DESARROLLO, IMPLEMENTACIÓN Y UTILIZACIÓN DE MODELOS PARA EL PROCESAMIENTO AUTOMÁTICO DE TEXTOS

Trabajos de las II Jornadas Argentinas de
Lingüística Informática: Modelización e Ingeniería

Compilado por
Víctor M. Castel

ISBN 987-575-019-0 del soporte *Internet*
Queda hecho el depósito que previene la Ley 11723.

Editorial de la
Facultad de Filosofía y Letras de la
Universidad Nacional de Cuyo

Centro Universitario - Parque Gral. San Martín
Casilla de Correo 345
5500 Mendoza, República Argentina
http://ffyl.uncu.edu.ar

Diseño de tapa: Verónica Bosio
Diagramación y diseño: Dani Brove
Mendoza, setiembre de 2005

ÍNDICE

INTRODUCCIÓN

Víctor M. Castel

Consejo Nacional de Investigaciones Científicas y Técnicas
Universidad Nacional de Cuyo
Mendoza, Argentina

Las *Primeras Jornadas Argentinas de Lingüística Informática: Modelización e Ingeniería* (JALIMI 2004) fueron organizadas por Gabriel G. Bès y Zulema Solana y se llevaron a cabo en la Facultad de Humanidades y Artes de la Universidad Nacional de Rosario entre el 23 y 25 de setiembre de 2004. Los objetivos de JALIMI 2004 fueron los siguientes: (1) presentar sintéticamente el estado actual de la lingüística informática en algunos de sus campos significativos, tanto en el plano de la modelización formal y el tratamiento automático, como en el de sus aplicaciones, en particular en el dominio de la ingeniería de la documentación; (2) favorecer o instaurar un diálogo efectivo en el campo abordado entre estudiantes avanzados o docentes de las áreas de la lingüística - no necesariamente informática o computacional -, de la computación y de la lógica; (3) favorecer o instaurar un diálogo efectivo en el campo del tratamiento automático de las lenguas entre la Universidad y el medio social: empresas, asociaciones o entidades públicas y privadas.

Las JALIMI 2005, realizadas en la Facultad de Filosofía y Letras de la Universidad Nacional de Cuyo entre el 12 y el 14 de setiembre de 2005, mantuvieron estos objetivos y se propusieron específicamente potenciar la interacción entre los recursos humanos, por cierto todavía muy escasos, dedicados al procesamiento automático de textos en la Argentina y Chile. Se continuó así con la idea de realizar reuniones relativamente pequeñas en cantidad de expositores pero con una muy activa participación de los mismos en la presentación de ponencias y talleres, de modo de privilegiar la profundidad en las discusiones frente a una lectura maratónica de una gran cantidad de trabajos.

El libro recoge ponencias y talleres seleccionados de JALIMI 2005 y está organizado en nueve Capítulos y un Apéndice. El Capítulo 1, "Clasificación de páginas *web* como posprocesamiento a la recuperación de la información", es de Cristina Bender, Claudia Deco y Liliana Perló; el Capítulo 2, "Conocimiento de la lengua y técnicas estadísticas en el análisis lingüístico", de Gabriel G. Bès, Zulema Sola y Celina Beltrán; el Capítulo 3, "Determinación dinámica de valores de verdad de condiciones de reglas de generación de textos", de Víctor M. Castel; el Capítulo 4, "Expansión de consultas utilizando recursos lingüísticos para mejorar la recuperación de información en la *web*", de Claudia Deco, Cristina Bender, Jorge Saer y Mario Chiari; el Capítulo 5, "Ejercicios de traducción automática catalán - castellano", de Gustavo A. González; el Capítulo 6, "Lingüística computacional: del prototipo a la aplicación", de Daniel E. Guillot; el Capítulo 7, "El sintagma nominal núcleo", de Zulema Solana y Andrea Rodrigo; el Capítulo 8, "Modelización de las fuentes declarativas en una herramienta de análisis y conjugación automáticos de verbos del español", de Zulema Solana, Rodolfo Bonino y Viviana Valenti; y el Capítulo 9, "Compactando el Cast3LB", de Demetrio Vilela y Gabriel Infante-López. El Apéndice contiene los resúmenes de dos Talleres: "La obtención de límites de oraciones", de Celina Beltrán y Gabriel G. Bès, y "El Grial. Interfaz computacional de anotación e interrogación de corpus en español: algunos resultados de su aplicación", de Giovanni Parodi S.

Si bien hay sustantivas diferencias en los enfoques, las metodologías, las propiedades específicas estudiadas y las aplicaciones propuestas o proyectadas, todos estos trabajos comunican resultados de investigaciones que pretenden contribuir a alcanzar el objetivo a largo plazo de la Lingüística Informática, a saber: emular en términos cibernéticos la extraordinaria capacidad humana de producir y comprender textos en lengua natural.

