferior al necesario para la implementación de uno basado en reglas [34].

El planteamiento del problema de la desambiguación morfosintáctica, desde un punto estadístico, radica en buscar qué secuencia única de etiquetas de salida T es la de máxima probabilidad dada la secuencia de palabras de entrada W. Formalmente, de lo que se trata es de resolver la siguiente expresión de probabilidad condicionada:

$$T = MaxP(t_1, t_2, ..., t_n | w_1, w_2, ...w_n) \quad (1)$$

Para encontrar una solución a este planteamiento inicial, se busca una aproximación -asunciones de Markov- que, aunque no siempre garantizará la solución correcta, se ha demostrado que produce resultados satisfactorios con costes computacionales razonables. Entre otra bibliografía, puede encontrarse en [18] y [3], el razonamiento detallado que reduce el problema, en los *modelos supervisados* -esto es, que han utilizado uno o varios corpora previamente etiquetados para la adquisición del conocimiento lingüístico- al cálculo de la *probabilidad léxica* y de la *probabilidad contextual* para cada etiqueta posible de cada palabra.

La *probabilidad léxica*, independiente para cada palabra, se aproxima de acuerdo a la expresión:

$$N(w, t)/N(t) \quad (2)$$

siendo $N(w, t)$ la frecuencia de aparición de la palabra w con la etiqueta t y $N(t)$ la frecuencia de aparición de la etiqueta t en el Corpus.

La *probabilidad contextual*, se aproxima, para un modelo basado en *bigramas* de acuerdo a la expresión:

$$N(t_i - 1, t_i)/N(t_i - 1) \quad (3)$$

donde $N(t_i - 1, t_i)$ es la frecuencia de aparición de la secuencia de etiquetas $t_i - 1$, t_i y $N(t_i - 1)$ la frecuencia de aparición de la etiqueta $t_i - 1$. Este modelo puede extenderse fácilmente al modelo trigramas, para considerar un contexto mayor.

Para la reconstrucción de la secuencia más probable de etiquetas de cada frase se suele utilizar el algoritmo de Viterbi, fundamentalmente porque su implementación es sencilla [24].

El problema que plantea este modelo, tan simple y a la vez eficiente, estriba en los tamaños de los Corpora etiquetados necesarios para obtener unas estimaciones aceptables [42]. No obstante, independientemente de las magnitudes, hay que adoptar alguna técnica de suavizado (*smoothing*) del modelo con el fin de evitar la asignación de probabilidades nulas, tanto para el cálculo de la probabilidad contextual como de la probabilidad léxica. Es decir, siempre será posible encontrarnos al analizar textos reales, por un lado, con secuencias de etiquetas correctas de las que no se ha obtenido información de los corpora utilizados para aprender y, por otro, con palabras desconocidas, es decir, no aparecidas en dichos corpora [33]. Esto último será más frecuente en lenguas con morfología rica, como es el caso del español.

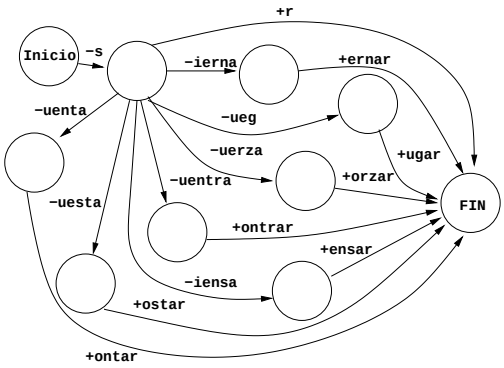

Figura 1: FST para el sufijo *-as* (2 persona presente indicativo)

El corpus que hemos utilizado ha sido el `LexEsp`[4] , que cuenta con aproximadamente 100.000 palabras etiquetadas de acuerdo a la propuesta PAROLE del grupo EAGLES sobre el etiquetado morfosintáctico, validadas manualmente. Las características del desambiguador construido, en parte condicionadas por el tamaño del corpus utilizado, son las siguientes:

- El juego de etiquetas lo hemos reducido a los dígitos que recogen información de categoría y subcategoría gramatical. De acuerdo a la codificación PAROLE, a los 3 primeros caracteres para los verbos y, a los dos primeros para las restantes categorías. La elección de este juego está motivada, también, por los objetivos perseguidos en nuestro trabajo tras el proceso de desambiguación.

