Extracción de una terminología multilingüe de Wikipedia

Sergio Cajal Mariñosa

Ingeniería en Informática

Facultat d'Informàtica de Barcelona (FIB)

Universitat Politècnica de Catalunya (UPC)

Director: Horacio Rodríguez Hontoria

Departament de Llenguatges i Sistemes Informàtics

Mayo de 2014

A mis dos Elenas

Índice general

1

Introducción

Sólo podemos ver poco del futuro, pero lo suficiente para darnos cuenta de que hay mucho que hacer.

Alan Turing

1.1. Motivación

Desde hace varias décadas existe la necesidad creciente desde distintos ámbitos de extraer terminologías de textos especializados. Es una tarea que interesa a lingüistas computacionales, traductores, periodistas científicos e ingenieros informáticos. Las aplicaciones de la obtención de terminologías son muchas y diversas:

- Elaboración de glosarios y diccionarios terminológicos que sirvan de referenica a cualquier persona que escriba sobre un tema muy específico.
- Mejora de los sistemas de resumen automático, ya que se tiene conocimiento de cuáles son los términos más relevantes en el dominio semántico del texto que se está resumiendo.
- Traducción automática de textos, ya que ayuda a desambiguar palabras

polisémicas.

- Construcción de ontologías de dominios semánticos concretos.
- Afinado (*tuning*) de recursos de Procesamiento del Lenguaje Natural que se aplican a un dominio semántico concreto.

En Krauthammer et al. (2004) [1] se señala que «la identificación de términos está considerada el cuello de botella en la minería de textos y por lo tanto un campo de investigación interesante en el PLN».

Para aplicaciones como la traducción automática o incluso para elaborar diccionarios terminológicos multilingües como los que existen en el dominio de la medicina resulta especialmente interesante disponer de una terminología multilingüe.

1.1.1. Estado del arte

La tarea de obtención de terminologías se hacía originariamente de forma manual usando expertos en el dominio semántico dado. Con la aplicación de la informática y técnicas de inteligencia artificial en prácticamente todos los ámbitos de la ciencia en las últimas décadas, apareció la posibilidad de automatizar la elaboración de terminologías usando estas técnicas.

La mayoría de aproximaciones existentes se valen como fuente información de los corpus, grandes recopilaciones de textos que sirven como base para casi todos los métodos empíricos del Procesamiento del Lenguaje Natural. Y para ello usan conocimiento lingüístico (dependiente del lenguaje) y técnicas estadísticas. La extracción automática de términos a partir de corpus encuentra varios problemas [2]:

- Identificación de expresiones multipalabra (*MWE*, del inglés *multiword expressions*), es decir, determinar dónde empieza y acaba un término que conste de varias palabras.

- Reconocimiento de *MWE*, es decir, poder determinar si un conjunto de palabras constituyen un término o hay que tomar el significado de cada una por separado.

- Determinar si una unidad léxica que se encuentra en un texto especializado tiene naturaleza de término dentro del dominio del texto o por lo contrario pertenece al lenguaje general.

Cabe señalar que no todos los *MWE* son terminológicos, ni todos los términos son *MWE*. Por ejemplo, la expresión «estirar la pata» representa una sola idea, «morir», distinta del significado de «estirar» y «pata» considerados por separado, mientras que la expresión «estirar la cuerda» no, y por lo tanto no es un término.

Las aproximaciones basadas en corpus se pueden clasificar en tres tipos según la información que usan para determinar qué palabras de un texto son términos [3]:

- Estadísticas: buscan básicamente palabras o secuencias de unidades léxicas que se repiten en un texto, por ejemplo, buscando aquellas que tienen una frecuencia relativa en textos especializados superior a la frecuencia en textos genéricos. Su principal ventaja es que es una aproximación independiente del idioma con el que se esté trabajando.

- Lingüísticas: identifican combinaciones de palabras que siguen determinados patrones sintácticos (p.e. adjetivo + sustantivo o sustantivo + sustantivo). Esta aproximación es fuertemente dependiente del idioma, ya que estos patrones sintácticos varían mucho de un idioma a otro.

- Las que combinan las dos aproximaciones anteriores.

Para el caso específico de las terminologías multilingües la aproximación es en esencia la misma pero se vale como fuente de información de un corpus multilingüe.

1.2. Sobre las terminologías

Para la buena comprensión de la importancia de las terminologías en los diferentes ámbitos que se han enumerado anteriormente es necesario empezar por definir qué son los términos y las terminologías, y ahondar en por qué son importantes.

1.2.1. Definición de término y termhood

Un término es una unidad léxica que designa un concepto en un dominio específico. Puede estar formado por una o por varias palabras que se consideran una unidad a la hora de designar algo. Así, por ejemplo, «proyecto final de carrera» es un término en tanto que designa un concepto más específico que el que denotan las palabras que lo conforman si se toma el significado de cada una por separado.

Quedan fuera del concepto de término las entidades con nombre o *Named Entities* (NE), que son unidades léxicas que representan nombres de personas, organizaciones y lugares así como también expresiones de tiempo, cantidades de dinero, etc. Si se quiere hacer un símil con la ingeniería de software se podría decir que un término designa una clase y una *Named Entity* designa una instancia de esa clase.

Se conoce como *termhood* «el grado en el que una unidad lingüística está relacionada con conceptos específicos del dominio» [4], que en nuestro caso se puede ver también como la probabilidad de que un término forme parte del dominio. Sin embargo, el *termhood* no es una medida discreta sino continua.

1.2.2. Definición de terminología

Una terminología es un conjunto de términos propios de un dominio semántico concreto. La misma palabra designa la ciencia o estudio de los términos, pero

aquí nos quedaremos con la definición anterior. La definición más amplia de qué es una terminología engloba distintas estructuras [5]:

- Glosarios: listas de pares palabra (o conjuntos de palabras) + definición
- Listas de palabras clave o *keywords*: similares a los glosarios pero sin definición y que normalmente representan el subconjunto más relevante de los términos de un dominio.
- Taxonomías: listas de términos organizados de manera jerárquica, como por ejemplo las clasificaciones de seres vivos.
- Tesauros (*thesaurus*), también organizados jerárquicamente, son similares a las taxonomías pero contienen más información de relaciones, sobre todo de sinonimia y de categoría.
- Ontologías, que representan relaciones semánticas de todo tipo, no estrictamente formando una jerarquía.

1.3. Sobre Wikipedia

Wikipedia es proyecto web de enciclopedia multilingüe que se escribe colaborativamente entre usuarios de todo el mundo con la aspiración de abarcar todo el conocimiento humano. Está administrada por la Wikimedia Foundation, una organización sin ánimo de lucro que también se encarga de proyectos hermanos como Wiktionary (diccionario), Wikisources (biblioteca de documentos libres) o Wikitravel (información para viajeros). Tanto Wikipedia como sus proyectos hermanos son wikis, que es como se conoce a las aplicaciones web que permiten crear y modificar contenido colaborativamente. Las wikis usan un lenguaje de marcado (*markup language*) que permite añadir enlaces entre los artículos y contenidos adicionales tales como imágenes, sonidos o enlaces a páginas web externas.

1.3.1. Las entidades básicas de Wikipedia

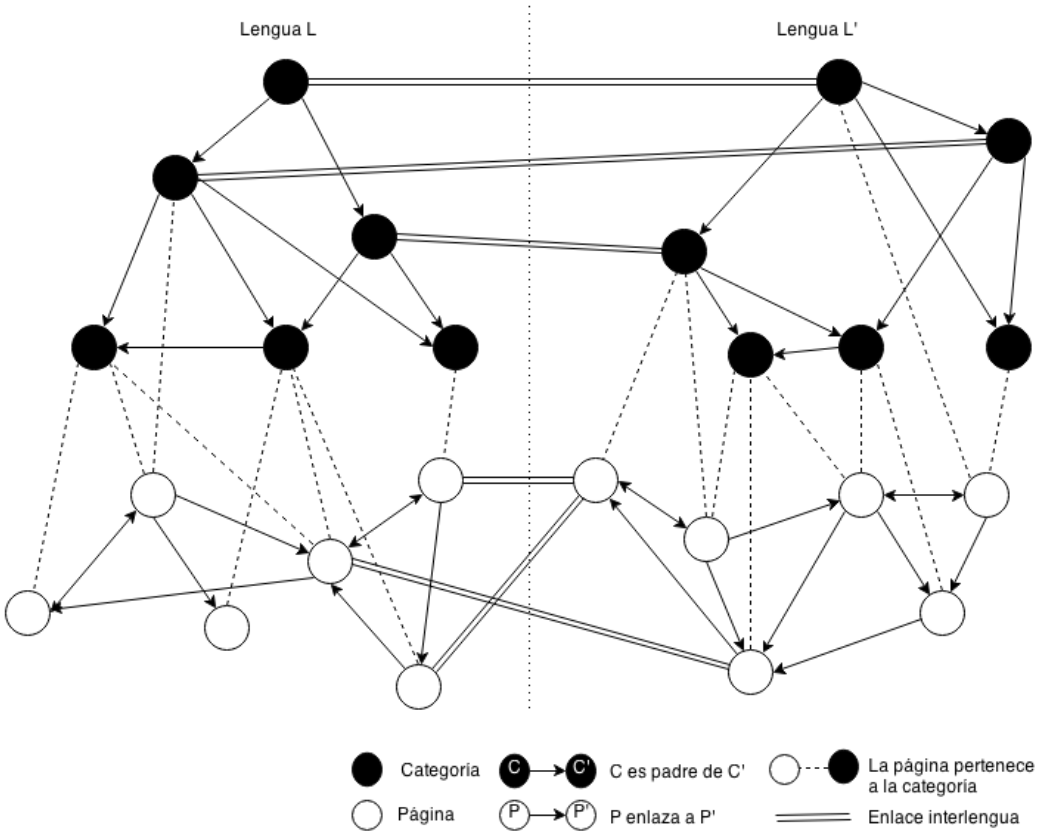


Figura 1.1: Representación de los objetos de Wikipedia y sus relaciones

1.3.1.1. La página

La unidad básica de Wikipedia es la página. Una página es un artículo que explica un concepto, y se compone esencialmente de texto y de información estructurada añadida mediante el lenguaje de marcado. Esta información añadida se puede clasificar en los siguientes grupos:

- Enlaces a otras páginas de Wikipedia mediante el resaltado de palabras de manera similar a como funcionan los enlaces en el lenguaje HTML y la WWW.

- Plantillas, que suelen contener botones para la navegación por páginas similares o englobadas en un mismo tema. Por ejemplo, en la página de Cataluña de la Wikipedia en castellano aparece una plantilla con enlaces que permite ir a otras páginas de Comunidades Autónomas españolas.
- Infoboxes, cajas con forma de tabla que aparecen en la parte superior derecha de algunos páginas mostrando información esquemática sobre el concepto que representan. Por ejemplo, en la página de un país la infobox nos mostrará la bandera de dicho país y datos básicos como población, área, divisa, etc.
- Enlaces a las categorías con las que está etiquetada la página, que se muestran siempre al final de todo
- *Interwiki links*, enlaces a páginas de otros proyectos de la Wikimedia Foundation como Wiktionary.
- Enlaces interlingüísticos, enlaces a la página de Wikipedia equivalente en otro idioma. Se muestran a la izquierda, y son en realidad un tipo de enlace interwiki.
- Imágenes y gráficos, desde fotografías a climogramas o pirámides de población.
- Enlaces web externos a la Wikipedia.

1.3.1.2. La categoría

Cada página está etiquetada con una o más categorías. Las categorías a su vez pertenecen a una o más categorías, y están organizadas jerárquicamente en forma de taxonomía, con una categoría raíz de todas las demás. En principio deberían formar un grafo acíclico dirigido (DAG) pero más adelante veremos que esto no siempre es así.

A pesar de que se podría pensar que la manera como se estructura el conocimiento es igual en todos los idiomas, y de que podría pensarse en un conjunto

de categorías igual para todos los idiomas, esto no es así. Para cada lengua el conjunto de categorías es distinto, si bien la organización general por temas es bastante similar en todas. Del mismo modo que las páginas, las categorías pueden tener enlaces interlingüísticos a la categoría equivalente en otro idioma.

1.3.1.3. Relaciones en Wikipedia

En la figura 1.1 se ilustra la relación descrita entre las entidades de Wikipedia. Las categorías, representadas por las bolas negras, tienen con otras categorías relaciones del tipo supercategoría $\rightarrow$ subcategoría, representadas en el diagrama como arcos dirigidos.

Las bolas blancas representan las páginas, que se relacionan con otras páginas con hiperenlaces equivalentes a los hiperenlaces de una página web convencional, y que se representan en el diagrama también como arcos dirigidos. Las páginas tienen una relación de pertenencia a una categoría, y se puede decir que las categorías contienen páginas. Esta relación se representa en el diagrama con un arco no dirigido marcado con línea discontinua.

Las páginas y categorías puede tener opcionalmente enlaces interlingüísticos a la página o categoría (respectivamente) que representa el mismo concepto. Los enlaces interlingüísticos están representados en el diagrama con una línea doble.

1.4. Objetivo del proyecto

Considerando las necesidades planteadas al principio, los problemas con los que se encuentran las aproximaciones actuales, y las posibilidades que ofrece la Wikipedia, este proyecto propone una nueva aproximación para obtener terminologías en varios idiomas a partir de la información disponible en la Wikipedia.

Concretamente, para un dominio semántico introducido en un idioma concreto por el usuario, se buscará la categoría tope en cada idioma que corresponde

a ese dominio semántico. Se recorrerá el árbol de categorías en cada idioma para obtener todas las categorías que tienen a la categoría dada como ancestro. A continuación, se explorarán las categorías obtenidas para recuperar todas las páginas que pertenecen a ellas. Finalmente se aplicará un algoritmo similar al PageRank de Google para ordenar los términos del conjunto por su relevancia en función de los enlaces entre páginas y entre categorías y páginas, tomando como hipótesis que existe una relación entre la relevancia calculada de este modo y el *termhood* de cada término.

Además, se hará uso de los enlaces interlingüísticos, con la hipótesis de que usar esta información mejorará los resultados. La idea detrás de ello es que si un término es relevante en un idioma, sus traducciones serán relevantes en otros idiomas. De este modo se corregirán en parte las diferencias entre los árboles de categorías de los diferentes idiomas.

La aplicación no da como un resultado una lista cerrada de términos, sino que funciona como *ranker* que ordena los candidatos a términos de mayor a menor por *termhood*. Con la lista ordenada de términos, el usuario podrá seleccionar los N primeros. Dado que están ordenados por una puntuación que representa el *termhood*, cuantos más se seleccionen, más falsos positivos habrá, es decir, más términos que no forman parte del dominio. Queda a decisión del usuario, en función de la finalidad de la terminología, decidir qué grado quiere de precisión (porcentaje de los términos que son correctos) y cobertura (número total de términos).

Este objetivo general se divide en objetivos más específicos que se detallan en los siguientes apartados y se representa gráficamente en el diagrama de la figura 1.2.

1.4.1. Descarga y almacenamiento de las Wikipedias

Para tratar la información desde un programa será necesario descargar la Wikipedia completa en varios idiomas y almacenarla de algún modo que permita

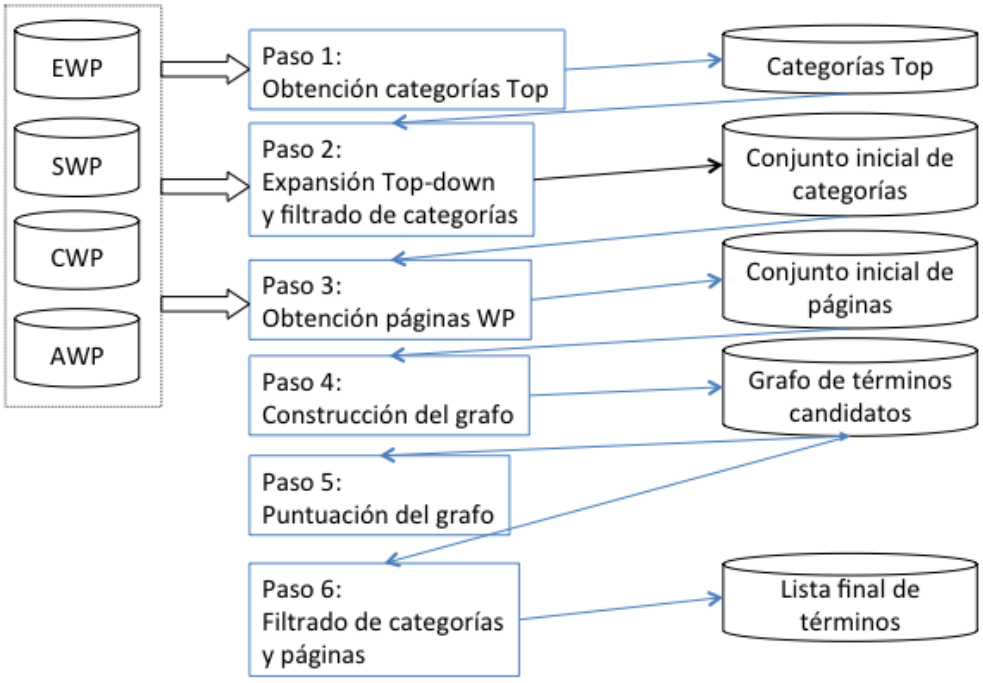


Figura 1.2: Diagrama de arquitectura del sistema

un acceso cómodo y eficiente a los datos. Deberá almacenar también las relaciones langlink entre categorías y páginas que tienen versión en varios idiomas. Es necesario tener un snapshot de las Wikipedias en un determinado punto en el tiempo para que se puedan hacer pruebas que trabajen siempre con los mismos datos, ya que Wikipedia es un proyecto web colaborativo que va cambiando constantemente.

1.4.2. Obtención de un conjunto inicial de términos

Partiendo del nombre de una categoría que designa un dominio semántico determinado en un idioma, se recorrerá el árbol de categorías para obtener recursivamente sus categorías descendientes. A continuación se obtendrán todas las páginas de esas categorías para configurar un conjunto inicial de términos del dominio semántico.

1.4.3. Valoración, filtrado y refinamiento del conjunto de términos

A partir del conjunto inicial será necesario hacer una serie de operaciones que permitan filtrar los términos obtenidos, ante la presunción de que parte de ellos serán falsos positivos que no pertenecen al dominio semántico dado.

1.4.4. Evaluación de los resultados

Será necesario diseñar y desarrollar un entorno de pruebas que permita probar el buen funcionamiento del algoritmo y la corrección de los resultados obtenidos. Para ello habrá que buscar un *gold standard*, una terminología existente para un dominio semántico concreto y que esté verificada por expertos.