Capítulo 1

CLASIFICACIÓN DE PÁGINAS *WEB* COMO POSPROCESAMIENTO A LA RECUPERACIÓN DE LA INFORMACIÓN

Cristina Bender, Claudia Deco y Liliana Perló

En Víctor M. Castel, Comp. (2005) *Desarrollo, implementación y utilización de modelos para el procesamiento automático de textos*. Mendoza: Editorial de la Facultad de Filosofía y Letras, UNCuyo: 1-12.
ISBN 987-575-019-0 del soporte Internet

Clasificación de páginas *web* como posprocesamiento a la recuperación de información

Cristina Bender, Claudia Deco y Liliana Perló

Universidad Nacional de Rosario
Facultad de Ciencias Exactas, Ingeniería y Agrimensura
Departamento de Sistemas e Informática
Rosario, Argentina
{bender,deco}@fceia.unr.edu.ar

Resumen

Una búsqueda en la *web* devuelve un único conjunto de documentos ordenados por el motor de búsqueda. Este resultado puede ser desalentador para el usuario, que debe examinar toda la lista a fin de determinar aquellas páginas que le pueden ser útiles. Los usuarios prefieren con mayor frecuencia navegar a través de directorios de contenido pre-clasificado. La gran cantidad de documentos que necesitan ser clasificados es una importante motivación para la búsqueda de métodos que categoricen las páginas automáticamente. La categorización automática de documentos es el proceso de asignar a documentos no vistos categorías predefinidas y es una importante tarea que puede ayudar en la organización de los mismos. Las técnicas y métodos para efectuar esta tarea generalmente son usadas en forma individual y muchas veces no brindan los resultados esperados. En este trabajo se presentan diversas formas de combinar técnicas de clasificación y se analizan los resultados obtenidos al aplicar dichas técnicas combinadas a las páginas *web* resultantes de una consulta, con el fin de mejorar la calidad en las clasificaciones con respecto a los resultados obtenidos con la aplicación de las técnicas en forma aislada.

1 Introducción

Desde el nacimiento de la *World Wide Web*, la cantidad de información disponible en ella se ha ido incrementando exponencialmente. La meta principal de la Recuperación de Información en la Web, es recuperar todos los documentos que sean relevantes a una consulta de un usuario. Los documentos resultantes son presentados al usuario como un único conjunto de documentos ordenados en un ranking de importancia según algún criterio preestablecido por el motor de búsqueda que se esté usando. El usuario debe examinar toda la lista a fin de determinar aquellas páginas que le pueden ser realmente útiles. En general los algoritmos de ranking no tienen forma de deducir la intención de búsqueda del usuario. Aunque la aplicación de técnicas de Recuperación de Información mejora los resultados de la búsqueda en la *web*, dada la cantidad de documentos existentes en este repositorio de información, los resultados pueden ser desalentadores para los usuarios.

Un reciente estudio (Yang et al. 1999) muestra que los usuarios prefieren con mayor frecuencia navegar a través de directorios de contenido preclasificado que provean una vista basada en categorías para los documentos recuperados, lo cual les permite encontrar mayor cantidad de información relevante en menor tiempo. Una primera aproximación para facilitarle al usuario la lectura de los resultados de su búsqueda es el uso de métodos de clasificación o categorización de documentos, aplicado a los resultados del buscador. El usuario recibe entonces, como resultado de una consulta, documentos agrupados de acuerdo a una medida de similitud entre ellos, y esto le permite descartar rápidamente los grupos que no le sean relevantes, y por lo tanto explorar una extensa colección de documentos más eficientemente.

Tradicionalmente, las investigaciones en categorización de texto se basaron en el texto del documento, ignorando la información adicional provista por la forma en que las páginas *html* se estructuran, por ejemplo los meta-*tags* y los enlaces entre las páginas. De acuerdo a un primer estudio efectuado en (Motz et al. 2003), dichas técnicas son mayormente utilizadas en forma individual, aunque es posible combinarlas de diversas maneras. Algunas de estas combinaciones son más adecuadas que otras para cumplir con los requisitos básicos de tiempo y *performance*, y permiten mejorar los resultados de la clasificación.

El objetivo principal de este trabajo es clasificar los resultados de una consulta para que los usuarios puedan enfocar su búsqueda de información con mayor precisión, y así mejorar la Recuperación de Información. En primer lugar, se analizan técnicas de clasificación aplicadas en forma individual y luego se analizan los resultados obtenidos al combinar estas técnicas.

2 Conceptos básicos

La clasificación de los resultados de una búsqueda permite explorar una extensa colección de documentos más eficientemente, porque el usuario recibe, como resultado de su consulta, documentos agrupados de acuerdo a una medida de similitud entre ellos. Para efectuar la agrupación de los documentos se utilizan métodos de análisis de grupos. Estos métodos pueden dividirse en no exclusivos y exclusivos. Si un método es exclusivo significa que un objeto debe pertenecer a un solo grupo, aún cuando pueda ser asignado a dos grupos de acuerdo a los valores resultantes al aplicar el criterio de comparación. En el caso de los métodos no exclusivos un documento puede pertenecer a más de una clase simultáneamente.

Al mismo tiempo, hay dos formas claramente diferenciadas de realizar la clasificación: estática y dinámica. Para la clasificación estática de documentos, también llamada categorización, se tienen categorías o clases predefinidas y la tarea consiste en asignar una de estas clases a cada uno de los documentos. Los métodos que utilizan un conjunto de categorías definido estáticamente no permiten agregar una nueva categoría o eliminar alguna. Las categorías definidas en una etapa temprana puede que no capturen de manera satisfactoria los documentos actuales. Por esto, se definieron métodos capaces de generar categorías dinámicamente, de acuerdo a las características propias del grupo de documentos que se desea clasificar en un momento dado. Esto se conoce como clasificación dinámica.