- El modelo contextual utilizado es el modelo de bigramas. En [28], sobre el mismo corpus para este mismo modelo, se obtienen resultados de precisión en el etiquetado del 96.97 %. El método de suavizado elegido para la aproximación del cálculo de la probabilidad contextual utilizado es el conocido como interpolación lineal [9].

- Para el cálculo de la probabilidad léxica, dado que efectuamos un procesamiento morfológico previo, empleamos la siguiente estrategia. Para aquellas palabras reconocidas por el procesador morfológico, si aparece en el corpus información de frecuencias para todas sus posibles categorías, se calculará de acuerdo a la expresión (3) y, en caso contrario, se utilizará el método de suavizado conocido como *Añadir 1*, que en los experimentos de [28] proporciona resultados similares a otros métodos de suavizado. Para aquellas palabras no reconocidas por el procesador morfológico, estimaremos la probabilidad léxica, aceptando que sólo pertenecerán a las categorías abiertas -esto es nombres, adjetivos, verbos y adverbios-, adaptando el método de suavizado propuesto en [42], que aproxima la probabilidad de pertenencia de una palabra desconocida dada una categoría, a partir de un corpus de entrenamiento, combinando ciertas propiedades morfológicas como fines de palabras, si la palabra contiene la letra inicial en mayúsculas o no tanto en principio de frase como en el resto de posiciones, así como recabando información de frecuencias de aparición de las palabras desconocidas en el proceso de entrenamiento en las diferentes categorías.

3.4. Lematización derivativa

El objetivo de la lematización derivativa es agrupar bajo un mismo lema todas las palabras derivadas de éste. Independientemente del proceso técnico que haya de seguirse para ello, debe tenerse en cuenta que la derivación puede acabar produciendo palabras muy alejadas semánticamente del lema original. Así, por ejemplo, de `chica` tenemos el derivado `chiquilla`, que podríamos aceptar como términos muy cercanos, pero de `sombra` tenemos `sombrilla`, e incluso `sombrero`, que parecen mucho más separadas en cuanto a significado.

Asumiendo estos riesgos, debe considerarse que las palabras derivadas también pueden ser flexio-

[4] `LexEsp` es propiedad de ELDA

nadas, y que, incluso, varios sufijos derivativos pueden encadenarse. Por ello, el primer paso para efectuar la lematización derivativa es llevar a cabo la flexiva, tal como se ha descrito antes. Posteriormente actuaremos sobre los lemas flexivos para conseguir la lematización derivativa.

Para ello, hemos dispuesto una lista de sufijos derivativos de 230 elementos, con 3692 reglas de aplicación. Un traductor de estados finitos, previa localización de los sufijos que pueden emparejar con la palabra a lematizar, intenta obtener el lema correspondiente, aplicando las reglas correspondientes a los sufijos localizados. Cada lema obtenido por el traductor es contrastado con una lista de lemas derivativos (un diccionario de algo más de 15.000 entradas). Si se encuentra en el diccionario, ya se ha conseguido el lema; en caso contrario, y teniendo en cuenta que los sufijos pueden encadenarse, se vuelve a buscar el sufijo o sufijos que emparejen con el resultado producido por el traductor. El proceso, pues, actúa de forma recursiva hasta que se consigue un lema válido.

4. Experimentos

Hemos llevado a cabo algunos experimentos tendentes a comprobar la efectividad de estos diversos sistemas de normalización con documentos y consultas en español. La colección de documentos empleada es la proporcionada por CLEF (*Cross Language Evaluation Forum*: `http://clef.iei.pi.cnr.it`)[5] , elaborada con todas las noticias de la Agencia EFE del año 1994. Esta colección contiene también un juego de consultas con sus correspondientes estimaciones de relevancia. El número de documentos es de 215.738, con una extensión media de 126.6 términos no vacíos por documento. Los documentos contienen varios campos, de los cuales, a efectos de contenido textual para la recuperación son útiles el que contiene el título de la noticia y el que contiene el texto de la noticia propiamente dicho; el resto de los campos indican cosas como la fecha, la sección, diversas claves y números de referencia, etc.

Para todos los experimentos, se han eliminado de documentos y consultas las palabras vacías (usando una lista estándar de unas 300 palabras), y se ha efectuado una normalización elemental a nivel de carácter, convirtiendo todas las letras a minúscula y eliminando acentos; esta última decisión incrementa las dificultades de la lematización flexiva, obviamente, pero la hemos estimado necesaria debido a la inconsistencia en la aplicación real de acentos en español (a pesar de que, en el caso de los acentos, las normas ortográficas son especialmente unívocas y carecen prácticamente de excepciones). En parecida línea, se ha considerado separador de términos cualquier carácter no alfabético; los números también se consideran caracteres no alfabéticos.