1.5. Guía de lectura

Tras el primer capítulo de introducción al problema, el capítulo 2 es de especificación y diseño del sistema. En el capítulo 3 se evalúa el sistema, para ello primero se describe el entorno de pruebas, a continuación se detallan los experimentos que se han hecho y finalmente hay una exposición de los resultados. En el capítulo 4 se describe la planificación temporal y se hace una valoración económica. El capítulo 5 es de conclusiones y opiniones sobre el trabajo futuro que se puede realizar en el ámbito de este proyecto.

También se incluyen cuatro anexos. Uno donde se presentan tablas complementarias de resultados (anexo A), otro con el manual de instalación de la aplicación (anexo B), así como un manual de usuario para usarla (anexo C). Por último se incluye un *paper* que se ha escrito a raíz de este proyecto (anexo D).

2

Especificación y diseño

Un algoritmo hay que verlo para creerlo.

Donald Knuth

En este capítulo se presenta la especificación del sistema desde el punto de vista de la ingeniería del software y se justifica el diseño a la vez que se describe la implementación desde el punto de vista de los diferentes componentes.

En la primera sección (2.1) se presentan los casos de uso de la aplicación, muy sencillos al tratarse de una aplicación orientada a la experimentación en torno a un algoritmo.

En la segunda sección (2.2) se describe y se justifica el diseño e implementación del sistema. Se describen el modelo de datos del sistema, el modelo de diseño, y a continuación se explica cómo actúa el algoritmo en cada uno de los pasos de la extracción de la terminología.

2.1. Especificación

Al ser una aplicación pensada para probar un algoritmo y extraer unos resultados experimentales, no hay muchos casos de uso ni opciones disponibles. Hay un solo

actor en el sistema, que es el usuario que interactúa con la aplicación.

2.1.1. Punto de partida

Para el uso de la aplicación previamente se habrá descargado la Wikipedia en varios idiomas y se habrá almacenado de alguna forma que haga que sea accesible de manera cómoda y eficiente por el programa principal.

2.1.2. Caso de uso principal

La aplicación a desarrollar tendrá un caso de uso principal, en el que intervendrá un usuario que introducirá una palabra que corresponda a una categoría de la Wikipedia en un idioma cualquiera relativa a un dominio semántico dado. La aplicación devolverá como resultado una lista de términos de dicho dominio semántico en todos los idiomas soportados por el sistema. Este caso de uso principal se compondrá de varios pasos:

1. El usuario seleccionará un idioma de entrada de entre los disponibles
2. El usuario introducirá el nombre de una categoría en el idioma seleccionado anteriormente. Se considerará que esta categoría será el tope del dominio considerado, es decir, la raíz de la eventual taxonomía correspondiente al dominio en el grafo de categorías de la lengua considerada. En ese momento el programa iniciará el proceso de obtención del conjunto inicial de categorías.
3. El usuario introducirá cuántos términos quiere obtener, ya sea de manera porcentual al total de los recogidos, de manera absoluta con un número dado o proporcionando un umbral correspondiente a la valoración mínima de los términos seleccionados. El resultado se volcará en un fichero de texto por cada idioma soportado.

2.1.3. Casos de uso secundarios

- Almacenamiento y carga del conjunto de términos: una vez obtenido el conjunto de términos inicial en varios idiomas, el usuario podrá almacenar esos términos en un fichero para recuperarlo en otro momento.
- Configuración de los parámetros del algoritmo de refinamiento: El usuario podrá cambiar los parámetros del algoritmo de refinamiento editando un fichero de texto

2.2. Diseño

En esta sección se detalla cómo se ha implementado la aplicación que extrae las terminologías de la Wikipedia. Primero se describe la base de datos que replica el contenido de Wikipedia, luego se describen las clases y objetos principales que conforman el modelo de dominio y a continuación se describe y se justifica el algoritmo que se ha implementado.

El algoritmo para obtener la lista de términos en cada idioma ordenados por *termhood* se puede dividir en los pasos siguientes, que se corresponden aproximadamente con los representados en la figura 1.2:

1. Obtención del conjunto inicial de categorías
2. Obtención de las páginas
3. Creación de los enlaces interlingüísticos
4. Ordenación de los términos por relevancia

Cada una de estas fases se describe en un apartado distinto dentro de esta sección.

2.2.1. Base de datos

Wikipedia es un proyecto web cooperativo en el que participan diariamente como editores miles de usuarios, por lo que su contenido va cambiando y ampliándose continuamente. Dado el objetivo de este proyecto no solo de desarrollar un algoritmo para obtener una terminología sino también de evaluarlo, es necesario contar con un snapshot, una fotografía fija de Wikipedia en un momento dado. Para ello hay descargar la Wikipedia en cada idioma en un momento dado y almacenarla en el disco.

Para el almacenamiento se ha optado por una base de datos MySQL para cada idioma que queramos tratar. Así, para cada idioma tendremos una base de datos con las tablas que se describen a continuación.

Cuadro 2.1: Estructura de las tablas principales

Category	
id	bigint(20)
pageId	int(11)
name	varchar(255)
depth	int(11)

Page	
id	bigint(20)
pageId	int(11)
name	varchar(255)
text	longtext
isDisambiguation	bit(1)

Cuadro 2.2: Estructura de las tablas asociativas

category_inlinks	
id	bigint(20)
inLinks	int(11)

category_outlinks	
id	bigint(20)
outLinks	int(11)

category_pages	
id	bigint(20)
pages	int(11)

page_inlinks	
id	bigint(20)
inLinks	int(11)

page_outlinks	
id	bigint(20)
outLinks	int(11)

page_categories	
id	bigint(20)
categories	int(11)

Cuadro 2.3: Estructura de la tabla *langlinks*

langlinks	
ll_from	int(8)
ll_lang	varchar(20)
ll_title	varchar(255)

Cuadro 2.4: Estructura de la tabla PageMapLine

PageMapLine	
id	bigint(20)
name	varchar(255)
pageID	int(11)
stem	varchar(255)
lemma	varchar(255)

2.2.1.1. Descripción de las tablas

- Tablas principales, detalladas en el cuadro 2.1:
  - `Category`: corresponde a las categorías de Wikipedia. Cada categoría tiene un identificador numérico único para cada lengua.
  - `Page`: corresponde a las páginas o artículos de Wikipedia. Del mismo modo que las categorías, cada página tiene un identificador numérico.
- Tablas asociativas que almacenan relaciones, detalladas en la tabla 2.2
  - `category_inlinks`: relaciona una categoría con sus supercategorías
  - `category_outlinks`: relaciona una categoría con sus subcategorías
  - `category_pages`: relaciona una categoría con las páginas que pertenecen a ella
  - `page_inlinks`: relaciona una página con las páginas que la enlazan

- `page_outlinks`: relaciona una página con las páginas enlazadas desde el texto del artículo
- `page_categories`: relaciona una página con las categorías a las que pertenece
- `langlinks`: relaciona una página o categoría con su equivalente en otro idioma. Esta relación es por nombre, por lo que hay que buscar en la BD de la lengua de destino. Esta tabla está detallada en el cuadro 2.3

- Tabla auxiliar PageMapLine para permitir el acceso rápido al nombre de una página, ya que las consultas SQL por nombre a la tabla Page tienen un coste inasumible. El detalle de los campos se ve en el cuadro 2.4.

2.2.1.2. Proyecto JWPL

La estructura general de la base de datos se ha obtenido a partir de la que usa el proyecto Java Wikipedia Library (JWPL) [6] desarrollado por un grupo de trabajo de la Technische Universität Darmstadt dirigido por la Dra. Iryna Gurevych para el acceso a los datos volcados en los dumps que publica la Fundación Wikimedia.

Dentro del proyecto del proyecto JWPL existe un componente, DataMachine, que convierte los dumps publicados por Wikipedia[1] en dumps de tablas más estructuradas, pensadas para ser importadas a una base de datos MySQL. DataMachine se distribuye en un jar que incluye dependencias y funciona como una aplicación standalone, así que la hemos podido usar para crear nuestra base de datos inicial sin necesidad de mezclar código Java con el código Python que se ha usado para la implementación de nuestra aplicación.

En los scripts obtenidos de la aplicación DataMachine las tablas se crean con el motor MyISAM, que no soporta claves foráneas, por ello no se especifican las

[1] http://dumps.wikimedia.org/backup-index.html

relaciones en el diagrama de la base de datos a pesar de que existen de forma conceptual.

2.2.1.3. Tabla de langlinks

El proyecto JWPL no soporta langlinks[2], los enlaces interlingüísticos que relacionan páginas y categorías equivalentes entre diferentes idiomas. Por tanto, ha sido necesario añadir esa tabla manualmente a partir del fichero `langlinks.sql.gz`. El fichero contiene directamente la operación `CREATE TABLE` que crea la tabla y las operaciones `INSERT` que la llenan con los valores, pero hay que cambiar los valores del charset por defecto a Unicode para ser consistentes con el resto de tablas. Además, es conveniente cambiar el motor de base de datos a MyISAM, ya que se recomienda no mezclar distintos motores en una misma base de datos.

2.2.1.4. Columna depth en la tabla Category

En la tabla Category se ha añadido una columna «depth» que representa la profundidad absoluta de una categoría en el grafo de categorías, es decir, la distancia desde dicha categoría a la categoría raíz de Wikipedia. Esta profundidad absoluta se usará en la extracción del conjunto inicial de categorías, como se verá más adelante.

Como categoría raíz no conviene tomar en todos los casos la categoría raíz real («Índice de categorías» en castellano, «Contents» en inglés), ya que dentro de esta se incluyen metacategorías del tipo «Anexos» o «Ayuda» que no se corresponden con el carácter semántico que damos a las categorías. Por tanto, es mejor tomar como categoría raíz, cuando exista, la categoría que engloba concretamente los artículos («Articles» en inglés, «Artículos» en castellano).

El cálculo del valor de esta columna «depth» se debe hacer al principio como

[2]Debido a que el fichero de langlinks se incorporó a los dumps con posterioridad a la creación de JWPL

preproceso tras importar la base de datos, y consiste sencillamente en un hacer un recorrido en anchura del grafo en el que se propague la profundidad desde el nodo raíz, actualizando la de cada nodo cuando la propagada sea menor. Podemos ver el pseudocódigo de la implementación en el algoritmo 1.

Algoritmo 1 Cálculo de la profundidad absoluta

```
queue ← [(rootcat,0)]
while queue ≠ ∅ do
    cat, depth ← queue.pop(0)
    former_depth ← get_depth(cat)
    if former_depth = −1 ∨ depth < former_depth then
        set_depth(cat, depth)
        for all subcat ∈ subcats(cat) do
            queue.append((subcat, depth + 1))
        end for
    end if
end while
```

2.2.2. Modelo de dominio

En la capa de dominio se mantiene en memoria una estructura representada por el diagrama que se puede ver en la figura 2.1. Para cada lengua, se rellena un conjunto de categorías y otro de páginas, y entre ellos existen las relaciones de Wikipedia representadas en el capítulo de introducción en la figura 1.1.

2.2.2.1. Descripción de los elementos

Los objetos de dominio que se mantienen en memoria y representan el estado de la aplicación son esencialmente dos, que contienen a su vez a los demás: el `catset` y el `pageset`:

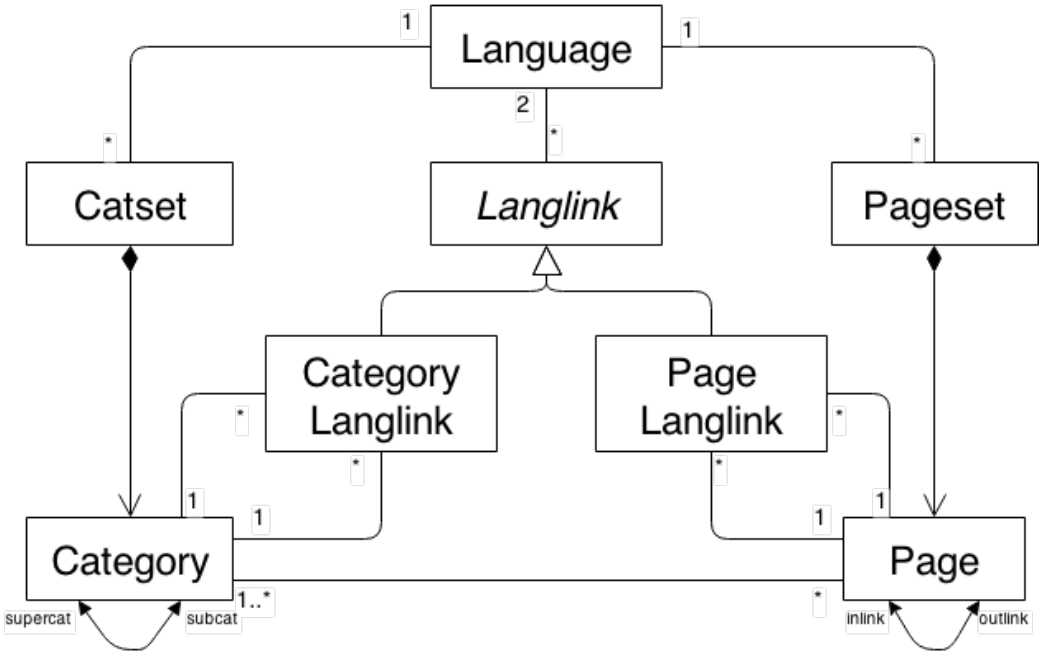


Figura 2.1: Diagrama de dominio

- `catset`: conjunto de categorías pertenecientes al dominio semántico introducido por el usuario. Está implementado con un diccionario cuya clave es el código de idioma de Wikipedia («es» en el caso del castellano, por ejemplo), y que para cada idioma contiene otro diccionario para almacenar las categorías recuperadas, en este caso usando como clave el id de la categoría, que dentro del ámbito de cada idioma. De este modo podemos acceder en tiempo constante a una categoría a partir de su id escribiendo simplemente catset[lang][catID]. Podemos ver la descripción de la clase «Category» y sus atributos en el cuadro 2.5.

- `pageset`: conjunto de páginas que pertenecen a las categorías presentes en el conjunto `catset`. De un modo análogo a este, está implementado con un diccionario cuya clave es el idioma, y contiene otro diccionario con el id de la página como clave para poder acceder en tiempo constante. Podemos ver la descripción de la clase «Page» y sus atributos en el cuadro 2.6.

Cuadro 2.5: Estructura de la clase Category

Category	
catID	identificador de la categoría en la BD
title	nombre de la categoría
subcats	subcategorías o categorías hijas
supercats	supercategorías o categorías padre
langlinks	diccionario con los enlaces interlingüísticos para cada lengua
depth	distancia a la categoría raíz introducida por el usuario
absolute_depth	distancia a la categoría raíz de Wikipedia
score	puntuación según el algoritmo de PageRank

Las relaciones 1:N que vemos en el diagrama 2.1 están materializadas en el código de la aplicación como listas de identificadores. En el caso de los enlaces interlingüísticos, como diccionario, para distinguir entre los idiomas de destino.

2.2.3. Obtención del conjunto inicial de categorías

Como se ha dicho en la especificación, para obtener la terminología para un dominio semántico dado, el usuario introduce en uno de los idiomas soportados y descargados el nombre de la categoría que se corresponde con dicho dominio semántico. A partir de esa categoría que llamaremos *top*, se obtienen sus categorías descendientes en el grafo de categorías y a continuación las páginas pertenecientes a estas categorías.

Por tanto al inicio el algoritmo tiene que recorrer el grafo de categorías para cada lengua y almacenar todas sus descendientes, creando para cada una un objeto Category como el descrito anteriormente y colocándolo en el diccionario `catset` que acabamos de ver. Según dicta el sentido común y hemos explicado en el punto 1.3, el grafo de categorías debería tener forma de grafo dirigido acíclico (DAG en sus siglas en inglés), por lo que este recorrido debería ser parecido al recorrido de una estructura de datos tipo árbol, pero teniendo en cuenta los

Cuadro 2.6: Estructura de la clase Page

Page	
pageID	identificador de la página en la BD
title	nombre de la página
cats	categorías a las que pertenece la página
inlinks	páginas que enlazan a esta página
outlinks	páginas a las que enlaza esta página
langlinks	diccionario con los enlaces interlingüísticos para cada lengua
score	puntuación según el algoritmo de PageRank
depth	distancia a la categoría raíz de Wikipedia

nodos visitados, ya que según la definición de las categorías wn Wikipedia puede haber varios caminos desde una categoría a una de sus descendientes.

Lo que sigue es una descripción de los problemas encontrados en la obtención de este conjunto inicial junto con los *workarounds* que se han buscado para solucionarlos, así como una explicación detallada de las partes en las que se ha dividido el desarrollo.

2.2.3.1. Ciclos en el grafo de categorías

Al programar la aplicación, las pruebas de obtención del conjunto inicial de categorías con dominios pequeños y muy concretos dieron el resultado esperado, y se obtuvo un conjunto de categorías pequeño y, bajo observación a simple vista, relacionado con el dominio semántico para todos los idiomas estudiados. Pero al realizar este recorrido para un dominio más amplio, se observó que el grafo de categorías contiene una gran cantidad de ciclos y, por lo tanto, no tiene la forma esperada de DAG. A modo de ejemplo, vemos que hay un ciclo en las siguientes categorías (la flecha representa que pasamos de categoría a subcategoría):

$$\text{Drogas} \rightarrow \text{Drogas y Derecho} \rightarrow \text{Narcotráfico} \rightarrow \text{Drogas}$$

El hecho de que haya ciclos, considerando qué son y qué representan las categorías en Wikipedia, parece carente de sentido, pero si tenemos en cuenta que es un proyecto colaborativo hecho por miles de usuarios, se entiende que pueda haber errores de este tipo. Se deduce que no hay ningún mecanismo automático en el software Mediawiki que prevenga esta situación.

En nuestra aplicación estos casos se han tratado sencillamente guardando el camino de categorías desde la categoría *top* a la que estamos visitando, comprobando que cada nodo nuevo no esté en dicho camino y descartando los que sí lo están.

Esta situación anómala en el grafo de categorías se da en mayor medida en Wikipedias grandes como la inglesa o la castellana, y en menor medida por ejemplo en la catalana, que además de tener menos categorías y artículos está escrita por un número reducido de editores, lo que contribuye a su consistencia.