El término clasificación, al hablar de documentos, es usado frecuentemente para referirse a dos tipos de análisis diferentes: categorización y *clustering*. La categorización puede ser vista como la asignación de documentos o partes de documentos en una categoría de un conjunto fijo y predefinido de ellas. Usualmente este conjunto es creado al comienzo de la tarea de clasificación y permanece constante durante el transcurso de la misma. La mayoría de los métodos no proveen mecanismos para la circunstancia que la estructura de las categorías cambie, por ejemplo con el descubrimiento de una nueva categoría, o la unión de dos o más de ellas en una sola. Los métodos de categorización más usados son árboles de decisión, reglas de decisión, k vecinos más cercanos, redes *bayesianas*, redes neuronales (Farkas 1994), métodos basados en regresión (Lam y Low 1997) y métodos basados en vectores (Joachims 1998). Una descripción detallada de estos métodos puede encontrarse en (Mitchell 1997). En contraste con la categorización, el *clustering* es un procedimiento que no está basado en una estructura predefinida de conocimiento, puesto que su objetivo es agrupar en cada clase objetos similares entre sí y disímiles del resto. Para esto se exploran las similitudes en los contenidos de los documentos y se los agrupa de acuerdo a sus propiedades (Mitchell 1997). Contrariamente a las técnicas de categorización, los algoritmos de *clustering* permiten determinar las categorías dinámicamente.

La mayor parte de los métodos de clasificación y *clustering* asumen que los datos son independientes e idénticamente distribuidos y se diseñaron pensando en una estructura plana de los datos. Sin embargo, en las páginas *web* encontramos una estructura mucho más rica. Una cuestión interesante que aún se sigue investigando, es cómo explotar la información que se tiene sobre la estructura de los documentos individuales, o la estructura de una colección de ellos, es

decir sus interconexiones, para optimizar la clasificación. El lenguaje de marcado *html* otorga una forma de añadir más estructura a la representación del documento. Algunas palabras son marcadas como título, otras como párrafo, otras como enlaces a otras páginas. Es razonable creer que una palabra del título de la página pueda aportar más información sobre el tema al que una página refiere, que una palabra cualquiera del cuerpo. A su vez, la información de las páginas cercanas puede ser útil para decidir qué categoría le corresponde a una determinada página. Se puede encontrar mucha información en los enlaces entrantes. Una página puede entonces ser clasificada por los enlaces que le llegan, que suelen ir acompañados de una pequeña descripción de la página o junto a otros enlaces similares. También hay información en los enlaces salientes. El texto de las descripciones (*anchors*) de los enlaces a una página puede aportar datos importantes para determinar el tema de dicha página (Hearst 1999). En la Tabla 1 se presentan los análisis efectuados (Motz et al. 2003) a fin de determinar cuáles de las características de las páginas *web* son más adecuadas para realizar la clasificación, siempre buscando mejorar el tiempo de respuesta y la calidad de la clasificación.

Característica analizada	Descripción	Aspectos favorables	Aspectos desfavorables
Dirección de la página web	La dirección de una página posee extensiones que permiten identificar mediante un grupo de tres letras (los más frecuentes son com, edu, gov, org y net) el tipo o clase de la página, y mediante un grupo de dos letras (por ejemplo ar) el país de origen de la página.	– La extracción de la extensión de tres letras puede hacerse directamente de los resultados del buscador. – El proceso para la extracción es simple y rápido.	– En algunas consultas, aproximadamente el 40% del total de páginas recuperadas no tienen extensiones como las indicadas, pero no pueden descartarse. ⇒ no se utiliza esta característica para la determinación de las categorías
Párrafo más significativo de la página web	La información contenida en ciertos párrafos de una página suele ser altamente representativa del tema al que ésta refiere y por ende a la categoría a la que podría pertenecer. Para obtener el párrafo más significativo de un documento, se puede comparar la importancia de las etiquetas intervinientes (por ejemplo, del tipo <Hn>) y el tamaño de las tipografías. El algoritmo para la determinación del párrafo más significativo se encuentra en (Motz et al. 2003).	– Identificar las etiquetas y el tamaño de las tipografías es un proceso simple y rápido. – Establecer la jerarquía de etiquetas es un proceso simple y rápido. – En búsquedas donde previamente se realizó una expansión de la consulta, la determinación del párrafo más significativo fue exitosa.	– La importancia de las etiquetas es relativa a cada página, y debe establecerse la jerarquía de etiquetas y tipografías en forma independiente sobre cada una. – En búsquedas realizadas sin expansión de la consulta, sólo se pudo determinar el párrafo más significativo en un 60% de las mismas. ⇒ no se utiliza esta característica para la clasificación.
Etiquetas de meta data de la página web	El código HTML posee diversas etiquetas (ó tags) que permiten marcar palabras, frases ó párrafos como pertenecientes a un título, las keywords, el ó los autores, y la descripción de la página, entre otros.	– Los procesos de identificación de las etiquetas y de extracción de los términos, son simples y rápidos. – En un altísimo porcentaje de páginas (98%) se encontraron las etiquetas que marcan ciertos términos como título del documento (términos entre <title> y </title>). ⇒ aunque los títulos tienen pocas palabras (entre 1 y 10), se usará para la clasificación.	– Las etiquetas correspondientes a keywords, autor y descripción de la página, sólo se observaron entre un 10% y un 50% de las páginas resultantes. Si bien las palabras y frases identificadas como keywords y descripción son muy representativas de la categoría a la que la página pertenece, fueron descartados.
Interconexiones entre páginas web	Las páginas web están dentro de una red de interconexiones. Poseen enlaces entrantes (in-links) desde otras páginas hacia ella, generalmente acompañados de una descripción de la página a la que referencian. Los in-links pueden ser absolutos o relativos. Los relativos permiten la navegación dentro de la misma página y el texto que los acompaña, si es que existe, no suele aportar información útil. Los in-links absolutos son hacia una página diferente, y son estos los que suelen contener palabras más representativas de la clase de la página enlazada. Las páginas también poseen enlaces salientes (out-links) desde la página a analizar hacia otras y también están acompañados por un breve texto.	Enlaces salientes (out-links): – Están presentes en la mayoría de las páginas. – El proceso de extracción es simple. ⇒ los enlaces salientes (out-links) serán utilizados en la tarea de clasificación.	Enlaces salientes (out-links): – Para extraerlos se debe acceder y recorrer en toda su extensión el código HTML de la página. El costo de procesamiento es alto. Enlaces entrantes (in-links) – El texto que acompaña a los in-links relativos no suele aportar información útil. – Acceder al texto de los in-links absolutos obliga a acceder a otras páginas, por lo que no se necesita un motor de búsqueda que permita obtener la lista de páginas con enlaces a nuestra página. – Los in-links absolutos se observaron sólo en aproximadamente el 10% de las páginas, y generalmente sólo en las 20 primeras páginas.
Descripciones dadas por los buscadores	Los motores de búsqueda suelen presentar los resultados mediante una lista de páginas en algún orden establecido por el propio motor, y acompañando el enlace a la página con una breve descripción de la misma. Estas descripciones suelen tener aproximadamente entre 10 y 30 palabras en todos los motores utilizados y contienen información útil para identificar la clase de página.	– El proceso de extracción es simple. – No es necesario entrar a las páginas a clasificar, dado que el texto se obtiene directamente a partir de las páginas otorgadas por el buscador, con lo cual disminuyen los tiempos de procesamiento. ⇒ esta característica se utilizará para la clasificación.	