El modelo de recuperación utilizado es el del bien conocido modelo vectorial [35], con un esquema de pesos `ntc` que, como es sabido, es el más frecuente. El *software* utilizado es de realización propia: un programa llamado `URRAKA` (`U`na `R`eforma `R`evisada y `A`mpliada de `K`arpanta).*Karpanta* (`http://reina.usal.es`) es nuestro motor experimental de recuperación.

Para estimar el efecto de los ditintos sistemas de normalización, se ha efectuado una ejecución previa sin ningún tipo de *stemming*, como línea base de comparación.

Para *el s-stemming* se ha aplicado un algoritmo ligeramente mejorado, en el sentido de que elimina también las vocales finales `a`, `e` y `o`. Con ello se pretende sortear los problemas de género; como se ha indicado antes para los plurales, el algoritmo es ciego, en el sentido de que trata por igual todas las palabras, sin discriminar adjetivos o sustantivos de otras formas gramaticales.

Para el caso de los n-gramas, se ha aplicado la práctica habitual de considerar que los términos a descomponer van precedidos y seguidos de un espacio en blanco. Se han considerado diversos tamaños (n=3, n=4, n=5, n=6). El experimento con n-gramas presenta un interés adicional: la colección CLEF está plagada de errores tipográficos, y una de las ventajas potenciales de los n-gramas es atenuar esta clase de errores [43].

Por lo que se refiere a la lematización flexiva, nuestro lematizador utiliza una lista de 88 sufijos, con unas 2.700 reglas de aplicación. El diccionario o lexicon consta de unas 70.000 entradas.

Dado que se considera, generalmente, que el *stemming* produce mejores resultados con consultas cortas, se han efectuado pruebas con las consultas completas (los tres campos) y sólo con el campo `<TITLE>` de dichas consultas. Debe considerarse, además, que, al menos en sistemas de recupera-