2.2.3.2. Saltos atrás en el grafo de categorías

A parte de la situación descrita en el apartado anterior, se observó un comportamiento anómalo más grave al recorrer en la Wikipedia inglesa. Concretamente se observó que, al recorrer los descendientes de la categoría «Medicine», se recuperaban el 90 % del total de categorías. Es decir, por errores e inconsistencias como la anterior relativa a los ciclos, el 90 % de las categorías tienen entre sus supercategorías la categoría «Medicine». Esto se detectó sobre todo en la Wikipedia inglesa y para dominios que están cerca de la categoría raíz, no tanto en dominios más específicos.

Este fenómeno se da porque en multitud de puntos hay categorías que contienen subcategorías que en realidad se corresponden con dominios mucho más genéricos y que no tienen que ver estrictamente con el dominio en cuestión. Un ejemplo claro se da en la Wikipedia en inglés y para el dominio de la vulcanología, que se corresponde con la categoría «Volcanology», donde se observa la siguiente cadena de saltos (nuevamente, la flecha representa un salto de categoría a subcategoría):

Volcanology → Volcanoes → Volcanic islands → Iceland

Por tanto, al explorar desde la categoría *top* «Vulcanology» obtendríamos todas las categorías y páginas que tengan que ver con Islandia: su economía, su historia, su cultura; aunque nada tengan que ver con volcanes.

Esta situación implica un problema mucho más grande que el visto en el anterior apartado, ya que requiere que se plantee algún tipo de heurística o criterio para podar el recorrido por el grafo de categorías siguiendo algún criterio, evitando recorrer algunas aristas cuando se dé una circunstancia concreta. Además, al ser la implementación de esta aplicación independiente del dominio y del lenguaje, no hay más información disponible que la que recuperamos de la BD y que tenemos en memoria en las clases descritas anteriormente.

En concreto, las variables a nuestro alcance para descartar o no un nodo cuando recorremos el grafo de categorías son:

- La profundidad relativa del nodo visitado en ese momento respecto a la categoría *top*. Es igual a la longitud del camino desde dicha categoría.
- La profundidad absoluta del nodo respecto a la categoría raíz de Wikipedia, que hemos precalculado como se explica en el apartado 2.2.1.4.
- La profundidad absoluta del nodo padre del nodo actual, que nos puede indicar si estamos dando un «salto atrás» en el grafo pasando a una categoría que está mucho más cerca de la raíz y que por lo tanto puede ser incorrecta.
- La profundidad absoluta del nodo que corresponde a la categoría *top* desde la que hemos empezado el recorrido.

Con la relación entre la profundidad absoluta de la categoría y la de su categoría padre (supercategoría) podemos intentar evitar saltos atrás, evaluando condiciones tales como:

- Que la profundidad absoluta de una categoría no pueda ser inferior a la de su supercategoría, es decir, filtrando si la siguiente condición es cierta

$$parent\_depth > cat\_depth$$

Esta condición es demasiado restrictiva.

- Que la diferencia entre la profundidad absoluta de la nueva categoría que exploramos y la de su padre no pueda ser mayor que la mitad de la diferencia entre la profundidad del padre y la profundidad de la categoría *top*.

$$(parent\_depth - rootcat\_depth)/2{,}0 > (parent\_depth - cat\_depth)$$

Vemos que comparando la profundidad absoluta de un nodo con la de su padre se evitan algunos de estos «saltos atrás» incómodos, pero no se consigue atajar el problema de raíz, bien por podar en exceso, bien por quedarse corto.

Otra aproximación consiste en partir de la hipótesis de que para toda categorías relacionada con el dominio semántico que estamos tratando, el camino más corto a la raíz de categorías de Wikipedia pasará por la categoría *top* que hemos fijado al principio. Si este supuesto se da, la longitud del camino de una categoría a la categoría *top* será igual a la diferencia de las profundidades absolutas. Esto es algo que podemos comprobar sin apenas coste porque el camino desde la categoría *top* a la que estamos visitando ya se propaga por el recorrido recursivo para comprobar si hay ciclos (véase apartado anterior).

Como este criterio se puede ver como demasiado estricto, se ha añadido una constante para que la diferencia entre longitud del camino y diferencia de profundidades no sea estrictamente igual, sino que esta restricción pueda ser algo más laxa.

La implementación función que retorna si una categoría visitada es válida o no en función de la profundidad considerando la hipótesis explicada se detalla en el algoritmo 2.

El primer criterio que se aplica para descartar una categoría es que su profundidad absoluta sea menor que la profundidad absoluta de la categoría *top*, lo cual indicaría que está más cerca de la categoría raíz de Wikipedia, y por tanto es improbable que forme parte del dominio semántico.

Algoritmo 2 Función que retorna si una categoría es válida

```
function VALID_DEPTH(cat_depth, topcat_depth, relative_depth)
    if cat_depth < topcat_depth then
        return false
    end if
    if relative_depth > (cat_depth − topcat_depth + depth_diff_thr) then
        return false
    end if
    return true
end function
```

La constante `depth_diff_thr` («depth difference threshold») es la que permite que la diferencia entre las profundidades no sea cero sino que pueda tomar algún valor positivo para no ser tan restrictivos. Se ha encontrado empíricamente que para obtener el máximo posible de categorías del dominio sin un gran número de falsos positivos el valor óptimo de la constante es 1.

2.2.3.3. Recorrido del grafo

En el algoritmo 3 se detalla la implementación del algoritmo de recorrido del grafo de categorías teniendo en cuenta los problemas considerados. La función auxiliar `get_info()` se encarga de obtener de base de datos y rellenar con ella un objeto del tipo Category. La variable `path` almacena el camino desde la categoría *top* hasta esa categoría.

2.2.4. Obtención de las páginas

En este punto el conjunto de categorías `catset` de cada idioma contiene las categorías que consideramos que forman parte del dominio. Este paso consiste simplemente en recorrer ese conjunto, obtener para cada categoría todas las páginas que contiene y añadirlas al conjunto `pageset`. Se considera que si una

Algoritmo 3 Obtención del conjunto inicial de categorías

```
queue ← [(topcat.id,∅)]
topcat_depth ← topcat.depth
while queue ≠ ∅ do
    cat_id, path ← queue.pop(0)
    if cat_id ∉ path ∧ cat_id ∉ catset then
        cat ← get_info(cat_id)
        if valid_depth(cat.abs_depth, topcat_depth, length(path)) then
            catset[cat_id] ← cat
            for all subcat_id ∈ subcats(cat) do
                queue.append((subcat_id,path.append(cat_id)))
            end for
        end if
    end if
end while
```

categoría forma parte del dominio, las páginas que contiene también forman parte de él.

2.2.4.1. Filtrado de Named Entities

Como hemos visto en el apartado apartado 1.2.1, la definición de término no incluye las llamadas *Named Entities*, que en Wikipedia son muy numerosas. El reconocimiento de NE es un campo del Procesamiento del Lenguaje Natural muy extenso y que queda fuera del ámbito de este proyecto. Sin embargo, podemos aplicar algún criterio superficial que permite filtrar muchas páginas que representan NE.

En la mayoría de lenguas que usan el alfabeto latino, la mayúsucula se usa para marcar el inicio de una palabra que designa un nombre propio, así que podemos usar esta información para filtrar las páginas de lugares o biografías de personas. Sin embargo, las páginas de Wikipedia empiezan todas por letra mayúscula, por

lo que la primera palabra no se puede tener en cuenta, pero sí las demás.

Así que antes de añadir una página se comprueba mediante una expresión regular que ninguna palabra que no sea la primera empieza por letra mayúscula.

2.2.5. Creación de los enlaces interlingüísticos

En este punto disponemos de un conjunto de categorías `catset` y un conjunto de páginas `catset` para cada idioma, pero no existe ningún enlace interlingüístico, es decir, ninguna relación entre una categoría o página en un idioma y su equivalente en otro idioma, que son las únicas relaciones del diagrama presentado en la figura 1.1 que nos falta por representar.

Los enlaces interlingüísticos padecen los mismos problemas que el resto de entidades de Wikipedia, y es que al ser editados manualmente por humanos existen diversos errores e inconsistencias [7], principalmente de dos tipos:

- Enlaces no recíprocos, es decir, una página p_L en un idioma L tiene un enlace interlingüístico a una página $p_{L'}$ en un idioma L', pero $p_{L'}$ no tiene ningún enlace interlingüístico para la lengua L.
- Hay inconsistencia entre enlace interlingüístico entrante y saliente, es decir, una página p_L tiene un enlace a una página $p_{L'}$, pero el enlace recíproco de $p_{L'}$ para la lengua L apunta a una tercera página p'_L distinta de p_L. Aunque raro, este caso se da para algunas palabras que tienen significados ligeramente diferentes en cada idioma y que no tienen una traducción exacta.

En el *paper* citado se detalla un complejo método para, literalmente, «desenmarañar» los enlaces intelingüísticos en Wikipedia. Dado que la función los enlaces interlingüísticos en esta aplicación es enriquecer el grafo de relaciones pero no tienen un impacto directo muy grande en los términos obtenidos, se ha considerado que tratar este problema queda fuera del ámbito de este proyecto, por lo que ignoraremos estas inconsistencias y la implementación supondrá que no existen.

Para crear estos enlaces interlingüísticos recorreremos todos los elementos de los conjuntos `catset` y `pageset`. Para cada elemento, consultaremos en la tabla `langlinks` si existe una versión en cada uno de los otros idiomas almacenados, y si es así se busca el id del elemento en la otra base de datos y se almacena en el diccionario `langlinks` del objeto. Sólo se almacena un enlace interlingüístico si ya hemos obtenido el elemento (categoría o página) para el otro idioma, es decir, si existe en el `catset` o `pageset` del otro idioma.

El recorrido se representa en el algoritmo 4. En este caso se obtienen los enlaces interlingüísticos para las páginas, pero la implementación es análoga para el caso de las categorías.

Algoritmo 4 Creación de los enlaces interlingüísticos

```
for all lang ∈ langs do
    other_langs ← langs \{lang}
    for all target_lang ∈ other_langs do
        for all s doource_page ∈ pageset[lang]
            target_name ← get_page_translation(page.id, target_lang)
            if target_name ≠ ∅ then
                target_id ← get_pageID(target_name)
                if target_id ∈ pageset[target_lang] then
                    source_page.langlinks[target_lang] ← target_id
                    target_page ← pageset[target_lang][target_id]
                    target_page.langlinks[source_lang] ← source_page.pageID
                end if
            end if
        end for
    end for
end for
```

2.2.6. Ordenación por termhood

En este punto del algoritmo el conjunto de términos ya se ha obtenido completo y está en memoria, y todas las relaciones posibles entre los elementos están representadas.

El siguiente paso es ordenar los términos por su *termhood*, es decir, por el grado de pertenencia al dominio semántico inicial que tienen. Para ordenar los términos se parte de la hipótesis de que existe una relación entre el *termhood* de un elemento y la puntuación obtenida tras aplicar un algoritmo de relevancia como es el PageRank de Google.

En este apartado explicaremos cómo funciona y en qué se basa el algoritmo de PageRank de Google, y cómo lo hemos adaptado a la estructura de Wikipedia y a las relaciones que tenemos almacenadas, que son distintas a las relaciones de vínculo que hay entre las páginas de la World Wide Web.

2.2.6.1. El PageRank de Google

PageRank es el algoritmo que forma el núcleo del buscador Google y que en su día marcó una gran diferencia en calidad de resultados respecto al resto de buscadores web. Ordena las páginas web de Internet por relevancia partiendo de una premisa muy simple: una página es más relevante cuantos más páginas relevantes la enlacen.

PageRank usa como modelo de datos la representación de la World Wide Web como un grafo donde las páginas son nodos y los hiperenlaces son arcos del grafo. Como los hiperenlaces tienen un origen y un destino, se trata de un grafo dirigido.

Como hemos dicho, la relevancia de una página se define recursivamente y depende de la relevancia de las páginas que la enlazan. Se parte de un estado inicial donde a todos los nodos se les asigna una puntuación igual, que puede ser 1 o $1/N$. Entonces se empieza un algoritmo iterativo en el que en cada iteración

cada página u recibe por cada enlace entrante del conjunto B la puntuación PR de un nodo v dividida por el número de enlaces salientes L de dicho nodo v. Es decir, en cada iteración cada nodo reparte su puntuación en ese momento entre los otros nodos apuntados por él. La puntuación de cada nodo en cada iteración se representa con la siguiente fórmula:

$$PR(u) = \sum_{v \in B_u} \frac{PR(v)}{L(v)} \tag{2.1}$$

Esta es una versión simplificada del algoritmo. El algoritmo completo incorpora un factor extra d, el llamado *damping factor*, ya que PageRank representa en realidad la probabilidad de caer en una página con un usuario imaginario que navega por la web haciendo clic en enlaces aleatorios. Este usuario dejará de hacer clic en algún momento, y la probabilidad de que deje de navegar en la página actual está representada por el *damping factor*. La fórmula del PageRank de una página incluyendo ese factor d es la siguiente:

$$PR(p_i) = \frac{1-d}{N} + d \times \sum_{p_j \in M(p_i)} \frac{PR(p_j)}{L(p_j)} \tag{2.2}$$

donde $p_1, p_2, ..., p_N$ son las páginas que se están considerando, $M(p_i)$ son las páginas que enlazan a p_i, $L(p_j)$ es el número de enlaces salientes de la página p_j y N es el número total de páginas.

Esta última fórmula incluyendo el *damping factor* se ajusta a un modelo que representa un usuario que navega por la web, que es el ámbito en el que Google aplica el PageRank. Como nuestra aplicación no pretende representar ese usuario imaginario, hemos decidido ignorar ese *damping factor*.

Por último cabe destacar que el algoritmo de PageRank posee entre sus múltiples propiedades dos que nos resultan especialmente interesantes y que están relacionadas entre ellas:

- La suma de las puntuaciones de los nodos es siempre la misma. En el caso de haber empezado con una puntuación inicial de 1 este sumatorio de puntuaciones será siempre N, en el caso de empezar con $1/N$ el sumatorio será siempre 1.
- Es convergente, lo que garantiza que con una implementación iterativa en algún momento se llega a un estado estable en el que las puntuaciones no varían dentro de un umbral pequeño.

2.2.6.2. Aplicación del PageRank básico a nuestro sistema

En nuestra aplicación el grafo equivalente al grafo de páginas web sigue el modelo representado en la figura 1.1 de la introducción y en el diagrama de dominio de la figura 2.1. Las páginas web equivalen a nuestras páginas y categorías, y los arcos de hiperenlace son en este caso las relaciones entre categorías y páginas, así como los enlaces interlingüísticos.

El algoritmo de PageRank se puede implementar iterativamente, partiendo como hemos dicho de un estado inicial en el que se otorga a cada nodo una puntuación inicial de 1. En la implementación de Google este valor inicial es de $1/N$, pero esto no es relevante de cada a la ordenación final por puntuación. Lo importante es que todos los nodos empiecen con la misma puntuación:

$$PR_0(u) = 1 \tag{2.3}$$

En cada iteración se calcula el PR de cada categoría c con la siguiente fórmula, que es una adaptación del PageRank simplificado descrito en la fórmula 2.1 de la siguiente manera:

$$PR(c) = \sum_{sbc \in subcats_c} \frac{PR(sbc)}{L(sbc)} + \sum_{spc \in supercats_c} \frac{PR(spc)}{L(spc)} + \sum_{p \in pages_c} \frac{PR(p)}{L(p)} + \sum_{ll \in langlinks_c} \frac{PR(ll)}{L(ll)} \tag{2.4}$$

donde $subcats$ son las subcategorías de la categoría c, $supercats$ son las supercategorías, $pages$ son las páginas contenidas en c y $langlinks$ son las categorías en otro idioma que tienen un enlace interlingüístico con c.

De forma parecida la adaptación de la fórmula para el caso de las páginas queda de la forma siguiente:

$$PR(p) = \sum_{il \in inlinks_p} \frac{PR(il)}{L(il)} + \sum_{c \in categories_p} \frac{PR(c)}{L(c)} + \sum_{ll \in langlinks_p} \frac{PR(ll)}{L(ll)} \quad (2.5)$$

donde $inlinks$ son los enlaces entrantes provenientes de otras páginas, $categories$ son las categorías a las que pertenece p y $langlinks$ son las páginas en otro idioma que tienen un enlace interlingüístico con p.

La expresión L de la fórmula, que en el PageRank original representaba el número de enlaces salientes, en este caso representará, para el caso de las categorías:

$$L(c) = |subcats| + |supercats| + |pages| + |langlinks| \quad (2.6)$$

Y de forma análoga para las páginas:

$$L(p) = |outlinks| + |cats| + |langlinks| \quad (2.7)$$

Este algoritmo iterativo va convergiendo hacia un estado en el que la puntuación de cada página y categoría es estable. Consideraremos que esta puntuación es estable cuando de una iteración a la siguiente no varíe más de un determinado valor e, que se ha fijado empíricamente en $0,01$. Así que en cada iteración hay una comprobación de que la puntuación aún no es estable. El algoritmo 5 detalla la implementación de esta comprobación.

Algoritmo 5 Función que retorna si la puntuación de PageRank aún es inestable

```
function PAGE_RANK_IS_UNSTABLE
    for all lang ∈ langs do
        for all page ∈ pageset[lang] do
            if |page.new_score - page.score| > 0.01 then
                return true
            end if
        end for
        for all cat ∈ catset[lang] do
            if |cat.new_score - cat.score| > 0.01 then
                return true
            end if
        end for
    end for
    return false
end function
```

2.2.6.3. Modificación del PageRank

En el algoritmo de PageRank original todas las relaciones son iguales, pero en nuestra aplicación tenemos diferentes tipos de relaciones, que son las siguientes:

- supercategoría $\longleftrightarrow$ subcategoría
- página $\longleftrightarrow$ categoría
- página $\rightarrow$ página
- enlace interlingüístico

Siendo relaciones de diferente tipo, se puede pensar que dándole un peso distinto a algunas de estas relaciones se puede conseguir un algoritmo de PageRank que se ajuste más a la representación de *termhood* que estamos buscando. La intuición nos dice que las relaciones entre categoría y página deberían tener un peso más importante, ya que si una página pertenece a una categoría con mucho

termhood, es bastante probable que forme parte del dominio.

Una manera que intuitivamente viene a la cabeza es modificar el peso inicial que se concede a cada categoría y página. Sin embargo, tras probar esto se observó que no tiene efecto en la puntuación final en términos relativos (es decir, el orden sigue siendo el mismo), ya que esa puntuación inicial «hinchada» se acaba distribuyendo a todos los nodos.

La otra opción que queda es aplicar un factor multiplicador a la puntuación que recibe un nodo de otro en función de qué tipo de relación de las que hemos listado sea. Esta es una operación delicada porque el algoritmo original converge en un punto, por lo que hay que garantizar que esta convergencia se conserve.