Tabla 1. Características de las páginas resultantes de una consulta.

De acuerdo a estos análisis, la extracción de las palabras representativas en este trabajo se efectúa de: el Título de la página (aquellas palabras entre las etiquetas <title> y </title>), la Descripción de la página otorgada por el buscador, y los Enlaces salientes (*anchors* de los enlaces de la página a clasificar hacia otras). Estos elementos están presentes en un alto porcentaje de páginas, y a la vez que se proveen una gran cantidad de palabras.

Para la clasificación, en este trabajo, se utiliza un método de categorización no exclusivo, debido a que éstos usan algoritmos más simples, lo cual agiliza mucho los cálculos. Dentro de los algoritmos posibles de categorización, se opta por uno basado en vectores, ya que brinda una alta *performance*. Para aplicar el algoritmo basado en vectores, se necesita generar un *diccionario* para cada una de las categorías. Un diccionario es una lista de palabras representativas de la categoría, que serán luego comparadas con las palabras que se extraigan de

las páginas para determinar con cuál de las categorías tienen más palabras en común. Por ejemplo, el diccionario generado para la categoría *Educación* contiene los siguientes términos:

{ actividad, aplicar, asignatura, base, computación, dato, *desarrollo*, descripción, doctorado, especialización, estudio, facultad, informática, ingeniería, introducción, *investigación*, maestría, modal, modular, plan, programa, *proyecto*, sistema, software }

Los términos obtenidos no pueden usarse directamente, ya que contienen numerosas palabras superfluas, verbos conjugados, números, etc.. Por esto, se aplican técnicas de *stopword* y *stemming* a fin de reducir la dimensión de la lista resultante. La técnica de *stopword* permite remover palabras tales como artículos y preposiciones, que no aportan información útil. Usualmente, se realiza comparando las palabras del diccionario con una lista predefinida de palabras no significativas y eliminando las concordantes (Baeza 1999). Además, muchas de estas palabras no pueden usarse directamente para generar el diccionario, ya que la magnitud del mismo sería inmanejable si consideramos todas las palabras de una misma familia; como por ejemplo: educación, educativo, educativos, educador, etc. Para solucionar esto, se busca la raíz común entre todos estos términos. La técnica conocida como *stemming* permite extraer sufijos y prefijos comunes, de tal forma que, palabras que literalmente son diferentes pero tienen una raíz común, se consideran como un solo término en base a su raíz. Para este trabajo se usa una adaptación del algoritmo de Porter para palabras en castellano (Figuerola et al 2001) (Pannesi y Bordignon 2001). Con este pre-procesamiento se logra normalizar las palabras y reducir el tamaño de los diccionarios.