[5]Las colecciones CLEF son propiedad de CLEF Consortium y ELDA

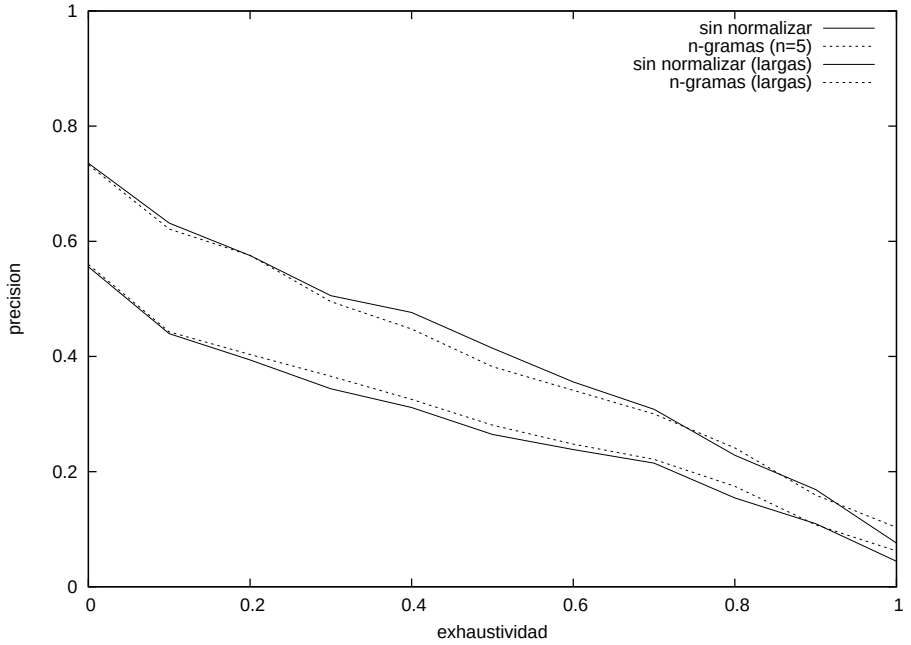

Figura 2: n-gramas frente a no normalización

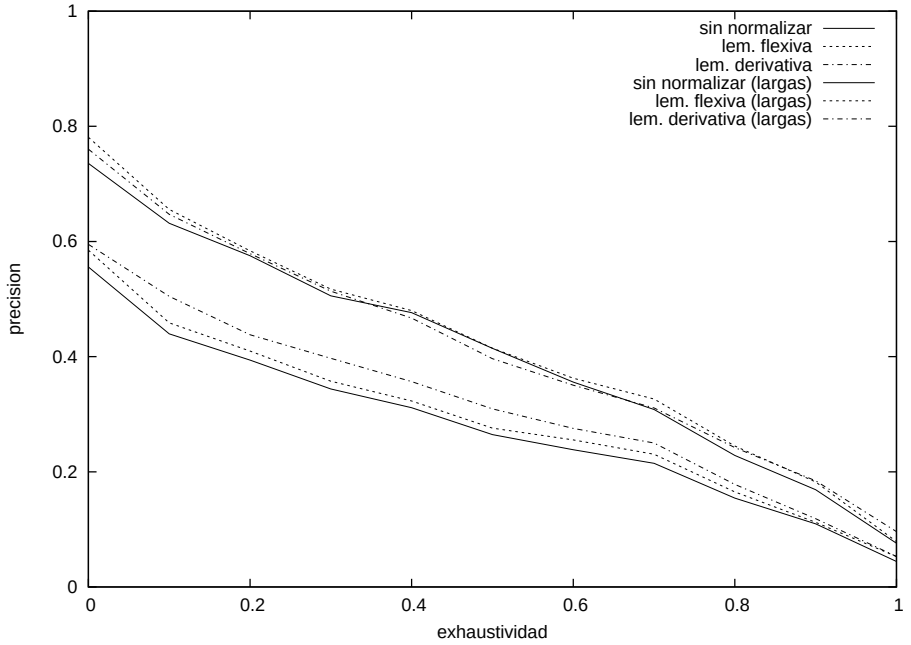

Figura 3: lematización flexiva y derivativa

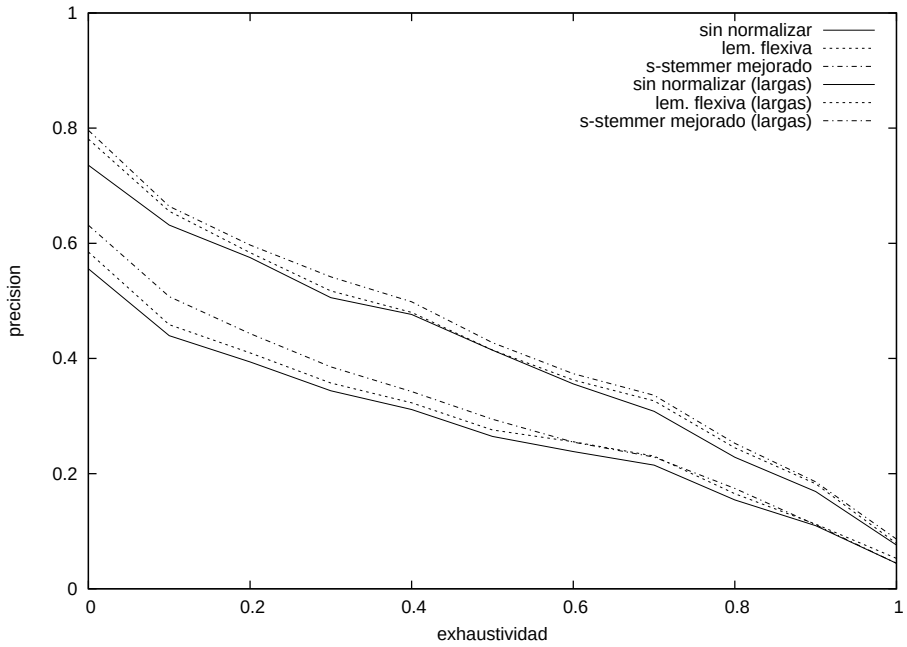

Figura 4: lematización flexiva y s-stemmer mejorado

ción interactivos, las consultas reales de los usuarios tienden a ser muy breves.