Para conservar la convergencia hay que asegurarse de que se conserva la propiedad de PageRank por la cual la suma de todas las puntuaciones (la relevancia total del sistema) se mantiene constante a lo largo de todas las iteraciones. Por lo que si aplicamos un factor que afecta al numerador de una fracción, hay que aplicarlo también a su denominador.

Veamos cómo hemos conseguido esto con el ejemplo del cálculo de puntuación de una página. Las páginas reciben puntuación de tres formas:

- il: de los enlaces entrantes de otras páginas (*inlinks*).
- cp: de las categorías a las que pertenece la página.
- ll: la proveniente de los *langlinks* o enlaces interlingüísticos.

Para cada tipo de enlace e aplicamos un factor distinto F_e:

$$PR(p) = F_{il} \times \sum_{il \in inlinks_p} \frac{PR(il)}{L(il)} + F_{cp} \times \sum_{c \in categories_p} \frac{PR(c)}{L(c)} + F_{ll} \times \sum_{ll \in langlinks_p} \frac{PR(ll)}{L(ll)} \tag{2.8}$$

Para las categorías el cálculo es similar. En este caso las categorías reciben puntuación de cuatro formas:

- sbc: de las subcategorías o categorías hijas.
- spc: de las supercategorías o categorías padre.
- pc: de las páginas que pertenecen a la categoría.
- ll: la proveniente de los *langlinks* o enlaces interlingüísticos.

Para cada tipo de enlace e aplicamos un factor distinto F_e:

$$PR(c) = F_{sbc} \times \sum_{sbc \in subcats_c} \frac{PR(sbc)}{L(sbc)} + F_{spc} \times \sum_{spc \in supercats_c} \frac{PR(spc)}{L(spc)} + F_{pc} \times \sum_{p \in pages_c} \frac{PR(p)}{L(p)} + F_{ll} \times \sum_{ll \in langlinks_c} \frac{PR(ll)}{L(ll)} \quad (2.9)$$

Estos factores, al actuar multiplicando, rompen la propiedad de PageRank mencionada anteriormente por la cual la suma de las puntuaciones es constante, y disparan el valor de esas puntuaciones echando por tierra la convergencia del algoritmo. Por ello, para equilibrar ese factor, lo que se ha hecho ha sido aplicar también un factor al denominador de cada sumando de la fórmula, que es la expresión $L(x)$ que nos da el número de enlaces salientes de un nodo x que nos manda su puntuación. Así, el cálculo de la L de una página y de una categoría quedan respectivamente de la siguiente forma:

$$L(p) = F_{ol} \times |outlinks| + F_{pc} \times |cats| + F_{ll} \times |langlinks| \quad (2.10)$$

$$L(c) = F_{sbc} \times |subcats| + F_{spc} \times |supercats| + F_{cp} \times |pages| + F_{ll} \times |langlinks| \quad (2.11)$$

2.2.6.4. Valores de los factores de modificación

Siguiendo la hipótesis de que una alteración de los factores que se aplican a cada tipo de relación mejora la relación entre puntuación y *termhood*, se buscó cuáles eran los valores óptimos de estos factores en un entorno de pruebas. Este entorno es el mismo que se ha usado para la evaluación global del algoritmo y que se explica a fondo en el siguiente capítulo, por lo que aquí se expondrán simplemente los resultados obtenidos.

Existen seis relaciones a las que podemos aplicar un factor diferente, que son las cuatro vistas en el apartado anterior pero desdoblando aquellas que se pueden interpretar en las dos direcciones:

1. supercategoría $\rightarrow$ subcategoría
2. subcategoría $\rightarrow$ supercategoría
3. página $\rightarrow$ categoría
4. categoría $\rightarrow$ página
5. página $\rightarrow$ página
6. enlace interlingüístico

Se han hecho pruebas con el dominio semántico de la Medicina y tres lenguas (castellano, catalán y árabe) comparando la calidad de la ordenación obtenida usando como *gold standard* una terminología médica conocida. No se incluyó el inglés porque debido al tamaño del conjunto de términos la evaluación de cada combinación de factores tomaba demasiado tiempo. Concretamente se calculó la precisión del primer 20 % de los términos ordenados por PageRank y se buscó la combinación de factores que mejoraba al máximo esa precisión.

El punto de partida es que todos los factores valen 1, lo que equivale a no aplicar ningún factor y calcular el PageRank normal. Entonces se buscaron los factores que más influían en la mejora de la precisión. Para ello se multiplicó cada factor por un valor grande $(10x)$ dejando el resto de factores a 1.

Mediante este procedimiento se observó que incrementar mucho algunos factores factores empeoraba la precisión de ese primer 20 %, y que incrementar otros la hacía aumentar en consideración. Finalmente se consideró que la precisión máxima se alcanzaba dándole un valor de 100 al factor de los enlaces interlingüísticos, de 100 al factor de las relaciones categoría $\rightarrow$ página, y dejando el resto de factores a 1.

Esto se corresponde con la intuición de que una página o categoría es muy probable que forme parte del dominio si también se ha encontrado su versión en otra lengua (y por tanto tiene enlace interlingüístico), y con la intuición de que es muy probable que una página sea del dominio si pertenece a una categoría que lo es.

3

Experimentos y evaluación

Una de los aspectos más maravillosos de la ciencia es que no importa dónde naciste, y no importa las creencias que tuvieran tus padres: si realizas el mismo experimento que otra persona ha realizado, en un momento y lugar distintos, obtendrás el mismo resultado

Neil deGrasse Tyson

En este capítulo se describe el entorno de pruebas en el que se han realizado las pruebas, se describen los experimentos y sus objetivos específicos y se presentan y comentan los resultados obtenidos.

3.1. Entorno de pruebas

En esta sección se describe en qué entorno se han realizado las pruebas para evaluar el algoritmo diseñado, en concreto se justifica sobre qué idiomas se ha trabajado y qué dominio semántico se han seleccionado para extraer sus terminologías.

3.1.1. Idiomas elegidos

Como base para realizar las pruebas se ha utilizado la Wikipedia en cuatro idiomas: inglés, castellano, catalán y árabe. Se ha escogido la versión en inglés por ser la más grande y compleja en número de páginas y categorías; la castellana y la catalana por ser las lenguas maternas del autor, lo que facilita la comprobación visual de la validez de los resultados; y la árabe para probar que el algoritmo puede funcionar en una lengua totalmente distinta a las demás.

Cuadro 3.1: Características de las Wikipedias

Idioma	Inglés	Castellano	Catalán	Árabe
Fecha *dump*	4/6/2013	21/5/2013	18/6/2013	7/6/2013
Tamaño en disco	48,4 GB	8,2 GB	4,5 GB	2,7 GB
Categorías	1 018 606	207 443	46 179	67 907
Páginas	4 318 182	1 928 573	408 804	405 462
Enlaces cat. → cat.	2 693 912	454 975	75 126	151 543
Enlaces pág. → pág.	260 080 808	28 304 075	29 303 381	15 021 564
Enlaces cat. → pág.	26 199 708	3 052 929	805 172	1 053 029
Enlaces interling.	15 920 564	10 154 376	6 239 768	6 239 768

En el cuadro 3.1 se muestran el tamaño de las diferentes Wikipedias que se han tratado en términos de número de categorías, páginas y relaciones. Dejando de lado la diferencia obvia en el número absoluto de páginas y categorías, se observan grandes diferencias en la relación entre número de enlaces y número de elementos. Así, por ejemplo, en la Wikipedia en catalán hay más enlaces de página a página que en la Wikipedia en castellano, a pesar de que la primera tiene una quinta parte de las páginas de la segunda. El análisis comparativo de estas diferencias queda fuera del ámbito de este proyecto.

3.1.2. Dominios semánticos elegidos

Las pruebas se realizarán sobre dos dominios semánticos: «Medicina» y «Finanzas». Se ha seleccionado el dominio médico porque cuenta con Snomed-CT, una colección de términos médicos conocida y validada que se puede usar como *gold standard* y que tiene versión en inglés y castellano. Al tener dos idiomas en los que evaluar sistemáticamente el funcionamiento del algoritmo podemos justificar que la aplicación es independiente del idioma.

El dominio semántico de las finanzas se ha seleccionado para mostrar visualmente con una lista de palabras que el algoritmo también funciona en un dominio que no tiene que ver con el dominio médico, y justificar así que la aplicación es independiente del dominio semántico que se esté tratando.

3.1.2.1. Snomed-CT

El acrónimo Snomed-CT a *Systematized Nomenclature of Medicine - Clinical Terms* y es la «terminología clínica integral, multilingüe y codificada de mayor amplitud, precisión e importancia desarrollada en el mundo» [8].

Es producto de la fusión de dos terminologías existentes, el Snomed RT (*Snomed Reference Terminology*) del *College of American Pathologists* y el *Clinical Terms Version* desarrollado por el *National Health Service* (NHS) del Reino Unido. Actualmente es mantenida por la *International Health Terminology Standards Development Organisation*, una organización en la que tienen participación gobiernos de todo el mundo, entre ellos el de España.

Snomed-CT tiene está disponible como recurso (tras registro y aceptación) en la página web de la Biblioteca Nacional de Medicina de EE.UU., dependiente del Departamento de Salud y Servicios Sociales de EE.UU. Tiene versión en inglés y en castellano, y se distribuye en sendos ficheros comprimidos que constan de multitud de ficheros, de entre los cuales hay uno de texto plano tabulado con todos los términos en una de sus columnas.

Para tratar la lista de términos contenida en Snomed-CT fue necesaria una pequeña conversión para adaptarla y poder compararla de manera cómoda con la lista de candidatos a términos que obtenemos con nuestra aplicación. En concreto, se realizaron tres transformaciones:

- Eliminar todas las palabras contenidas entre paréntesis mediante una expresión regular
- Pasar todas las palabras a minúsculas
- Sustituir los espacios en blanco por el carácter barra baja "_".
- Eliminar los elementos repetidos

En el cuadro 3.2 se muestra la fecha de la versión de Snomed-CT utilizada en cada idioma y el número de entradas de las listas de términos una vez realizada la transformación.

Cuadro 3.2: Características de Snomed-CT

Idioma	Fecha	Número de términos
Inglés	31/7/2013	570 669
Castellano	31/10/2013	505 337

3.2. Descripción de los experimentos

En esta sección se explica primero qué es lo que se quiere probar, y a continuación se describen los experimentos que se han realizado para conseguir ese objetivo.

3.2.1. Objetivos de los experimentos

Los experimentos que se han realizado pretenden probar principalmente tres cosas:

1. Que con el recorrido top-down del grafo de categorías y posterior obtención de las páginas expuesto en los apartados 2.2.3 y 2.2.4 sirve para obtener una terminología del dominio semántico.

2. Que existe una relación entre el PageRank calculado de la manera expuesta en el apartado 2.2.6 y el *termhood*, y que por tanto se puede usar dicho PageRank para obtener una terminología con mayor precisión pero menor cobertura.

3. Que el algoritmo expuesto a lo largo de la sección 2.2 es independiente del dominio semántico y los idiomas seleccionados, y por lo tanto se puede usar para obtener una terminología de cualquier dominio semántico en cualquier idioma.

3.2.2. Experimento 1: Relación PageRank - termhood

Este experimento busca probar que el algoritmo expuesto en este proyecto sirve para obtener una terminología para un dominio concreto (objetivo 1), y que la podemos ordenar por *termhood* usando una modificación de PageRank (objetivo 2). Para ello lanzaremos el algoritmo para las lenguas seleccionadas (inglés, castellano, catalán y árabe) y sobre el dominio semántico de la Medicina, y ordenaremos los términos usando diferentes variantes del PageRank, alterando los factores tal como se expone en el apartado 2.2.6.3. A continuación calcularemos la relación entre la ordenación y la precisión comprobando si los términos obtenidos existen o no dentro de la terminología Snomed-CT. Ordenaremos los términos siguiendo cuatro métodos distintos:

1. `all_zeros`: El caso sencillo. Todas las relaciones se ignoran, para todos los enlaces e el factor F_e será 0, por lo tanto es equivalente a no ordenar.

2. `no_langlinks`: Se consideran todas las relaciones excepto los enlaces interlingüísticos, cuyo factor F_{ll} será 0.

3. `all_ones`: El PageRank básico. Se consideran todas las relaciones, y se les

da el mismo peso, por tanto para todos los enlaces e el factor F_e será 1.

4. `best`: El PageRank modificado con la ponderación expuesta en el apartado 2.2.6.4, es decir, todos los factores valen 1 excepto el factor F_{cp} correspondiente al enlace categoría categoría $\rightarrow$ página y el factor F_{ll} correspondiente a los enlaces interlingüísticos, que valdrán ambos 100.

3.2.3. Experimento 2: Independencia de dominio e idioma

Este experimento busca probar que el algoritmo puede obtener una terminología para cualquier dominio semántico en cualquier idioma, es decir, que su funcionamiento es independiente de estas dos variables.

Para probar esta independencia usaremos el algoritmo en su variante `best` (punto 4 del apartado anterior) para obtener una terminología sobre el dominio semántico «finanzas». Dado que no contamos con una terminología validada para este dominio semántico, se mostrará la lista de los términos obtenidos en todos los idiomas a modo ilustrativo. Además, se calculará la correlación entre las distribuciones de puntuación de las listas ordenadas de términos para cada idioma para comprobar que tienen la misma forma.

3.3. Resultados

En esta sección se exponen y se describen los resultados de los experimentos descritos en el apartado anterior (3.2).

3.3.1. Resultados del experimento 1

Tras lanzar el algoritmo de obtención de categorías y páginas para el dominio semántico de la Medicina y los cuatro idiomas seleccionados, se obtuvieron los términos que se detallan en el cuadro 3.3.

Cuadro 3.3: Términos obtenidos para el dominio médico

Idioma	Categorías	Páginas	Total términos
Inglés	5 874	61 575	67 449
Castellano	687	8 186	8 872
Catalán	275	2 553	2 827
Árabe	548	6 771	7 318

A continuación se lanzó el cálculo del PageRank para todos los idiomas con cada uno de las ponderaciones de los factores que se detallan en la sección anterior (apartado 3.2.2). La comparación con Snomed-CT se ha hecho para los dos idiomas para los que existe dicha terminología, es decir, castellano e inglés. Para establecer una relación entre puntuación y precisión, se ha dividido la lista ordenada de términos en 10 *chunks*, cada uno con el 10 % de los términos, y se ha calculado la precisión si tomamos n de esos primeros chunks. Es decir, la precisión si tomamos el primer 10 % de los términos (primer *chunk*), si tomamos el primer 20 % (los dos primeros *chunks*), y así. Estos valores de precisión en función de la cobertura se detallan en los cuadros 3.4 y 3.5, y gráficamente en las figuras 3.1 y 3.2.

En el anexo A se muestra la lista de los 25 primeros términos obtenidos para cada idioma al ordenar mediante el método `best`, para mostrar a modo ilustrativo que los términos más relevantes aparecen realmente en las primeras posiciones.

En el gráfico de la figura 3.1 que representa la precisión en relación a la cobertura para cada método en el caso de Medicina - castellano vemos cómo el algoritmo de PageRank funciona para ordenar los términos por su *termhood*, ya que existe una relación positiva entre puntuación y precisión. Se observa que considerar los enlaces interlingüísticos mejora la precisión respecto a ignorarlos. También se observa cómo el *tuning* de los factores propuesto en el apartado 2.2.6.4 (que se usa en el método `best`) ha servido para mejorar notablemente la precisión independientemente de cuánto porcentaje de términos mejores tomemos.

Cuadro 3.4: Precisión para Medicina - castellano por cada 10 % de resultados

%/Método	all_zeros	no_langlinks	all_ones	best
10	0,5152	0,6913	0,7063	0,7819
20	0,5152	0,6396	0,6521	0,7207
30	0,5256	0,6254	0,6353	0,6782
40	0,5232	0,6078	0,6191	0,6447
50	0,5197	0,5782	0,5959	0,6107
60	0,5128	0,5577	0,5713	0,5976
70	0,5061	0,5390	0,5481	0,5767
80	0,5093	0,5198	0,5268	0,5493
90	0,5099	0,5005	0,5091	0,5226
100	0.4874	0,4874	0,4874	0,4874

En el gráfico de la figura 3.2 vemos la precisión en relación a la cobertura para el caso Medicina - inglés. En este caso observamos que el método llamado `best` sólo mejora la precisión para el 10 % mejor, pero no funciona mejor que el PageRank básico (método `all_ones`) una vez tomamos más porcentaje de términos. También se observa que la puntuación considerando los enlaces interlingüísticos no hace mejorar la precisión. Ambas cosas se explican porque en inglés hay muchos menos enlaces interlingüísticos que en castellano en proporción al número total de términos, al ser el idioma en el que se han obtenido más número de términos. Esto hace que los enlaces interlingüísticos tengan un peso menor en el cálculo de la puntuación.

Además, el cálculo de los factores óptimos calculados en el apartado 2.2.6.4 se hizo tomando solo el castellano, el catalán y el árabe, porque incluir el inglés suponía un coste computacional excesivo (cada cálculo de PageRank considerando los cuatro idiomas tarda varias horas), por lo que puede ser que ese óptimo encontrado no aplique al inglés.