El siguiente paso es generar los vectores representativos de cada documento. Estos se construyen en función del diccionario y de las palabras que aparecen en el sector del documento del cual se extraerán los términos. El vector representativo de un documento, de dimensión igual al diccionario, indica la frecuencia de aparición de cada palabra del diccionario en el documento: la componente i-ésima indica la frecuencia absoluta de aparición en el documento del término i-ésimo del diccionario (Baeza 1999). Por ejemplo, el vector representativo para una página cuyo título es: *Proyectos de investigación en desarrollo*, es:

{ 0, 0, 0, 0, 0, 0, *1*, 0, 0, 0, 0, 0, 0, 0, 0, *1*, 0, 0, 0, 0, 0, *1*, 0, 0 }

Luego se agrupan los documentos de acuerdo a un cierto grado de similitud entre ellos. Este grado de similitud suele asociarse a una medida matemática de distancia. De acuerdo al método de generación de vectores usado, cada documento es representado por un vector. Por lo tanto, la medida de similitud entre documentos se transforma en una distancia entre vectores. Puede utilizarse la distancia euclídea u otras. Otra posibilidad es usar el producto escalar como medida de similitud. Habiendo definido una noción de similitud, el problema consiste en formar grupos de documentos llamados *clusters*, donde cada uno contiene documentos muy similares entre sí, y relativamente diferentes al resto. Para solucionar este problema existen diversos algoritmos o métodos de análisis de clusters. Se decidió usar el algoritmo particionante, H-medias, debido a que su tiempo de ejecución es bajo por la simplicidad de los cómputos que realiza.

El siguiente paso es comparar los resultados obtenidos de los productos de los vectores representativos de cada documento con los vectores diccionarios de cada categoría a fin de determinar el mayor de los valores. Este corresponderá a la categoría a asignar a ese documento.

3. Descripción de la experiencia

Para el intercambio de datos entre los distintos módulos se utiliza XML (*eXtensible Marckup Language*). Para la extracción de información contenida en las páginas *web* se usa JEDI (*Java based Extraction and Dissemination of Information*) (Huck et al. 1998) que es una herramienta que utiliza gramáticas con atributos, evaluadas a través de una estrategia tolerante a fallas para tratar con gramáticas ambiguas y fuentes de información irregulares. Este enfoque se adapta especialmente bien a páginas html, donde es muy frecuente la aparición de irregularidades en

documentos con una cierta estructura. En las diferentes etapas de la clasificación se hace un uso extensivo de las principales características de JEDI para obtener información a través de reglas desarrolladas para cada caso. El proceso de clasificación se esquematiza en la Figura 1.

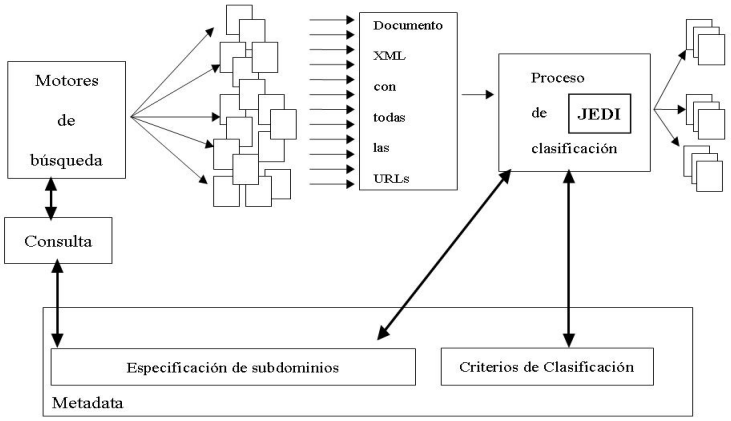


Figura 1. Esquema general de la clasificación.

Consideremos que se desea recuperar información sobre: *ingeniería de software aplicada a proyectos*. Realizada esta consulta en un buscador, se obtiene una lista de enlaces resultantes que se subclasifican de acuerdo a la especificación de subdominios contenida en la *metadata*. Para poder hacer esta clasificación automática, los resultados obtenidos a partir del buscador deben procesarse para dejarlos en formato de intercambio de datos. La Figura 2 muestra, en forma abreviada, el documento XML resultado de este procesamiento. Junto con los enlaces a las páginas, también se extraen las descripciones dadas por el buscador para cada una de las páginas, y con la información necesaria extraída de títulos y enlaces salientes.

```
<links>
    <consulta>seguridad informatica</consulta>
    <link>www.hispasec.com</link>
        <descripcion>Noticias Diarias y toda la informacion sobre SEGURIDAD INFORMATICA Virus,
            Criptologia, antivirus, software, internet, etc</descripcion>
        <titulo>Hispasec - Seguridad Informática</titulo>
        <out-links>una-al-dia CheckDialer Análisis Utilidades Online Búsqueda Auditorías Consultorías
            Formación S.A.N.A Consultar noticias anterioresGuia supervivencia Windows XP
            Comparativa Antivirus 2001 Comparativa Antivirus 2000 Chroot y seguridad Más
            información Curso de Seguridad Internet en servidores Windows (NT/2000) </out-links>
    <link>www.virusprot.com</link>
        <descripcion>Tiene como objetivo cubrirel tema de la seguridad informática</descripcion>
        <titulo>VIRUSPROT.COM - El Sitio Líder en Seguridad Informática</titulo>
        <out-links></out-links>
    ….
</links>
```

Figura 2. Documento XML generado a partir de una consulta.

Para la experiencia se estableció el uso de cuatro categorías *disjuntas*, para evitar que alguna página pueda pertenecer a más de una de las categorías a la vez. Las cuatro categorías son*: Comercial, Educativos, Publicaciones y Noticias*. Realizados los productos internos entre los vectores representativos de cada página y los vectores diccionario de las categorías, el mayor de los valores corresponde a la categoría a asignar. Si los cuatro productos resultan ser nulos, la página no puede asignarse a ninguna de las cuatro clases, y recae en una categoría *Otros.*

Todas estas tareas se realizaron en tres motores de búsqueda a fin de evaluar y comparar la

eficiencia y eficacia de la aplicación de las clasificaciones en forma individual en títulos, enlaces salientes y descripciones de las páginas brindadas por el buscador, así como los resultados al combinar las distintas formas de clasificación.