4.1. Resultados

Los resultados obtenidos se muestran en los gráficos adjuntos. La evaluación se ha efectuado tomando en cuenta los 1.000 primeros documentos devueltos para cada consulta en cada modalidad. En ellos se muestran los resultados obtenidos para cada método de normalización empleado, tanto con consultas largas (completas) como cortas (sólo el campo `<TITLE>`). Para el caso del *s-stemmer* se presentan los resultados obtenidos por el *s-stemmer* mejorado, ligeramente superiores a los obtenidos por el *s-stemmer* normal. En lo que se refiere a n-gramas, se muestran los resultados obtenidos con el mejor tamaño de n (n=5).

Lo primero que se aprecia es que la normalización produce mejoras claramente en todas las modalidades, salvo en la de n-gramas; en este último caso,la mejora es muy escasa para las consultas cortas, mientras que para las consultas largas los resultados empeoran frente a la no normalización. Para las demás modalidades de normalización, cuando se trata de consultas cortas, los resultados son peores que con las consultas largas, lo cual entra dentro de lo esperable. Sin embargo, en consultas cortas las diferencias sobre la no normalización se acentúan.

La lematización flexiva mejora resultados en todos los casos, y la derivativa supera a ésta cuando las consultas son cortas. Con consultas largas, sin embargo, la derivativa produce resultados inferiores a la flexiva, e incluso, para valores medios de exhaustividad, ligeramente inferiores a la no normalización.

Cabe destacar el buen comportamiento del *s-stemmer* mejorado, superior a todos los demás sistemas con consultas largas, y también para prácticamente todos los tramos con consultas cortas, salvo en lo que se refiere a lematización derivativa; en éste último caso, sin embargo, también es superior en los primeros tramos, con exhaustividad baja (normalmente, para los primeros documentos recuperados); el resto de la curva muestra resultados peores que los de la lematización derivativa, aunque las diferencias no son especialmente acusadas. Esto es especialmente importante si tenemos en cuenta la simpleza (incluso

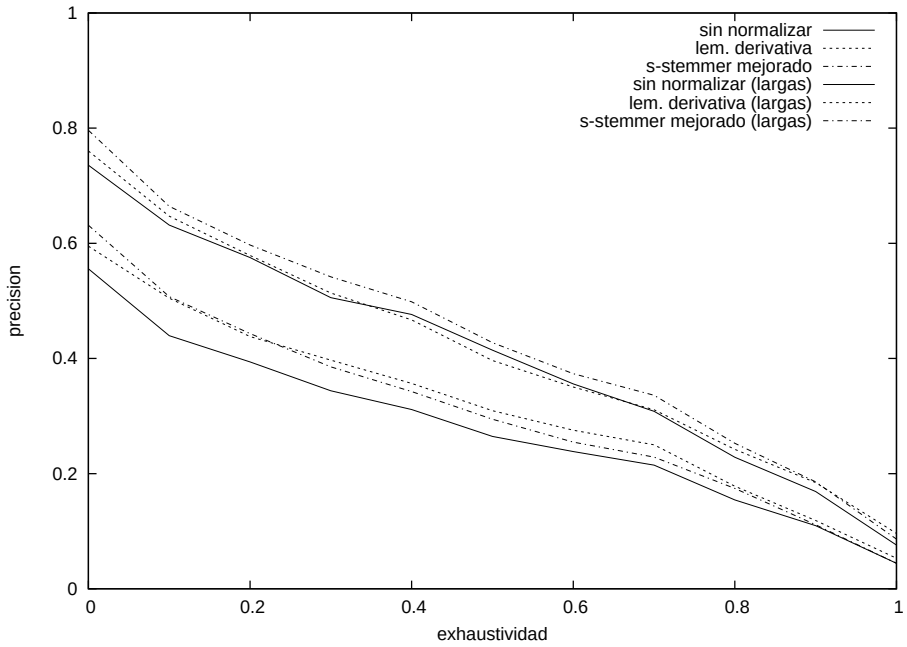

Figura 5: lematización derivativa y s-stemmer mejorado

la crudeza) de este algoritmo de normalización. Esta simpleza contrasta con los sistemas de lematización (tanto flexiva como derivativa), que requieren algoritmos sofisticados y conocimiento lingüístico complejo.

Sobre este punto, debe notarse que las técnicas lingüísticas tienen un margen de error; aunque este margen es reducido (y puede reducirse más, con la mejora del conocimiento lingüístico aplicado), penaliza, pues introduce ruido con cada error. La lematización derivativa, al operar sobre los resultados del lematizador flexivo, añade a los errores de éste sus propios problemas, fundamentalmente los derivados de la distancia semántica entre palabras derivadas y sus respectivos lemas. Su mayor poder de fusión de términos explica sus buenos resultados cuando las consultas son cortas; pero cuando éstas contienen un número mayor de términos, tales problemas afloran, empobreciendo comparativamente sus resultados.