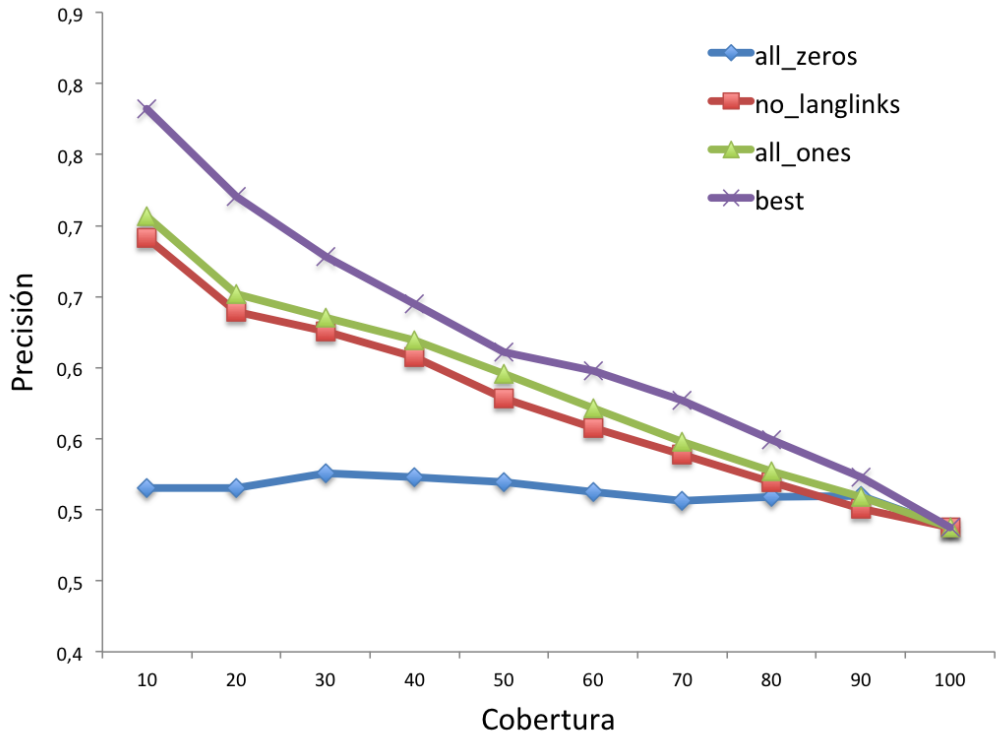


Figura 3.1: Relación entre PageRank y precisión para Medicina en castellano

3.3.1.1. Validación manual de resultados

La evaluación de la precisión validando contra Snomed-CT lo que da en realidad es una cota mínima de la precisión, ya que sólo consideramos correctos los términos que figuran en Snomed-CT. Es decir, el porcentaje de términos encontrados que pertenecen al dominio médico es como mínimo el que se muestra en las tablas 3.4 y 3.5.

Para acabar de comprobar la corrección del algoritmo y dado que Snomed-CT no es perfecto como referencia (puede haber términos que no estén, o que estén pero formulados de otra manera) se hizo también una validación manual de una parte de los términos en castellano. Concretamente se tomaron los primeros 1 000 términos de la lista ordenada que no aparecen en Snomed-CT, y se evaluó manualmente si pertenecían o no al dominio médico. La validación la hicieron dos

Cuadro 3.5: Precisión para Medicina - inglés por cada 10 % de resultados

%/Método	all_zeros	no_langlinks	all_ones	best
10	0,2489	0,5061	0,5062	0,5520
20	0,2426	0,4761	0,4774	0,4805
30	0,2441	0,4439	0,4440	0,4210
40	0,2448	0,3985	0,3991	0,3863
50	0,2446	0,3552	0,3557	0,3478
60	0,2443	0,3210	0,3214	0,3137
70	0,2437	0,2922	0,2927	0,2836
80	0,2416	0,2672	0,2677	0,2619
90	0,2400	0,2442	0,2443	0,2442
100	0,2238	0,2238	0,2238	0,2238

personas independientes y se pusieron en común los casos de discrepancia. La validación manual de estos 1 000 términos junto con la validación hecha contra Snomed-CT cubre poco más del 30 % de los términos. En el cuadro 3.6 y en el gráfico de la figura 3.3 se detallan los resultados. Como vemos, la precisión aumenta drásticamente y se sitúa cerca del 100 %.

Cuadro 3.6: Comparación de precisión con validación automática y manual

%	Validación con Snomed-CT	Validación manual
10	0,7819	0,9819
20	0,7207	0,9745
30	0,6782	0,9798

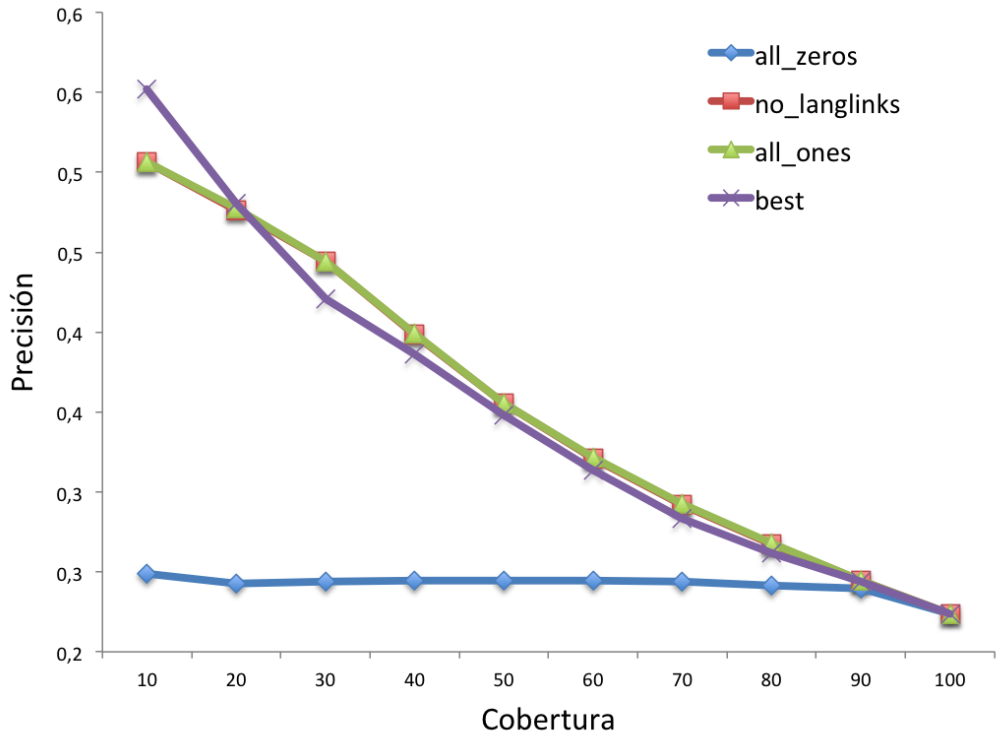


Figura 3.2: Relación entre PageRank y precisión para Medicina en inglés

3.3.2. Resultados del experimento 2

Como hemos explicado en el punto 3.2.3, además de evaluar el funciomiento del algoritmo para el dominio médico en castellano e inglés, también resulta interesante ver si el algoritmo se comporta de la misma forma en otros idiomas y dominio semántico.

La ejecución del algoritmo usando el método `best` sobre el dominio semántico de las finanzas y los cuatro idiomas seleccionados, se obtuvieron los términos que se detallan en el cuadro 3.7. Del mismo modo que para el dominio médico, en el anexo A se pueden ver los 25 primeros términos por puntuación para cada idioma.

Para comprobar que el algoritmo se comporta de la misma forma para el dominio médico y en el de las finanzas se ha hecho una comparación entre las

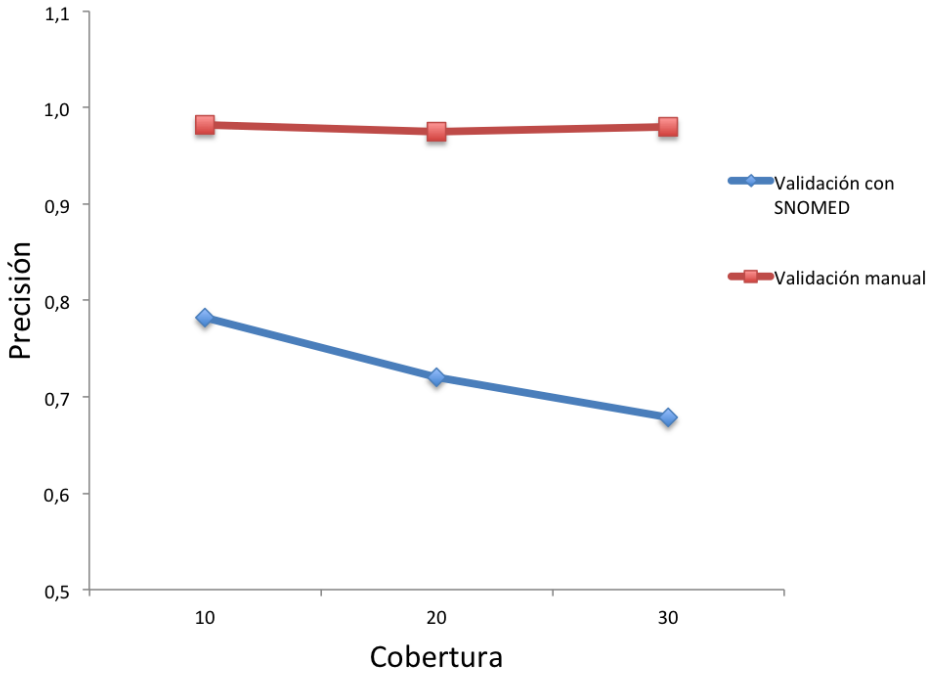


Figura 3.3: Comparación entre validación con Snomed-CT y manual

distribuciones de puntuación en cada idioma para los dos dominios. Los datos de puntuación media para cada bloque con el 10 % de términos se muestran en las tablas 3.8 y 3.9. Como podemos comprobar en la representación gráfica de estas distribuciones, que se muestran en las figuras 3.4 y 3.5, la distribución de la puntuación entre los distintos idiomas tiene la misma forma, y entre los dos dominios semánticos también se observa una forma muy similar.

En la tabla 3.10 se muestra el coeficiente de correlación de Pearson entre las distribuciones de puntuación en cada combinación de idiomas para el dominio médico. Vemos que la correlación en todos los casos es muy alto, confirmando lo que se observa en la figura 3.4. Para el caso de las finanzas los valores son muy similares y no se incluyen.

Con los datos expuestos en este apartado en la mano, podemos formular la hipótesis de que la extracción de términos y la ordenación por *termhood* que se ha probado válida para el dominio de medicina en castellano e inglés es extensible

Cuadro 3.7: Términos obtenidos para el dominio de las finanzas

Idioma	Categorías	Páginas	Total términos
Inglés	749	7 963	8 712
Castellano	107	1 204	1 311
Catalán	30	2 645	675
Árabe	121	1 437	1 558

a cualquier otro dominio semántico y cualquier idioma.

Cuadro 3.8: Puntuación media para medicina por cada 10 % de resultados

%/Idioma	Inglés	Castellano	Catalán	Árabe
10	8,4685	8,6248	9,9309	8,8442
20	4,6450	5,6365	6,6541	5,8282
30	3,5005	4,4734	5,6824	4,5826
40	2,7639	3,7234	4,9800	3,7348
50	2,2423	3,2023	4,3597	2,9965
60	1,7989	2,5718	3,7828	2,4778
70	1,4169	2,0756	3,1163	1,7756
80	1,1901	1,6430	2,4092	1,4173
90	0,8359	1,2954	1,5968	1,2488
100	0,1453	0,5331	0,5103	0,7234

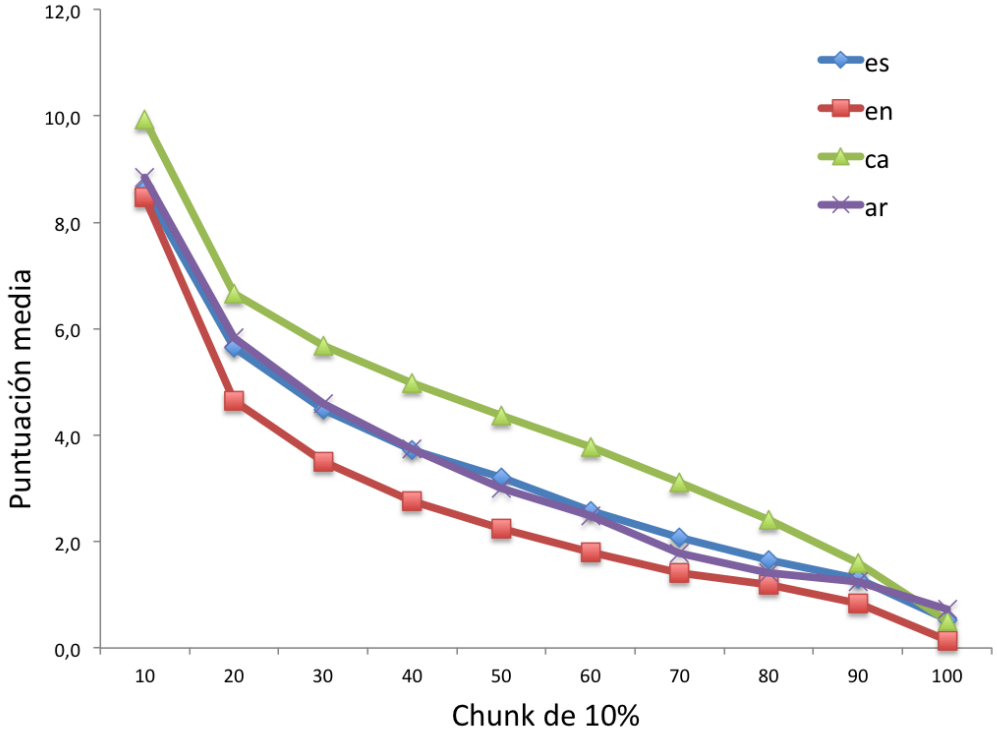


Figura 3.4: Distribución de la puntuación para Medicina

Cuadro 3.9: Puntuación media para finanzas por cada 10 % de resultados

%/Idioma	Inglés	Castellano	Catalán	Árabe
10	5,9407	6,6165	6,6414	7,3953
20	3,3832	4,6518	4,8939	5,2585
30	2,6729	3,9576	4,2592	3,9496
40	2,0626	3,1249	3,7578	2,9512
50	1,4363	2,7269	3,3365	2,6768
60	1,3457	2,3233	2,7995	1,7821
70	1,3051	1,5063	2,4985	1,4175
80	1,2213	1,3812	2,3529	1,3752
90	1,0551	1,2821	1,4933	1,3362
100	0,2871	0,7265	0,8562	0,7499

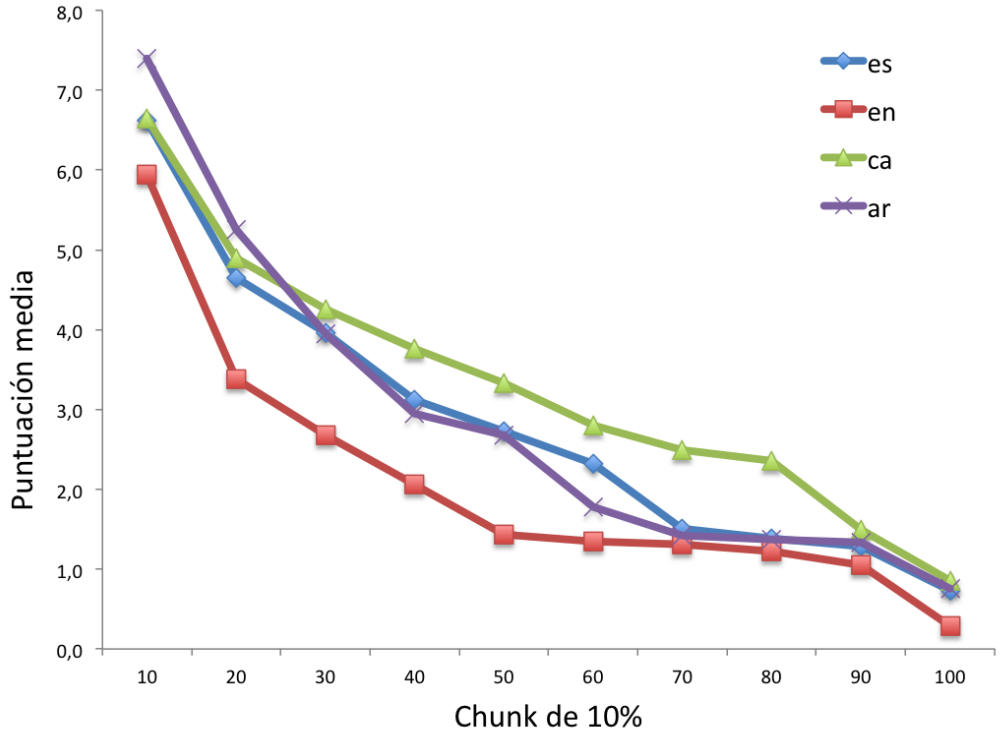


Figura 3.5: Distribución de la puntuación para Finanzas

Cuadro 3.10: Tabla de correlaciones entre las distribuciones de puntuación para el dominio médico

Idioma	Inglés	Castellano	Catalán	Árabe
Inglés	1,0	0,9961	0,9906	0,9927
Castellano	0,9961	1,0	0,9953	0,9947
Catalán	0,9906	0,9953	1,0	0,9820
Árabe	0,9927	0,9947	0,9820	1,0

4

Planificación

Un objetivo sin un plan no es más que un deseo

Antoine de Saint-Exupéry

En este capítulo se presentan las tareas en las que se dividió el proyecto en la planificación inicial, un diagrama de Gantt la planificación temporal de esas tareas, y otro que muestra cómo se han ejecutado realmente. Se acompaña el segundo diagrama con una justificación del desajuste en la previsión. Por último, se hace una valoración del coste económico de la realización del proyecto.