Para evaluar la *performance* de la combinación de los métodos, se proponen dos indicadores que miden qué tan bien se ajustan los resultados de cada método con los de una clasificación manual. Los indicadores propuestos son:

I1: Ratio de la cantidad de páginas clasificadas automáticamente sobre la cantidad de páginas clasificadas manualmente.

I2: Ratio de la cantidad de páginas clasificadas por el método automático que coinciden con la clasificación manual sobre la cantidad de páginas clasificadas manualmente.

Para el cálculo de estos indicadores se necesitan conocer ciertos valores que se definen de la siguiente forma:

- Número de *páginas clasificadas manualmente*: aquellas páginas que corresponden a alguna de las cuatro categorías en la clasificación manual.
- Número de *páginas clasificadas automáticamente*: aquellas páginas que caen en alguna de las cuatro categorías en la clasificación automática en estudio.
- Número de *páginas bien clasificadas automáticamente*: páginas que son correctamente asignadas a alguna de las cuatro categorías por el método automático. Es decir, que coinciden en su clasificación con la clasificación manual hecha previamente.

Estos indicadores se calcularon para todos los métodos combinados, a fin de determinar cuáles de ellos tienen mejor *performance*. Esto es, para qué combinaciones *ambos* indicadores, I1 y I2, tienen los valores más altos. Para este cálculo no se consideraron las categorías separadamente, puesto que interesa determinar, para cada método, qué tan bien se ubican las páginas en las cuatro categorías propuestas.

4. Resultados de la experiencia

Se realizó la consulta: *seguridad informática*, en tres motores de búsqueda: Altavista, *Yahoo*! y *Google*. Para el análisis se tomaron las 250 primeras páginas en español del total de páginas retornadas por cada buscador, que fue diferente en cada caso: Altavista retornó 23.398 páginas, *Yahoo*! 70.300 páginas y *Google* 86.000.

Antes de realizar las tareas de clasificación automáticas, y con el fin de probar luego la eficacia de cada una, los resultados ante dicha consulta fueron clasificados en forma manual.

	Clasificación	Noticias	Comercial	Publicac.	Educativo	Otros	Error
GOOGLE	***Manual***	***12.40***	***26.80***	***28.80***	***18.40***	***11.60***	---
	Una etapa: título	4.40	22.00	8.00	14.40	**48.80**	**3.10**
	Una etapa: descripción	7.60	23.20	17.20	18.40	**31.60**	10.30
	Una etapa: out-links	8.40	27.20	16.80	14.80	**30.80**	11.00
YAHOO	**Clasificación**	**Noticias**	**Comercial**	**Publicac.**	**Educativo**	**Otros**	**Error**
	Manual	***18.00***	***13.20***	***33.20***	***16.80***	***16.40***	---
	Una etapa: título	10.40	9.60	6.40	14.40	**56.80**	18.80
	Una etapa: descripción	6.80	10.80	9.60	16.00	**54.40**	14.40
	Una etapa: out-links	15.60	10.80	15.60	21.60	**34.00**	20.40
ALTAVISTA	**Clasificación**	**Noticias**	**Comercial**	**Publicac.**	**Educativo**	**Otros**	**Error**
	Manual	***14.40***	***32.40***	***20.00***	***17.20***	***12.00***	---
	Una etapa: título	3.60	20.40	2.80	8.40	**60.80**	**9.60**
	Una etapa: descripción	8.00	22.40	9.20	12.00	**44.40**	17.20
	Una etapa: out-links	9.60	31.60	12.00	12.80	**30.00**	16.40

Tabla 2. Resultados de la clasificación manual y la clasificación automática independiente por título, por descripción y por enlaces salientes para los motores Altavista, *Yahoo*! y *Google*.

Los resultados obtenidos para los motores de búsqueda Altavista, *Yahoo*! y *Google* se muestran en la Tabla 2. Los valores corresponden a porcentajes sobre el total de páginas analizadas, es decir, sobre las primeras 250. Una vez obtenidos los tres vectores representativos: uno para el título, uno para la descripción dada por el motor de búsqueda y uno para los enlaces salientes; para cada una de las 250 páginas, y habiendo efectuado los productos internos, se obtienen las tres clasificaciones, en forma independiente unas de otras. Estas clasificaciones se comparan con las clasificaciones efectuadas en forma manual para establecer cuán precisas han sido. Una clase otorgada a una página se considera errónea si difiere de la clase que le asignó la clasificación manual. La columna correspondiente a *Error* muestra el porcentaje de páginas mal clasificadas sobre el total de páginas asignadas a una de las cuatro categorías principales.

Tanto para Altavista como para *Google* la clasificación por títulos es la que proporcionó el menor error porcentual, pero dejó gran parte de las páginas sin clasificar (aquellas que se agrupan en la categoría *Otros*). La clasificación por enlaces salientes, ó out-links, en los tres casos es la que dejó menos páginas sin clasificar, aunque esta cantidad, de aproximadamente el 30% para los tres motores, es igualmente alta. En general los mejores resultados se dieron para *Google*, sin embargo, para todos los buscadores quedan entre el 30 y el 60% de las páginas sin clasificar, cuando estos valores debieran variar entre el 11 y el 17% según la clasificación manual que se efectuó previamente.