5. Conclusiones

Se han descrito diversos algoritmos de normalización de términos y se ha probado experimentalmente con una amplia colección de documentos y consultas en español. Parece claro que esta normalización mejora los resultados en la recuperación, especialmente cuando las consultas son cortas, lo que suele ser el caso más frecuente. El uso de los n-gramas, por otra parte, parece desaconsejable, pues los resultados obtenidos no alcanzan los que se obtienen sin aplicar ningún tipo de normalización.

Los algoritmos más complejos, que incluyen conocimiento lingüístico, no obstante, no consiguen superar los resultados conseguidos con un simple *s-stemmer*, mucho más fácil de implementar.

Referencias

[1] H. Abu-Salem, M. Al-Omari, and M. W. Evens. Stemming methodologies over individual queries words for an arabian informa-

tion retrieval system. *JASIS*, 50(6):524–529, 1999.

[2] F. Ahmad, M. Yussof, and M. T. Sembok. Experiments with a stemming algorithm for malay words. *JASIS*, 47(12):909–918, 1996.

[3] J. Allen. *Natural Language Understanding*. Benjamin/Cummings, 1995.

[4] C. Bell and K. P. Jones. Toward everyday languaje information retrieval system via minicomputer. *JASIS*, 30:334–338, 1979.

[5] J. Carmona, S. Cervell, L. Márquez, M. Martí, L. Padró, R. Placer, H. Rodríguez, M. Taulé, and J. Turmo. An environment for morphosyntactic processing of unrestricted spanish text. In *LREC 98: Proceedings of the First International Conference on Language Resources and Evaluation*, number 1, Granada, España, 1998.

[6] W.B. Cavnar. N-gram based text filtering for trec-2. In D.K. Harman, editor, *The Second Text REtrieval Conference (TREC-2)*, number 2, pages 171–180, Gaithersburg, Maryland, noviembre 1993. National Institute of Standards and Technology (NIST), Advanced Research Projects Agency (ARPA).

[7] W.B. Cavnar. Using an n-gram based document representation with a vector processing retrieval model. In D.K. Harman, editor, *Overview of the Thrid Text REtrieval Conference (TREC-3)*, number 3, pages 269–278, Gaithersburg, Maryland, noviembre 1994. National Institute of Standards and Technology (NIST), Advanced Research Projects Agency (ARPA).

[8] William B. Cavnar and John M. Trenkle. N-gram based text categorization. In D.K. Harman, editor, *Proceedings of the Third Annual Symposium on Document Analysis and Information Retrieval*, number 3, pages 191–176, University of Nevada, Las Vegas, 1994. NIST.

[9] E. Charniak. *Statistical Language Learning*. The MIT Press, Cambridge (Massachussetts), 1993.

[10] M. Damashek. Gauging similarity with n-grams: Language-independent categorization of text. *Science*, (267):843–848, 1995.

[11] J. Dawson. Suffix removal and word conflation. *ALLC bulletin*, 2(3):33–46, 1974.

[12] Carlos G. Figuerola. La investigación sobre recuperación de información en español. In C. Gonzalo García and V. García Yedra, editors, *Documentación, Terminología y Traducción*, pages 73–82. Síntesis, Madrid, 2000.

[13] C.G. Figuerola, R. Gómez, and E. López de San Román. Stemming and n-grams in spanish: an evaluation of their impact on information retrieval. *Journal of Information Science*, 26(6):461–467, 2000.

[14] D. Harman. How effective is suffixing? *JASIS*, 42(1):7–15, 1991.

[15] D. Harman. Overview of the fourth text retrieval conference (trec-4). In D.K. Harman, editor, *The Fourth Text REtrieval Conference (TREC-4)*, number 4, pages 1–24, Gaithersburg, Maryland, noviembre 1995. National Institute of Standards and Technology (NIST), Defense Advanced Research Projects Agency (DARPA).

[16] S. Huffman. Acquaintance: Language-independent document categorization by n-grams. In D.K. Vorhees, E.M.; Harman, editor, *The Fourth Text REtrieval Conference (TREC-4)*, number 4, pages 359–372, Gaithersburg, Maryland, noviembre 1995. National Institute of Standards and Technology (NIST), Defense Advanced Research Projects Agency (DARPA).