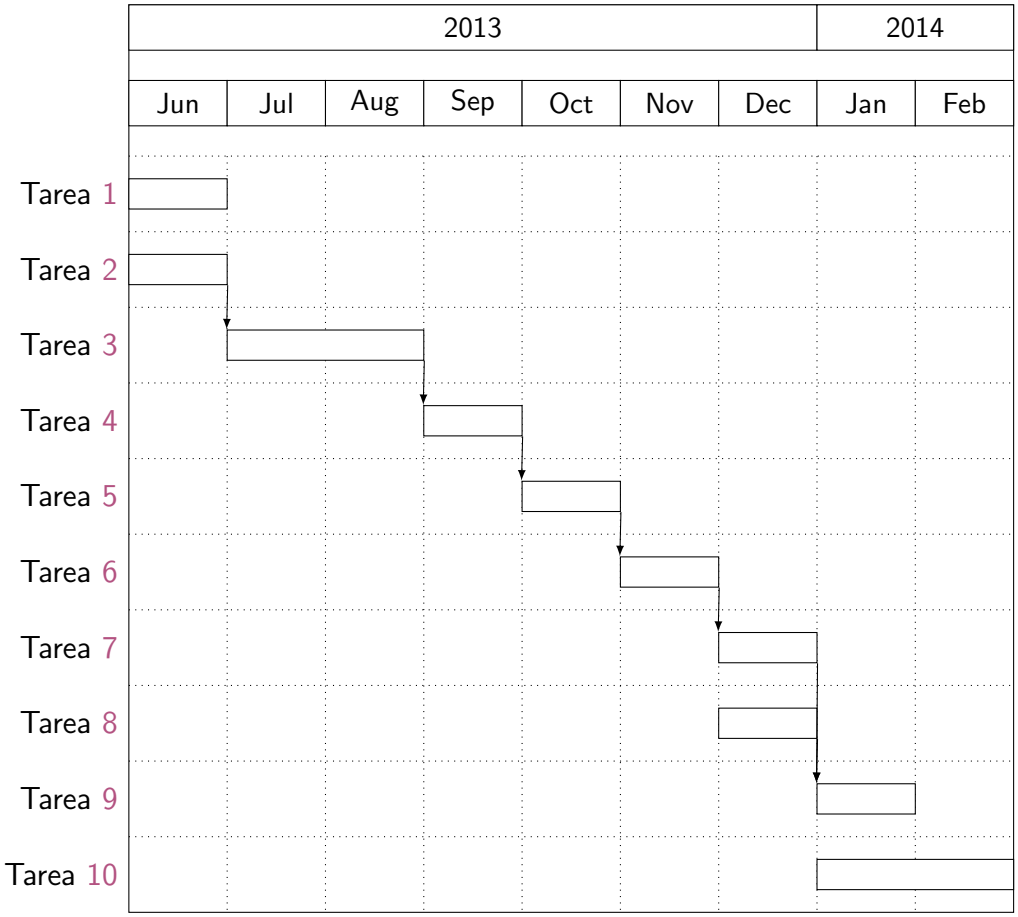
4.1. Tareas

1. Lectura de investigación y evaluación del state of the art. (20h)
2. Configurar un acceso cómodo a los datos de Wikipedia mediante una base de datos. Para esto se usará la aplicación JWPL complementada con scripts específicos para añadir las relaciones entre artículos del mismo concepto de Wikipedias en diferentes idiomas. (40h)
3. Desarrollo de clases y funciones de acceso cómodo a la BD, modelado de las clases y operaciones básicas del dominio. (80h)

4. Recorrido top-down por el árbol de categorías y obtención de las páginas de cada categoría para configurar el conjunto inicial de términos. (80h)
5. Implementación del algoritmo de PageRank básico para el conjunto de términos. (40h)
6. Adaptación del algoritmo de PageRank para el grafo de entidades y relaciones con el que contamos. (80h)
7. Desarrollo del entorno de test/evaluación del algoritmo (40h)
8. Obtención y procesado de los recursos que servirán de *gold standard* para hacer las pruebas (10h)
9. Evaluación del algoritmo usando los recursos obtenidos (40h)
10. Escritura de la memoria del proyecto. (100h)

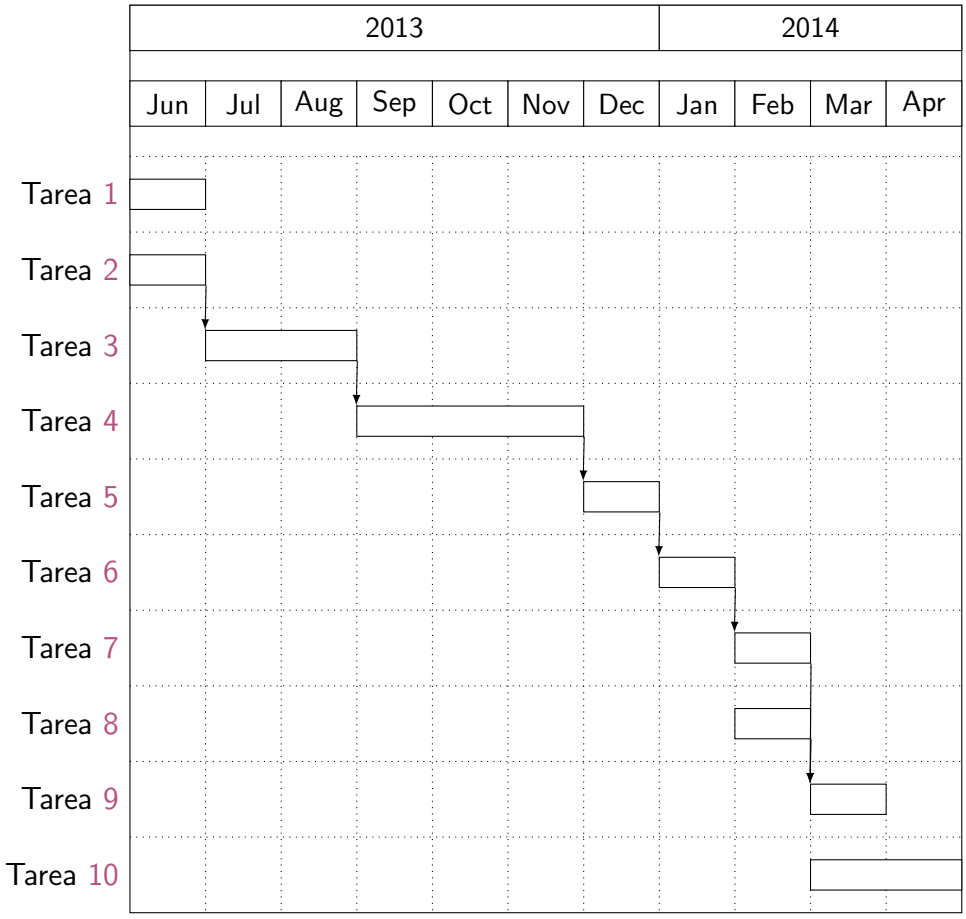
4.2. Planificación inicial

El proyecto se planificó con una duración de 8 meses, de junio de 2013 a febrero de 2014, con un tiempo total de dedicación de 530 horas.



4.3. Ejecución

Finalmente la ejecución del proyecto se extendió hasta finales del mes de abril de 2014, principalmente por imprevistos surgidos en la fase 4 de implementación del recorrido top-down por el grafo de categorías, que son los problemas documentados en el apartado 2.2.3 del diseño. Esa fase se alargó de las 80 horas previstas inicialmente a aproximadamente 200 horas. El tiempo real de ejecución del proyecto ha sido pues de unas 650 horas.



4.4. Valoración económica

En esta sección se realiza una valoración del coste económico que ha comportado este proyecto. Dado que se ha usado un ordenador que ya estaba a nuestra disposición y que se ha usado únicamente software libre para el desarrollo, el coste del proyecto depende exclusivamente del coste de las horas de trabajo invertidas.

Para el cálculo del coste por hora de trabajo, distinguiremos entre tres perfiles:

- Analista, que se encarga del estudio previo, el análisis del problema y el diseño de la solución que se implementará.
- Programador, que se encarga de implementar la solución siguiendo la especificación del analista.
- Evaluador, que realiza las pruebas de evaluación de la solución.

En la tabla 4.1 se muestran los costes en función de las horas invertidas de cada una de las tareas descritas en la sección 4.1, adaptadas a los perfiles definidos.

Cuadro 4.1: Tabla de coste económico

Tarea	Horas	Rol	€/hora	Coste (€)
Lectura y estudio previo	20	Analista	100	2 000
Diseño del acceso a datos	10	Analista	100	1 000
Descarga y configuración de la BD	30	Programador	50	1 500
Diseño de las clases de dominio	60	Analista	100	6 000
Implementación del dominio	340	Programador	50	17 000
Preparación entorno de pruebas	50	Evaluador	50	2 500
Evaluación del algoritmo	40	Evaluador	50	2 000
Documentación (memoria)	100	Analista	100	10 000
TOTAL	650			42 000

El coste total estimado del desarrollo del proyecto es de 42 000 €.

5

Conclusiones y trabajo futuro

No importa lo bonita que sea tu teoría, no importa lo inteligente que seas. Si no concuerda con la experimentación, es incorrecta.

Richard P. Feynman

En este capítulo se exponen las conclusiones que se extraen del desarrollo del proyecto y de la evaluación de los resultados, y a continuación se plantean las mejoras que se podrían realizar sobre el algoritmo y sobre la aplicación como trabajo futuro.

5.1. Conclusiones

Como conclusión principal podemos afirmar que los objetivos planteados en la sección 1.4 del capítulo de la introducción se han cumplido de manera satisfactoria. Una vez finalizado el proyecto disponemos de una aplicación que implementa un algoritmo que permite extraer una terminología en varios idiomas a partir de Wikipedia, y ordena la lista de términos en función de su *termhood*.

Un objetivo fundamental que se ha conseguido satisfacer es que la implementación tanto del algoritmo que extrae el conjunto de términos de Wikipedia como

el de ordenación fueran independientes tanto del dominio semántico como de los idiomas con los que se trabajara.

Además de la implementación del algoritmo, era tanto o más importante poder evaluarlo para comprobar su funcionamiento y poder medir su corrección respecto a otros sistemas existentes más allá de lo observable a simple vista. Esto se ha podido hacer satisfactoriamente gracias a que se ha podido validar la corrección de los términos calculando la precisión usando Snomed-CT como *gold standard*.

Como aportación al campo de la extracción de terminologías, cabe decir que tanto el hecho de considerar los enlaces interlingüísticos como el hecho de usar una modificación de PageRank para ordenar los términos suponen un enfoque novedoso respecto al trabajo que se había hecho anteriormente en este campo.

A partir de este proyecto se ha escrito un *paper* que ha sido presentado al XXX Congreso de la Sociedad Española para el Procesamiento del Lenguaje Natural (SEPLN) que se celebrará en septiembre de 2014 y está pendiente de aprobación. El texto del *paper* está adjuntado a esta memória en el apéndice D.

5.2. Trabajo futuro

Si bien el sistema funciona de manera razonable y la aplicación ejecutable ofrece la funcionalidad básica, existen algunos puntos en los que se puede mejorar lo presentado en este proyecto.

5.2.1. Mejoras en el algoritmo

- Como hemos visto en el análisis de resultados, la modificación del PageRank no funciona de forma óptima para el inglés, ya que el *tuning* de los factores se ha realizado sin tener en cuenta este idioma. Una vía de mejora del algoritmo sería mejorar dicho *tuning* para obtener unos valores que den mejores resultados en inglés. Además, sería interesante lanzar el algoritmo

sobre un dominio semántico distinto que también cuente con una terminología validada, para así ver si el *tuning* es independiente o no del dominio y el idioma.

- El algoritmo exige que se introduzca el nombre de una categoría que se corresponda con un dominio semántico, para usarla como categoría *top*. Una mejora interesante sería permitir partir de un conjunto de categorías *top*, o que a partir de un conjunto de términos dados el algoritmo encuentre automáticamente esas categorías *top*.

5.2.2. Mejoras en la aplicación

- De cara a mejorar la aplicación desde el punto de vista del usuario, sería útil contar con una interfaz más amigable que la línea de comandos, ya sea mediante una aplicación de escritorio usando toolkits gráficos o mediante una aplicación web.
- El proceso de descarga y conversión de las Wikipedias para pasarlas a base de datos podría estar automatizado con un script.
- El hecho de tratar varios idiomas permite pensar en múltiples formas de presentar los resultados. La aplicación en su estado actual solo permite listar las palabras más relevantes para cada idioma. Sería muy útil por ejemplo mostrar los términos en todos los idiomas alineando aquellos que tienen enlaces interlingüísticos entre ellos, de forma que se pueda usar la tabla resultante como diccionario multilingüe del dominio semántico escogido.

Bibliografía

[1] Krauthammer, M. I., Nenadic, G., Term identification in the biomedical literature, *Journal of Biomedical Informatics*, pág. 512-26, diciembre 2004.

[2] Cabré, Mª T., Estopà R., Vivaldi J., Automatic term detection: A review of current systems, *Recent Advances in Computational Terminology*, 2001.

[3] European Parliament TermCoord, *Testing of term extraction tools* http://termcoord.wordpress.com/about/testing-of-term-extraction-tools/, 23/2/2014.

[4] Kageura, K., Umino, B. *Methods of automatic term recognition: A review. Terminology*, pág 259–289, 1996.

[5] US Environmental Protection Agency. *What is Terminology?*, http://ofmpub.epa.gov/sor_internet/registry/termreg/home/whatisterminology/, 22/2/2014.

[6] Zesch, T., Müller, C., Gurevych, I., "Extracting Lexical Semantic Knowledge from Wikipedia and Wiktionary", *Proceedings of the 6th International Conference on Language Resources and Evaluation*, mayo 2008.

[7] De Melo, G., Weikum, G., Untangling the Cross-Lingual Link Structure of Wikipedia, *48th Annual Meeting of the Association for Computational Linguistics*, Uppsala, Sweden, 2010.

[8] Ministerio de Sanidad, Servicios Sociales e Igualdad. *¿Qué es Snomed-CT?*, http://www.msps.es/profesionales/hcdsns/areaRecursosSem/snomed-ct/quees.htm, 11/3/2014.

Glosario

terminología es un conjunto de términos propios de un dominio semántico concreto.

dominio semántico es un orden determinado de ideas, materias o conocimientos (p.ej. medicina, filosofía, matemáticas).

Procesamiento del Lenguaje Natural es un campo de la Informática, la Inteligencia Artificial y la Lingüística que estudia las interacciones entre computadores y el lenguaje natural (humano).

MWE , del inglés *multiword expression*, es una expresión que consta de varias palabras.

Named Entity es un elemento del texto que representa categorías predefinidas como nombres de personas, organizaciones, localizaciones, expresiones de horas, cantidades, valores monetarios, porcentajes, etc.

termhood es el grado en el que una unidad lingüística está relacionada con conceptos específicos del dominio.

enlace interlingüístico es un enlace entre una página o categoría de Wikipedia y su versión en otro idioma.

PageRank es un algoritmo desarrollado por Google para asignar de forma numérica la relevancia de los documentos (o páginas web) indexados por un motor de búsqueda.

gold standard es un conjunto de resultados que se acepta como referencia para

la realización de tests.

Wikipedia es una enciclopedia libre, políglota y editada colaborativamente. Está administrada por la Fundación Wikimedia, una organización sin ánimo de lucro.

charset es la abreviatura de *character set* (codificación de caracteres), que es la manera en que un símbolo en lenguaje natural se almacena en bytes. Los más habituales son ASCII y la familia Unicode.

Snomed-CT es la terminología sobre el dominio médico integral, multilingüe y codificada de mayor amplitud, precisión e importancia desarrollada en el mundo. Su nombre es el acrónimo de *Systematized Nomenclature of Medicine - Clinical Terms*.

Apéndices

Apéndice A

Listas de términos

Figura A.1: Términos del dominio médico en inglés

#	Inglés
1	ICD-10
2	EMedicine
3	Medicine
4	MedlinePlus
5	Pregnancy_category
6	Spinal_cord
7	Oxygen
8	Regulation_of_therapeutic_goods
9	Cerebellum
10	Hydrogen
11	Meninges
12	Carbon
13	Central_nervous_system
14	Clinical_trial
15	Lateral_ventricles
16	Nitrogen
17	Cerebral_cortex
18	Chest_trauma
19	Diencephalon
20	Pathology
21	Circulatory_system
22	Blood
23	Fourth_ventricle
24	Pons
25	Cancer

Figura A.2: Términos del dominio médico en castellano

#	Castellano
1	MedlinePlus
2	CIE-10
3	CIE-9
4	Medicina
5	Categorías_farmacológicas_en_el_embarazo
6	Ventrículos_laterales
7	Riñón
8	Sangre
9	Inflamación
10	Enfermedad
11	Hígado
12	Médula_espinal
13	Bacteria
14	Corazón
15	Cerebelo
16	Reflejo
17	Bilis
18	Arteria
19	Conducto_colédoco
20	Diafragma_(anatomía)
21	Intestino
22	Duodeno
23	Espina_bífida
24	Tórax
25	Ensayo_clínico

Figura A.3: Términos del dominio médico en catalán

#	Catalán
1	MedlinePlus
2	CIM-10
3	Medicina
4	EMedicine
5	Categoria_de_risc_d'un_fàrmac_en_l'embaràs
6	Ventricles_laterals
7	CIM-9
8	Fetge
9	Sang
10	Medul·la_espinal
11	Pàncrees
12	Classificació_internacional_de_malalties
13	Cerebel
14	Cor
15	Infecció
16	Sistema_nerviós_central
17	Reflex
18	Tòrax
19	Meninge
20	Assaig_clínic
21	Coll_(anatomia)
22	Ronyó
23	Patologia
24	Malaltia
25	Digestió

Figura A.4: Términos del dominio médico en árabe

#	Árabe
1	طب
2	بلس_مدايين
3	ميديسين_إي
4	الحمل_أثناء_السلامة_في
5	جان_بي_بطين
6	Drugs.com*
7	للأمراض_الدولي_التصنيف
8	سريرية_تجربة
9	طب_بذرة
10	القانوني_الوضع
11	مرض
12	الباء_تاب
13	دم
14	سرطان
15	قلب
16	الأمراض_علم
17	منع_كس
18	عدوى
19	سرعة
20	فقري_عمود
21	تخثر
22	هذا؟_سمى_من
23	صدري_قفص
24	(عضو)_كلية
25	بطن

*Aparece así en la Wikipedia árabe

Figura A.5: Términos del dominio de las finanzas en inglés

#	Inglés
1	Finance
2	Currency
3	Value_investing
4	Bank
5	Private_equity
6	Magic_formula_investing
7	Economics_and_finance_stubs
8	Subordinated_debt
9	Technical_analysis
10	Cash_flow
11	Modern_portfolio_theory
12	Insurance
13	Collateralized_debt_obligation
14	Debt
15	Balance_sheet
16	Margin_(finance)
17	Stock
18	High-yield_debt
19	Securitization
20	Fundamental_analysis
21	Central_bank
22	Warrant_(finance)
23	Tax
24	Bond_(finance)
25	Credit_union

Figura A.6: Términos del dominio de las finanzas en castellano

#	Castellano
1	Finanzas
2	Divisa
3	Activo_(contabilidad)
4	Banco
5	Banca_de_reserva_fraccional
6	Crédito_subprime
7	Tasa_de_interés
8	Dividendo_(economía)
9	Banco_central
10	Cheque
11	Prima_de_riesgo
12	Inversión
13	Hipótesis_de_eficiencia_de_los_mercados
14	Bono_(finanzas)
15	Activo_tóxico
16	Moneda
17	Tarjeta_de_crédito
18	Dinero
19	Instrumento_financiero
20	Valor_(finanzas)
21	Estado_de_situación_patrimonial
22	Titulización
23	Flujo_de_caja
24	Valor_actual_neto
25	Activo_financiero

Figura A.7: Términos del dominio de las finanzas en catalán

#	Catalán
1	Finances
2	Moneda
3	Estats_financers
4	Inversió_econòmica
5	Transferència_bancària
6	Actiu_(comptabilitat)
7	Banc_(empresa)
8	Flux_de_caixa
9	Balanç_de_situació
10	Actiu_financer
11	Dividend_(economia)
12	Fallida
13	Derivat_financer
14	Facturatge
15	Valoració_(finances)
16	Obligació_(finances)
17	Moneda_de_reserva
18	Ingrés
19	Instrument_financer
20	Targeta_de_crèdit
21	Àngel_inversor
22	Deute_subordinat
23	Futur_(finances)
24	Fusions_i_adquisicions
25	Valor_mobiliari

Figura A.8: Términos del dominio de las finanzas en árabe

#	Árabe
1	عملة
2	تمويل
3	خاص_مهم
4	مصرف
5	المساهم_حق
6	الحالية_القيمة_صافي
7	اقتصاد
8	متداولة_عملة
9	اشتقاقي_عقد
10	وقائية_محفظة
11	القيمة_قياس
12	استثمار
13	السوق_كفاءة_فرضية
14	مكشوف_بيع
15	المدين
16	(مراسبة)_شهرة
17	آجل_عقد
18	قانونية_عملة
19	اقتصاد_بذرة
20	مالية_خدمات
21	مركزي_مصرف
22	للنقود_وقتية_قيمة
23	مالية_قياضة
24	مركبة_فائدة
25	متجمعة_فائدة

Apéndice B

Guía de instalación

B.1. Requisitos hardware

Como hardware basta con una máquina cualquiera de arquitectura x86 con los siguientes requisitos recomendados:

- 100 GB libres de disco duro. El espacio de disco es necesario para almacenar las Wikipedias como bases de datos. Como referencia, las versiones usadas en este proyecto ocupan:
  - Inglés (en): ∼48,4 GB
  - Castellano (es): ∼8,2 GB
  - Catalán (ca): ∼4,5 GB
  - Árabe (es): ∼2,7 GB
- Mínimo de 4GB de RAM. Hay que tener en cuenta que un conjunto de términos como el del dominio de la Medicina en inglés ocupan 1 GB en memoria.