Con estos primeros resultados se observa que las técnicas de clasificación que usualmente se realizan en forma independiente no brindan buenas soluciones. Por lo tanto, se realizan diferentes formas de combinación de estas técnicas a fin de lograr una mejor clasificación.

Una primera forma de combinación es asignarle a cada página el *resultado más frecuente* entre los retornados por el título, la descripción del motor y los out-links. Si no se tiene un resultado más frecuente, ya sea porque las tres técnicas dieron resultados diferentes, o porque alguna de ellas no retornó ninguna clasificación para una página determinada, se toma directamente uno de los tres valores, dependiendo del motor del que se esté tratando. Cuáles son las clases que deben tomarse para cada motor, se determinó en base a los resultados obtenidos, comparándolos con la clasificación manual, para determinar en cada caso qué método proporciona la menor cantidad de error. Así, se determinó que las clasificaciones a considerar en caso de que no haya resultado más frecuente, para *Google* será tomar el resultado dado por título y en caso de ser este nulo, tomar el dado por la descripción; para Altavista se tomará el resultado dado por título y en segunda instancia el dado por los enlaces salientes; y para *Yahoo*! se tomará el resultado dado por la descripción en primera instancia y el dado por título en caso de que esta sea nula. Los resultados obtenidos para la clasificación mediante el método de resultado más frecuente se muestran en la Tabla 3. Como se observa, las clasificaciones mejoraron. Si bien para Altavista la cantidad de páginas sin clasificar sigue siendo alta, para *Google* y *Yahoo*! los resultados se aproximan más a los reales que aquellos alcanzados con las clasificaciones individuales. La cantidad de páginas agrupadas en *Otros* llegó a bajar en órdenes de entre 20 y 40%, con un bajo porcentual de error para *Google* (14%) y Altavista (16%).

Otra forma de combinación es *generando nuevos vectores*. Las palabras extraídas del título, la descripción y los enlaces salientes se combinan para constituir un único vector de forma tal que todos los términos colaboren conjuntamente a conferir la categoría a la clase. En la Tabla 3 se observa que los resultados mejoraron bastante, en especial para *Google* y para Altavista. En todos los casos bajó significativamente el porcentaje de páginas sin clasificar, aunque para *Google* y Altavista se elevó el porcentaje de error.

Finalmente, pueden *aplicarse las técnicas de clasificación en etapas sucesivas*. En esta experimentación, se tomó una de las técnicas, por ejemplo título, y a aquellas páginas que con este método cayeron en la categoría *Otros* se les aplicó las otras técnicas independientemente, teniéndose dos nuevas clasificaciones. Se tiene así seis combinaciones diferentes: Título en la primera etapa, Descripción del motor en la segunda; Título en la primera etapa, Enlaces salientes en la segunda; Descripción y luego Título; Descripción y luego Enlaces salientes; Enlaces salientes y luego Título; Enlaces salientes y luego Descripción. Si bien para *Yahoo*! no se encontraron los mejores resultados, para **Google** y Altavista algunas de las combinaciones

fueron muy positivas. En especial, para ambos motores, las combinaciones de enlaces salientes con descripción del motor, en los dos órdenes posibles.