[17] D.A. Hull and G. Grefenstette. Queryng across languages: A dictionary-based approach to multilingual intormation retrieval. In *SIGIR 96*, volume 47, pages 49–57, 1996.

[18] D. Jurafsky and J. H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing*. Prentice-Hall, NJ, 2000.

[19] T. Z. Kalamboukis. Suffix stripping with moderm greek. *Program*, 29(3):313–321, 1995.

[20] W. Kraaij and R. Pohlmann. Viewing stemming as recall enhancement. In *SIGIR 96*, pages 40–48, 1996.

[21] W. Kraaij and Renée Pohlmann. Porter's stemming algorithm for dutch. In L. G. M. Noordman and W. A. M. de Vroomen, editors, *Informatiewetenschap*, pages 167 –180, Tilburg, 1994. STINFON.

[22] R. Krovetz. Viewing morphology as an inference process. In *SIGIR 93*, pages 191–203, 1993.

[23] J. B. Lovins. Development of a stemming algorithm. *Mechanical Translation and Computational Linguistics*, 11:22–31, 1968.

[24] B. Merialdo. Tagging english text with a probabilistic model. *Computational Linguistic*, 20(2):155–171, 1994.

[25] A. Molina and L. Moreno. Técnicas de análisis parcial en procesamiento del lenguaje natural. Technical Report DSIC-II/30/98, UPV, Departamento de Sistemas Informáticos y Computación, 1998.

[26] L. Moreno Boronat, M. Palomar Sanz, A. Molina Marco, and A. Fernández Rodríguez. *Introducción al Procesamiento del Lenguaje Natural*. Universidad de Alicante, Murcia, 1999.

[27] C. D. Paice. Another stemmer. *SIGIR Forum*, 24(3):56–61, 1990.

[28] F. Pla i Santamaría. *Etiquetado Léxico y Análisis Sintáctico Superficial basado en Modelos Estadísticos*. PhD thesis, Universidad de Valencia, Valencia, 2000.

[29] M. Popovic and P. Willet. The effectiveness of stemming for natural-language access to slovene textual data. *JASIS*, 43:384–390, 1992.

[30] M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, julio 1980.

[31] A. Robertson and P. Willet. Applications of n-grams in textual information systems. *Journal of Documentation*, 54(1):28–47, 1999.

[32] S. Rodríguez and J. Carretero. A formal approach to spanish morphology: the coes tools. In *XII Congreso de la SEPLN*, pages 118–126, Sevilla, 1996.

[33] H. Rodríguez Hontoria. *Filología e Informática: nuevas tendencias en los estudios filológicos*, chapter Técnicas estadísticas en el tratamiento del lenguaje natural, pages 111–140. UAB, Barcelona, 1999.

[34] H. Rodríguez Hontoria. Técnicas basadas en el tratamiento informático de la lengua. *Quark*, (19), Julio-Diciembre 2000.

[35] G. Salton and M. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, New York, 1983.

[36] O. Santana, J. Pérez, F. Carreras, J. Duque, Z. Hernández, and G. Rodríguez. Flanom: Flexionador y lematizador automático de formas nominales. *Lingüística Española Actual*, XXI(2):253–297, 1999.

[37] O. Santana, J. Pérez, Z. Hernández, F. Carreras, and G. Rodríguez. Flaver: Flexionador y lematizador automático de formas verbales. *Lingüística Española Actual*, XIX(2):229–282, 1997.

[38] J. Savoy. Effectiveness of information retrieval systems used in a hypertext environment. *Hypermedia*, 5:23–46, 1993.

[39] J. Savoy. A stemming procedure and stopword list for general french corpora. *JASIS*, 50(10):944–952, 1999.

[40] R. Schinke, A. Robertson, P. Willet, and M. Greengrass. A stemming algorithm for latin text databases. *Journal of Documentation*, 52(2):172–187, 1996.

[41] A. Voutilainen. A syntax-based part-of-speech analyser. In *Procs. of the Conference European of the ACL-95*, Dublin, 1995.

[42] R. et. al. Weischedel. Coping with ambiguity and unknow words through probabilistic models. *Computational Linguistics*, 19(2):359–382, 1993.

[43] E. M. Zamora, J. J. Pollock, and A. Zamora. The use of trigram analysis for spelling error detection. *Information Processing and Management*, (17):305–316, 1981.