B.2. Requisitos software

- Sistema operativo: Windows, Linux, Mac OS X, y de hecho cualquiera en el que puedan instalarse el resto de herramientas.
- Máquina virtual de Java: versión 1.6
- MySQL 5.5
- Python 2.7.3 con los siguientes módulos extra:
  - MySQL-python v1.2.3
  - numpy v1.8.0
- Aplicación JWPLDataMachine.jar disponible en este enlace (visible en PDF)

B.3. Descarga de Wikipedia mediante JWPL

1. Descargar los dumps de los idiomas que nos interesen en la página web Wikimedia dumps. Para cada idioma necesitaremos los siguientes ficheros:
   - `pages-articles.xml.bz2`: todos los artículos y plantillas
   - `pagelinks.sql.gz`: enlaces página $\rightarrow$ página
   - `categorylinks.sql.gz`: enlaces categoría $\longleftrightarrow$ página
   - `langlinks.sql.gz`: enlaces interlingüísticos
2. Crear la base de datos para cada idioma:

   ```
   # mysqladmin -u[USER] -p create [DB_NAME] DEFAULT CHARACTER SET utf8
   ```

   El nombre de cada base de datos será el código de idioma en Wikipedia seguido de la palabra «wiki». Así, por ejemplo, para inglés la BD se debe llamar «enwiki» y para el castellano «eswiki».

3. Para cada idioma, crear las tablas necesarias con el fichero `jwpl_tables.sql` proporcionado por JWPL:

   ```
   # mysql [DB_NAME] <jwpl_tables.sql
   ```

4. Lanzar la aplicación de conversión DataMachine del proyecto JWPL con el siguiente comando:

   ```
   # java -jar JWPLDataMachine.jar [LANGUAGE] [MAIN_CATEGORY_NAME]
   [DISAMBIGUATION_CATEGORY_NAME] [SOURCE_DIRECTORY]
   ```

   Algunos parámetros son triviales pero otros necesitan algo de intervención humana:

   - `LANGUAGE`: nombre del idioma en inglés y en minúsculas tal como figura en la lista de idiomas soportados por JWPL
   - `MAIN_CATEGORY_NAME`: nombre de la categoría raíz que contiene todas las demás categorías (p.e. en inglés es «Contents»)
   - `DISAMBIGUATION_CATEGORY_NAME`: nombre de la categoría raíz de las categorías de desambiguación (en inglés «Disambiguation_pages»)
   - `SOURCE_DIRECTORY`: directorio donde tenemos los ficheros del dump detallados anteriormente

5. Importar los ficheros generados por DataMachine a cada BD:

   ```
   # mysqlimport -uUSER -p --local --default-character-set=utf8
   [DB_NAME] `pwd`/*.txt
   ```

6. Descomprimir el fichero `langlinks.sql.gz` y añadirlo a cada BD:

   ```
   # mysql [DB_NAME] <langlinks.sql
   ```

7. Añadir la columna `langlinks.sql.gz` a la tabla `category` de cada BD, esta vez desde la línea de comandos de MySQL:

   ```
   mysql> alter table category add depth int default -1;
   ```

Apéndice C

Manual de usuario

C.1. Configuración previa

Antes de arrancar la aplicación hay que introducir algunos parámetros en el fichero de configuración `settings.py`:

- `langs`: lista de códigos Wikipedia de las lenguas que se quieren tratar, con el formato de ejemplo

  `langs = ['en','es']`

- `articles`: lista de categoría raíz de los artículos de Wikipedia en las lenguas en las que existe. Si no existe, categoría raíz de Wikipedia. Formato de ejemplo:

  `articlecats = {'en' : 'Articles', 'es' : 'Artículos'}`

- `output_folder`: Directorio donde se colocarán los ficheros de guardado de los *catsets* y *pagesets*. Deberá existir dentro del directorio de la aplicación.

- `host`, `user` y `passwd`: parámetros de la conexión a base de datos. Dentro de ese usuario se espera que haya una BD creada siguiendo las instrucciones del apartado B.3 para cada lengua de las especificadas en la variable `langs`.

C.2. Descripción del menú

La aplicación se ejecuta mediante línea de comandos lanzando el fichero `termEx.py`:

```
# python termEx.py
```

Para cada acción que puede realizar el usuario se muestra el siguiente menú:

```
Current language: en
Operations
(0) Exit
(1) Change language
(2) Generate initial sets
(3) Clean global
(4) Save catsets to file
(5) Load catsets from file
(6) Set absolute depth
(7) Print catset and pageset sizes
(8) Compute page rank
(9) Print top pages
(10) Write terms to file
```

(0) `Exit`: Salir de la aplicación

(1) `Change language`: Muestra la lista de los idiomas con los que está configurado la aplicación para poder seleccionar el idioma en el que se quiere introducir el nombre del dominio semántico. Por defecto el idioma seleccionado es el primero de la lista del fichero de configuración.

(2) `Generate initial sets`: Solicita que el usuario introduzca el nombre de un dominio semántico en el idioma seleccionado. A continuación carga en memoria el conjunto de categorías y páginas para todos los idiomas y busca los enlaces interlingüísticos entre ellas. Informa de los pasos que va dando y del progreso en tiempo real. Esta operación puede tardar minutos

o incluso horas para dominios muy generales.

(3) `Clean global`: Vacía los conjuntos de categorías y páginas de la memoria.

(4) `Save sets to file`: Vuelca todos los conjuntos almacenados en memoria a ficheros que coloca en un directorio dentro del directorio especificado en la variable de configuración `output_folders`.

(5) `Load sets from file`: Carga en memoria los conjuntos almacenados en un directorio especificado.

(6) `Set absolute depth`: Asigna el valor de profundidad absoluta (distancia a la raíz de los artículos) a todas las categorías de la base de datos. Esta operación solo es necesario realizarla una vez para cada base de datos.

(7) `Print catset and pageset sizes`: Imprime por pantalla información de los tamaños de los conjuntos de páginas y categorías que tiene en memoria.

(8) `Compute page rank`: Calcula el PageRank para todas las categorías y páginas cargadas en memoria. Esta operación puede tardar minutos o incluso horas para dominios muy generales.

(9) `Print top pages`: Imprime por pantalla la lista de los 50 términos con más puntuación de PageRank para cada idioma.

(10) `Write terms to file`: Vuelca en un fichero la lista completa de términos para cada idioma.

Apéndice D

Paper enviado al XXX Congreso de la SEPLN

Boosting Terminology Extraction through Crosslingual Resources

Mejora de la extracción de terminología usando recursos translingües

Author 1
Affiliation 1
Address 1
Mail 1

Author 2
Affiliation 2
Address 2
Mail 2

Resumen: La extracción de terminología es una tarea de procesamiento de la lengua sumamente importante y aplicable en numerosas áreas. La tarea se ha abordado desde múltiples perspectivas y utilizando técnicas diversas. También se han propuesto sistemas independientes de la lengua y del dominio. La contribución de este artículo se centra en las mejoras que los sistemas de extracción de terminología pueden lograr utilizando recursos translingües, y concretamente la Wikipedia y en el uso de una variante de PageRank para valorar los candidatos a término.

Palabras clave: Extracción de terminología. Procesamiento translingûe de la lengua. Wikipedia, PageRank

Abstract: Terminology Extraction is an important Natural Language Processing task with multiple applications in many areas. The task has been approached from different points of view using different techniques. Language and domain independent systems have been proposed as well. Our contribution in this paper focuses on the improvements on Terminology Extraction using crosslingual resources and specifically the Wikipedia and on the use of a variant of PageRank for scoring the candidate terms.

Keywords: Terminology Extraction, Wikipedia, crosslingual NLP, PageRank

1 Introduction

Terminology Extraction is an important Natural Language Processing, *NLP*, task with multiple applications in many areas. Domain terms are a useful mean for tuning both resources and *NLP* processors to domain specific tasks. The task is important and useful but it is also challenging. In (Krauthammer, Nenadic, 2004), it has been said that "terms identification has been recognized as the current bottleneck in text mining and therefore an important research topic in *NLP*".

Terms are usually defined as lexical units that designate concepts in a restricted domain. Term extraction (or detection) is difficult because there is no formal difference between a term and a non terminological unit of the language. Furthermore, the frontier between terminological and general units is not always clear and the belonging to a domain is more a fuzzy than a rigid function. (Hartmann, Szarvas and Gurevych, 2012) present the lexical units in a two dimensional space where *x* axe refers to *domainhood*, represented as a continuous, and *y* axe to *constituency* of the linguistic unit, i.e. single words and multiwords expresions, *MWE*, (2-grams, 3-grams, etc.). Several types of *MWE* can be considered such as idioms, "kick the bucket", particle verbs, "fall off", collocations, "shake hands", Named Entities, "Los Angeles", compound nouns, "car park", some of which are compositional and other not. Obviously not

all the *MWE* are terminological and not all the terms are *MWE*[1].

In this paper we prefer to refer to terms as term candidates (*TC*). As pointed out above, *TC* can be atomic lexical units or *MWE* composed by atomic units (usually named basic components of the term). There are some properties that must hold for a given *TC* in order to be considered a term: i) *unithood*, ii) *termhood* and iii) specialized usage. *Unithood* refers to the internal coherence of a unit: Only some sequences of POS tags can produce a valid term, N (e.g. "Hepatology" in the Medical domain), NN (e.g. "Blood test"), JN (e.g. "Nicotinic antagonist"), etc. and these combinations are highly language dependent), *termhood* to the degree a *TC* is related to a domain-specific concept and specialized usage (general language versus specialized domain). It is clear that measuring such properties is not an easy task. They can only be measured indirectly by means of other properties easier to define and measure like frequency (of the *TC* itself, its basic components or in relation to general domain corpus), association measures, syntactic context exploration, highlighting and/or structural properties, position in an ontology, etc.

We present in this paper a term ranker aimed to extract a list of *TC* sorted by *termhood*. Our claim is that the system is language and domain independent. In fact nothing in our approach depends on the language or the domain. The experiments and evaluation are carried out in two domains, *medicine* and *finance* and four languages: English, Spanish, Catalan, and Arabic.

Our approach is based on extracting for each domain the *TC* corresponding to all the languages simultaneously, in a way that the terms extracted for a language can reinforce the corresponding to the other languages. As unique knowledge sources we use the wikipedias of the involved languages.

Following this introduction, the paper is organized as follows. In section 2 we describe some recent work done in this area. Section 3 describes the methodology that we use to obtain new terms while section 4 describes the experiments carried out as well as its evaluation. Finally, in section 5, we present some conclusions and directions for future work.

2 Related work

Term extraction, *TE*, and related tasks (Term ranking, Named Entity Recognition, *MWE* extraction, lexicon and ontology building, multilingual lexical extraction, etc.) have been approached typically using linguistic knowledge, as in (Heidet al, 1996), or statistical strategies, such as ANA (Enguehard, Pantera, 1994), with results not fully satisfactory, see (Cabré, Estopà, Vivaldi, 2001) and (Pazienza. Pennacchiotti, Zanzotto, 2005). Also, *TE* systems often favor recall over precision resulting in a large number of *TC* that have to be manually checked and cleaned.

Some approaches combine both linguistic knowledge and Statistics, such as TermoStat (Drouin, 2003), or (Frantzi, Ananiadou and Tsujii, 2009), obtaining clear improvement. A common limitation of most extractors is that they do not use semantic knowledge, therefore their accuracy is limited. Notable exceptions are Metamap (Aronson, Lang, 2010) and YATE (Vivaldi, 2001).

Wikipedia[2], *WP*, is by far the largest encyclopaedia in existence with more than 32 million articles contributed by thousands of volunteers. *WP* experiments an explosive growing. There are versions of *WP* in more than 300 languages although the coverage (number of articles and average size of each article) is very irregular. For the languages covered by the experiments reported here the size of the corresponding *WPs* are 4,481,977 pages in English, 1,091,299 in Spanish, 425,012 in Catalan, and 269,331 in Arabic. A lot of work has been performed for using this resource in a variety of ways. See (Medelyan et al, 2009) and (Gabrilovich, Markovitch, 2009) for excellent surveys.

WP has been, from the very beginning, an excellent source of terminological information.

[1] Many authors claim that most terms are *MWE*- From our experience we think that almost half of the *TC* extracted are single words.

[2] https://www.wikipedia.org/

(Hartmann, Szarvas and Gurevych, 2012) present a good survey of main approaches, see also (Sabbah, Abuzir, 2005). Both the structure of *WP* articles (infoboxes, categories, redirect pages, input, output, and interlingual links, disambiguation pages, etc.) and their content have been used for *TE*. Figure 1 presents the bi-graph structure of *WP*. This bi-graph structure is far to be safe. Not always the category links denote belonging of the article to the category; the link can be used to many other purposes. The same problem occurs in the case of links between categories, not always these links denote hyperonymy/hyponymy and so the structure shown in the left of figure 1 is not a real taxonomy. Even worse is the case of inter-page links where the semantics of the link is absolutely unknown.

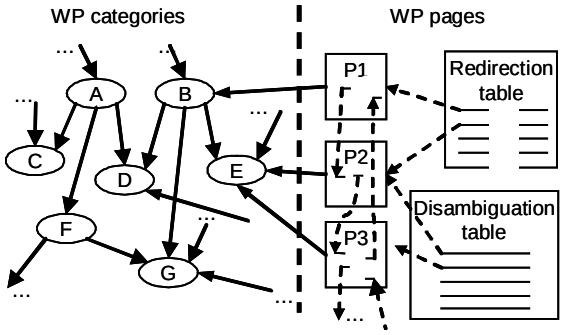


Figure 1: The graph structure of Wikipedia

Nakayama and colleagues, (Erdmann et al, 2008), and (Erdmann et al, 2009) face the problem of bilingual terminology extraction mainly using the interlingual links of *WP*, while (Sadat, 2011) uses, as well, the context of words and the Wikitionary[3]. Gurevych and collegues, (Wolf, Gurevych, 2010), (Niemann, Gurevych, 2011, map *WP* and WordNet[4], *WN*. Vivaldi and Rodríguez propose in (Vivaldi, Rodríguez, 2011) to use *WP* for extracting and evaluating term candidates in the medical domain, and in (Vivaldi, Rodríguez, 2012) propose to obtain lists of terms a multilingual/multidomain setting. (Alkhalifa, Rodríguez, 2010) use *WP* for enriching the Arabic WordNet with *NE*.

[3] http://www.wiktionary.org/
[4] http://wordnet.princeton.edu/

3 Our approach

The global architecture of our approach is displayed in Figure 2. As we can see it consists of 6 steps that are applied for each of the domains as detailed below. Let *d* be the domain considered (as we will see in Section 4 our experiments and evaluation have been carried out for medicine and finance). We will note WP^l the wikipedia for language *l* (*l* ranging on the four languages considered, i.e. en, sp, ca, ar).

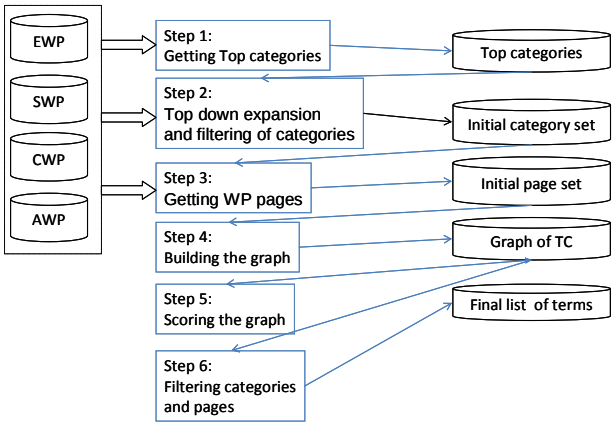


Figure 2: Architecture of our approach

As a preparatory step we downloaded the required *WP*s from the *WP* dumps site[5] and then we used the *JWLP*[6] toolbox, (Zesch, Müller, Gurevych, 2008) for obtaining a *MYSQL* representation of the *WP*s and the interlingual links. We then looked for the Top of the *WP* category graph (*topCat*), that for English *WP* corresponds to "Articles"[7]. Further on we enriched the *WP* category graph with the depth respect to *topCat* of all the categories. We have also downloaded the tables corresponding to the interlingual links. Although these links present problems of lack of reciprocity and inconsistency, see (De Melo, Weikum, 2010) for a method of facing these problems, we have made no attempt to face them and we hace accepted all the links as correct.

In step 1 the top category of domain *d* is looked for in WP^{en}. Let $topCaDom^d$ be this

[5] http://dumps.wikimedia.org/
[6] http://www.ukp.tu-darmstadt.de/software/jwpl/
[7] In fact the real top category is "Contents", we have used "Articles" instead as *topCat* for avoiding that the shortest paths to the top traverse meta-categories.

category. Once located $topCaDom^d$ for English, the top categories for the other languages are obtained through the corresponding interlingual links.

In step 2, the initial set of categories is obtained for each language *l* by navigating top down, from the top category, through category/category links, the category graph of WP^l. Although ideally the *WP* category graph is a *DAG*, it is not really the case because two problems: i) the existence of cycles and ii) the presence of backward links.

Both problems have the same origin: the way of building the resource by lots of volunteers working independently. Many cycles occur in *WK*, an example, from the Spanish *WP*, is *Drogas* → *Drogas y Derecho* → *Narcotráfico* → *Drogas.* Detecting cycles and removing them is quite straightforward.

The second problem is more serious and difficult to face. When working with English *WP* we discovered that for the domain *Medicine* 90% of the whole *WP* category graph was collected as descendants of the domain top category. Consider the following example, from English *WP: Volcanology* → *Volcanoes* → *Volcanic islands* → *Iceland*. In this case going Top Down from the category *Volcanology* a lot of categories related to *Iceland*, but with no relation with *Volcanology* will be collected. For facing the second problem (backward links) we can take profit of the following information:

- The relative depth of each category *c* regarding $topCaDom^d$, i.e, the length of the shortest path from *c* to $topCaDom^d$.
- The absolute depth of c, computed in the preparatory step, i.e, the length of the shortest path from *c* to *topCat*.
- The absolute depths of *topCat* and $topCaDom^d$.
- The absolute depth of the parent of *c* in the dop down navigation.

We have experimented with several filtering mechanisms, from the very simplest one, pruning the current branch when the depth of *c* is lower than the depth of the parent of *c*, to others more sophisticated. Finally we decided to apply the following filtering: *c* is pruned, and not further expanded, if the relative depth of *c* is greater than the difference between the absolute depths of *topCat* and $topCaDom^d$ plus 1.

In step 3 we build the initial set of pages, collecting for each category in the set of initial categories the corresponding pages through the category/page links. The process is, so, quite straightforward. A simple filtering mechanism is performed for removing Named Entities and not content pages.

In step 4, from the two sets built in step 2 and 3 a graph representing the whole set of *TC* for the domain *d* and for all the languages is built. Figure 3 presents an excerpt oh this graph. The nodes of the graph correspond to all the pages and categories selected in steps 2 and 3 for all the involved languages. The edges, which are directional, correspond to all the links considered (category → category, category → page, page → category, page → page and interlingual links).

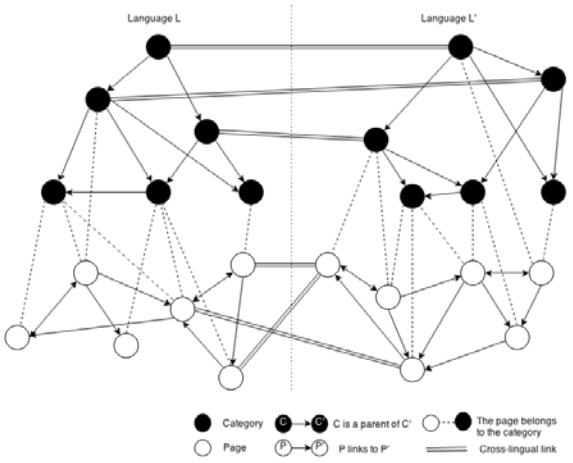


Figure 3: Graph representation of *TC*.

In step 5 the nodes of the graph of *TC* are scored. For doing so we use an algorithm inspired in Topic-Sensitive PageRank, (Haveliwala. 2002), in turn based on the original PageRank algorithm, (Page, Brin, 1998).

The original PageRank algorithm is based on a scoring mechanism that allows for a given node upgrading its score accordingly with the scores of its incident nodes. So in this setting all the incident edges are equally weighted and the new score is only affected by the old one and the scores of the incident nodes. As is discussed en section 4, this setting does not work very

well and we looked for some form of weighting of the edges, and not only of the nodes for computing the final score of a node.

In the case of nodes corresponding to pages there are three types of incident edges (for nodes corresponding to categories the formulas are similar):

- *il*: inlinks, links from other pages.
- *cp*: links from the categories the page belongs to.
- *ll*: langlinks, links from pages in other languages

The score of a page is computed by adding three weighted addends, one for each type of edge. The formula applied is the following:

$$PR(p) = F_{il} \sum_{il \in inlinks_p} \frac{PR(il)}{L(il)} + F_{cp} \sum_{c \in categories_p} \frac{PR(c)}{L(c)} + F_{ll} \sum_{ll \in langlinks_p} \frac{PR(ll)}{L(ll)}$$

where *PR(i)* is the PageRank score of node *i*, F_t are weights of edges of type *t* (*il*, *cp*, or *ll*), and L(n) are normalizing factors for pages or categories, computed as:

$$L(p) = F_{ol} \times |outlinks| + F_{pc} \times |cats| + F_{ll} \times |langlinks|$$

for pages and similarly for categories.

Finally, in step 6 the set of nodes corresponding to each language are sorted by descendent score giving the final result of the system. No distinction is made in this sorted sequence between *TC* corresponding to categories and these corresponding to pages.

4 Experiments and evaluation

4.1 Initial Settings

We performed some initial experiments for setting the parameters F_t defined in step 5. Finally we set F_{ll} and F_{cp} to 100 and the other parameters to 1. For evaluating these settings we limited ourselves to English and Spanish in the medical domain for which a golden repository of terms, *SNOMED*[8], is available. We consider four scenarios: i) *all_zeroes*, where no scoring procedure is used, ii) *all_ones*, where the standard *PageRank* algorithm is applied, iii) *no_langlinks*, where interlingual links weights are set to zero, i.e. the *TC* for each language are extracted independently, and, iv) *best*, where the setting described above was applied. The results are presented in Figures 4 , for English, and 5, for Spanish. All *PageRank* based scenarios clearly outperform the *all_zeroes* baseline. The differences between these scenarios are small for English but significant for Spanish where *best* outperforms clearly the others.

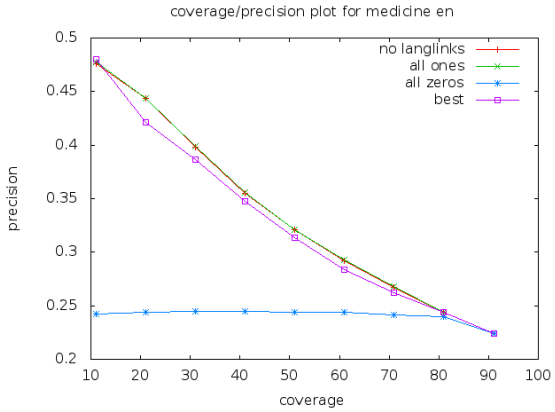


Figure 4: Initial experiments for English

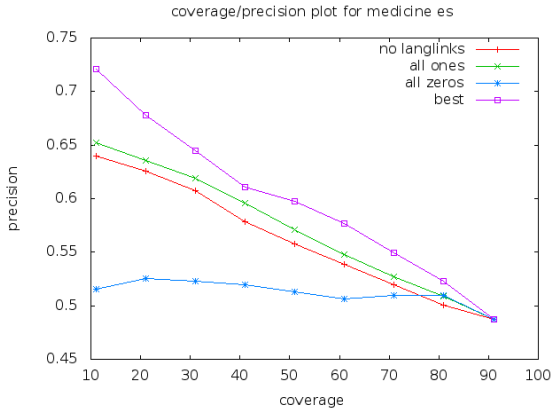


Figure 5: Initial experiments for Spanish

4.2 Experiments

We applied the procedure described in section 3 to the two domains and 4 languages using the setting of section 4.1. The results are presented in Table 1.

Language	Medicine	Finance
English	67,448	8,711
Spanish	8,872	1,310
Catalan	2,827	674
Arabic	7,318	1,557

Table 1: Overall results of our experiments

[8] http://www.ihtsdo.org/

The figures in Table 1 are not very informative. Being our system a ranker what is important is accepting as true terms the best ranked until some threshold. We depict, so, in Figures 6 (for medicine) and 7 (for finance) the distribution of *TC* in a coverage/score plots[9]. Content of these Figures and Table 1 are somewhat complementary.

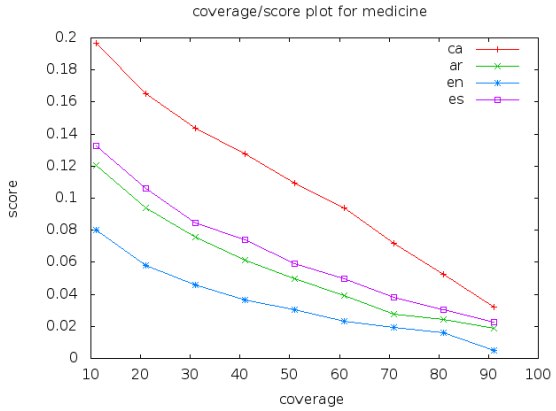


Figure 6: Results for medicine

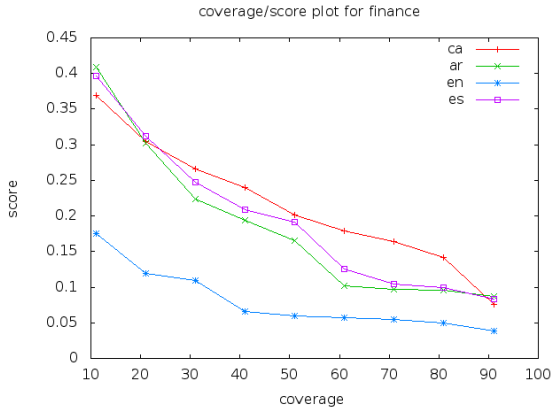


Figure 7: Results for finance

4.3 Evaluation

Evaluation of our results is not easy. For the pairs medicine/English and medicine/Spanish we can use as golden repository *SNOMED* and use as evaluation the results of the *best* curve in Figures 4 and 5. We have measured the correlation between precision in Figures 4 and 5 and score in Figure 6. Pearson's coefficient is 0.93 for Spanish and 0.98 for English, so we are pretty confident on our results for these two pairs. However, as pointed out in (Vivaldi, Rodríguez, 2012), *SNOMED* is far to be a reliable reference, for English only 62% of the correct *TC* were found in *SNOMED*. So the figures in Figures 4 and 5 can be considered a lower bound of the precision. For measuring a more accurate value we performed an additional manual validation[10] over the *TC* not found in *SNOMED* corresponding to the best 20% ranked ones. Figure 8 compares for this rank interval the precisions computed against *SNOMED* golden and those that combines it with the manual evaluation. At can be seen, results improvement is between 20 and 30 points.

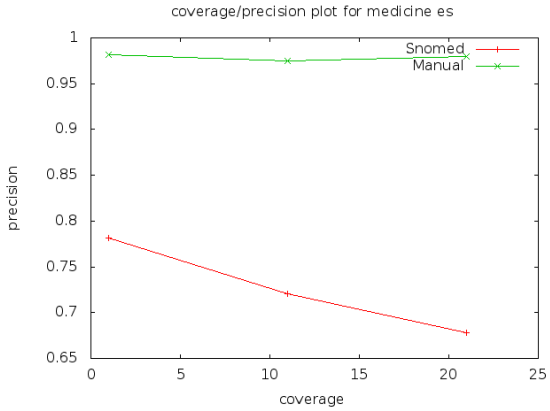


Figure 8: Comparison of SNOMED based and manual evaluation for Spanish.

Obviously all these evaluations are in some cases partial and in other cases partial. Could the evaluation results been extrapolated to other domains and/or languages. For having some insights we computed the Pearson's correlation coefficient between the non-cumulated ranked scores for the different languages. Table 2 shows the results for *medicine*. A very similar result has been obtained for *finance*. The high values of these coefficients seem to support out hypothesis. The score distribution correlates well between all the languages for all the domains. At the beginning of this section we saw that for *medicine* and for the languages English and Spanish scores and precision correlated well too. So our guess is that the evaluation based on *SNOMED* for English and Spanish and the manual one for a segment of Spanish can be likely been extended to the other cases.

A comparison with other systems is not possible globally but we can perform some

[9] Note that, contrary to Figures 4 and 5 where ordinates display precision, in this case ordinate display scores, i.e. *PR* values.

[10] Performed by the two authors independly, followed by a discusion on the cases with no agreement.

partial and indirect comparisons with the system closest to ours', (Vivaldi, Rodríguez, 2012). In this work, applied to Spanish and English, one of the domains included is *medicine* and *SNOMED* is used for evaluation. The main differences with ours' are that i) it is a term extractor, not a ranker and, ii) the evaluation is performed over terms belonging to *WordNet*. So the comparison has to be indirect. For the level of precision reported there, 0.2 for English, 0.4 for Spanish, the corresponding coverages in Figures 4 and 5 are 0.8 and 0.9. So, the number of terms we extract are 53,950 and 7,985 that clearly outperform largely the 21,073 and 4,083 reported there.

	en	es	ca	ar
en	1.0	0.996	0.990	0.992
es	0.996	1.0	0.995	0.994
ca	0.990	0.995	1.0	0.982
ar	0.992	0.994	0.982	1.0

Table 2: Correlations between non-cumulated ranked scores for the different languages

5 Conclusions and Future work

We have presented a terminology ranker, i.e. a system that provides a ranked list of terms for a given domain and language. The system is domain and language independent and uses as unique Knowledge Source the *Wikipedia* versions of the involved languages. The system proceeds in a cross-lingual way using for scoring a variant of the well known *PageRank* algorithm.

We have applied the system to four languages and two domains. The evaluation, though not complete, and somehow indirect, and the comparison with a recent system closely related to ours', at least at the level of the source, shows excellent results clearly outperforming the subjects of ours comparisons.

Future work includes i) the application of the system to other domains and, possibly, to other languages and, ii) the improvement of the evaluation setting applying the system to domains for which terminology exists.

6 Acknowledgements

Bibliography

A. Aronson, F. Lang (2010) An overview of MetaMap: historical perspective and recent advances. JAMIA 2010 17, p 229-236.

M.T. Cabré, R. Estopà, J. Vivaldi 2001. Automatic term detection. A review of current systems. *Recent Advances in Computational Terminology 2*, (2001) p. 53-87.4.3.

P. Drouin (2003) Term extraction using non-technical corpora as a point of leverage. Terminology 9(1), p. 99-115.

C. Enguehard, L. Pantera (1994) Automatic Natural Acquisition of a Terminology. Journal of Quantitative Linguistics 2(1), p. 27-32.

M. Erdmann, K. Nakayama, T. Hara, S. Nishio, 2009. Improving the extraction of bilingual terminology from Wikipedia. TOMCCAP 5(4) (2009)

M. Erdmann, K. Nakayama, T. Hara, S. Nishio, 2008. An Approach for Extracting Bilingual Terminology from Wikipedia. DASFAA 2008: 380-392

K. T., Frantzi, S, Ananiadou and J. Tsujii (2009). The C-value/NC-value Method of Automatic Recognition for Multi-word Terms. Lecture Notes in Computer Science, Volume 1513, 585-604

E. Gabrilovich, S. Markovitch, 2009. Wikipedia-based Semantic Interpretation for Natural Language Processing. *Journal of Artificial Intelligence Research* 34:443-498 (2009)

S. Hartmann, G. Szarvas, I. Gurevych, 2012. Mining Multiword Terms from Wikipedia. In M.T. Pazienza and A. Stellato: Semi-Automatic Ontology Development: Processes and Resources, p. 226--258, IGI Global, 2012

T. H. Haveliwala. 2002. Topic-sensitive PageRank. In *Proceedings of the 11th international conference on World Wide Web* (WWW '02). ACM, New York, NY, USA, 517-526.

U. Heid, S, Jauß, E. Krüger K. and A., Hofmann (1996) Term extraction with

standard tools for corpus exploration. Experiece from German. In Proceedings of Terminology and Knowledge Engineering (TKE'96). Berlin.

M. Alkhalifa, H. Rodríguez. 2010. Automatically Extending Named Entities coverage of Arabic WordNet using Wikipedia. In *International Journal on Information and Communication Technologies*, vol. 3, n. 3, June, 2010.

M. I. Krauthammer G. Nenadic, 2004. Term identification in the biomedical literature. In Jpournal of Biomed Inform. 2004 Dec;37(6) p. 512-26.

O. Medelyan, D. N. Milne, C. Legg and I. H. Witten, 2009. Mining meaning from Wikipedia. *International Journal of Human-Computer Studies*. 67(9): 716-754 (2009).

G. De Melo, G. Weikum, (2009) Untangling the Cross-Lingual Link Structure of Wikipedia, 48th Annual Meeting of the Association for Computational Linguistics, Uppsala, Sweden, 2010.

E. Niemann, I. Gurevych I. 2011. The People's Web meets Linguistic Knowledge: Automatic Sense Alignment of Wikipedia and WordNet}. In: Proceedings of the 9th International Conference on Computational Semantics, p. 205-214 (2011).

L. Page and S. Brin, 1998. The anatomy of a large-scale hypertextual web search engine. In Proceedings of the Seventh International Web Conference (WWW-98)

M.T. Pazienza, M. Pennacchiotti , F.M. Zanzotto (2005) Terminology Extraction: An Analysis of Linguistic and Statistical Approaches. *Studies in Fuzziness and Soft Computing* 185, p. 255-279

Y. W. Sabbah, Y. Abuzir, 2005. Automatic Term Extraction Using Statistical Techniques- A Comparative In-Depth Study and Application, In Proceedings of ACIT'2005

F. Sadat, 2011. Extracting the multilingual terminology from a web-based encyclopedia. RCIS 2011, p. 1-5

J. Vivaldi, 2001. Extracción de candidatos a término mediante combinación de estrategias heterogéneas. PhD Thesis, Universitat Politècnica de Catalunya.

J. Vivaldi, H. Rodríguez, 2008. Evaluation of terms and term extraction systems. A practical approach. *Terminology* 13(2): 225-248. John Benjamins.

J. Vivaldi, H. Rodríguez H. 2011. Using Wikipedia for term extraction in the biomedical domain: first experience. In Procesamiento del Lenguaje Natural 45, p. 251-254 (2011).

J. Vivaldi, H. Rodríguez H. 2012. Using Wikipedia for Domain Terms Extraction. In Gornostay, T. (ed.) Proceedings of CHAT 2012: The 2nd Workshop on the Creation, Harmonization and Application of Terminology Resources: co-located with TKE 2012;

E. Wolf, I. Gurevych, 2010. Aligning Sense Inventories in Wikipedia and WordNet. In: Proceedings of the First Workshop on Automated Knowledge Base Construction, p. 24-28, May 2010.

T. Zesch, C. Müller, I. Gurevych, 2008. Extracting Lexical Semantic Knowledge from Wikipedia and Wiktionary. In LREC 2008: Proceedings of the Conference on Language Resources and Evaluation, p. 1646-1652, Marrakech (2008).