	Clasificación	Noticias	Comercial	Publicac.	Educativo	Otros	Error
GOOGLE	*Manual*	*12.40*	*26.80*	*28.80*	*18.40*	*11.60*	—
	Resultado más frecuente	8.40	33.20	24.00	20.40	9.00	14.20
	Vector combinado	8.80	32.80	24.00	21.60	**7.80**	16.90
	Dos etapas: título y descripción	7.60	27.20	16.40	19.20	24.60	8.80
	Dos etapas: título y out-links	6.40	32.40	17.20	19.20	22.80	10.50
	Dos etapas: descripción y título	7.20	28.20	16.80	20.00	22.80	12.00
	Dos etapas: descripción y out-links	**8.40**	**31.20**	**24.00**	**24.20**	**10.20**	**16.50**
	Dos etapas: out-links y título	8.40	31.20	24.00	21.20	10.20	16.50
	Dos etapas: out-links y descripción	**8.80**	**32.40**	**24.40**	**17.20**	**12.20**	**16.80**
	Tres etapas: título, descripción y out-links	8.80	27.60	22.40	21.20	15.00	18.60
	Tres etapas: título, out-links y descripción	8.80	35.20	22.80	20.00	8.20	17.60
	Tres etapas: descripción, título y out-links	8.40	34.00	22.80	21.60	8.20	20.80
	Tres etapas: descripción, out-links y título	9.20	31.60	24.00	22.40	7.80	22.40
	Tres etapas: out-links, título y descripción	9.20	32.80	24.40	20.00	8.60	18.60
	Tres etapas: out-links, descripción y título	9.20	34.40	24.40	19.20	7.80	19.80
YAHOO	Clasificación	Noticias	Comercial	Publicac.	Educativo	Otros	Error
	Manual	*18.00*	*13.20*	*33.20*	*16.80*	*16.40*	—
	Resultado más frecuente	15.60	30.40	18.00	15.20	18.40	28.40
	Vector combinado	15.60	26.40	21.60	17.20	**16.80**	25.20
	Dos etapas: título y descripción	13.60	20.40	10.00	11.20	42.40	7.20
	Dos etapas: título y out-links	10.00	28.40	13.60	12.00	27.60	30.00
	Dos etapas: descripción y título	10.00	18.40	14.00	13.20	42.00	20.40
	Dos etapas: descripción y out-links	10.40	27.60	16.00	15.20	28.40	23.60
	Dos etapas: out-links y título	16.40	23.20	17.60	13.20	27.20	24.00
	Dos etapas: out-links y descripción	16.00	17.60	18.00	12.80	33.20	22.80
	Tres etapas: título, descripción y out-links	14.40	28.40	16.00	15.60	23.30	33.60
	Tres etapas: título, out-links y descripción	16.40	29.60	12.80	12.40	26.40	31.60
	Tres etapas: descripción, título y out-links	10.80	25.20	17.60	14.40	29.60	27.60
	Tres etapas: descripción, out-links y título	11.60	28.00	18.40	16.00	23.60	25.20
	Tres etapas: out-links, título y descripción	16.80	24.40	19.20	13.60	23.60	25.60
	Tres etapas: out-links, descripción y título	17.20	18.00	20.40	13.60	28.40	24.40
ALTAVISTA	Clasificación	Noticias	Comercial	Publicac.	Educativo	Otros	Error
	Manual	*14.40*	*32.40*	*20.00*	*17.20*	*12.00*	—
	Resultado más frecuente	10.80	37.20	13.60	12.80	30.00	16.40
	Vector combinado	11.20	36.00	14.80	18.40	**15.60**	20.40
	Dos etapas: título y descripción	6.80	29.20	8.00	15.20	36.80	18.00
	Dos etapas: título y out-links	9.20	38.80	10.40	14.40	23.20	16.40
	Dos etapas: descripción y título	8.40	26.00	10.00	15.20	36.40	17.60
	Dos etapas: descripción y out-links	**12.40**	**35.20**	**15.20**	**16.80**	**16.40**	**19.20**
	Dos etapas: out-links y título	10.00	35.20	12.80	15.20	22.80	21.60
	Dos etapas: out-links y descripción	**10.80**	**37.20**	**14.80**	**15.60**	**17.60**	**18.80**
	Tres etapas: título, descripción y out-links	**9.60**	**40.80**	**12.40**	**17.60**	**15.60**	**19.20**
	Tres etapas: título, out-links y descripción	9.20	41.20	11.20	15.20	19.20	41.20
	Tres etapas: descripción, título y out-links	11.20	30.00	14.40	17.60	22.80	18.80
	Tres etapas: descripción, out-links y título	12.40	35.60	15.20	17.20	15.60	19.20
	Tres etapas: out-links, título y descripción	10.00	37.60	13.60	16.00	18.80	22.40
	Tres etapas: out-links, descripción y título	10.80	37.60	14.80	16.00	16.80	18.80

Tabla 3. Resultados de combinaciones de técnicas de clasificación.

Si bien para *Google* el porcentaje de páginas sin clasificar es bajo, para *Yahoo*! continúa siendo elevado. Por esto la técnica que se aplicó luego es, a partir de los resultados de las clasificaciones anteriores, tomar aquellas páginas que no clasificaron y efectuarles una tercera clasificación, por el método que aún no fue aplicado, teniéndose así seis nuevas clasificaciones, donde las etapas sucesivas de clasificaciones son: Titulo, Descripción y Enlaces salientes; Título, Enlaces salientes y Descripción, Descripción, Título y Enlaces salientes, Descripción, Enlaces salientes y Título; Enlaces salientes, Título y Descripción; y Enlaces salientes, Descripción y Título. Algunos de los valores obtenidos son buenos, dado que lograron aumentar la cantidad de páginas clasificadas correctamente, como es el caso de la clasificación dada por título, descripción y enlaces salientes (en ese orden) para Altavista.

De entre todas las clasificaciones en dos etapas, las mejores fueron las que combinaban

descripción con enlaces salientes. Con respecto a las clasificaciones en tres etapas, las mejores fueron las que empezaban con la clasificación por título seguida por las otras dos en ambos órdenes. Si bien se observaron algunas diferencias en los resultados de los diferentes métodos y para los distintos motores, en todos los casos presentados, las clasificaciones combinadas fueron muy superiores en cuanto a la precisión a las efectuadas individualmente y, si bien estas formas combinadas presentan más carga computacional, el tiempo de respuesta no varía en forma significativa con respecto a las formas individuales.

Se calcularon los indicadores I1 e I2 para todos los métodos combinados, para determinar en cuáles ambos indicadores (I1 e I2) poseen los valores más altos. Para este cálculo no se consideraron las categorías separadamente, puesto que interesa determinar, para cada método, qué tan bien se ubican las páginas en las cuatro categorías propuestas.

Clasificación	GOOGLE		ALTAVISTA		YAHOO	
	I1	I2	I1	I2	I1	I2
Dos etapas: título y out-links	0,71	0,81	0,67	0,77	0,49	0,57
Dos etapas: descripción y out-links	0,76	0,78	0,72	0,76	0,56	0,66
Dos etapas: título y descripción	0,68	0,83	0,49	0,70	0,59	0,87
Dos etapas: descripción y título	0,65	0,78	0,50	0,70	0,43	0,63
Dos etapas: out-links y título	0,63	0,72	0,61	0,70	0,57	0,66
Dos etapas: out-links y descripción	0,67	0,75	0,71	0,76	0,51	0,65
Tres etapas: título, descripción y out-links	0,66	0,72	**0,73**	**0,76**	**0,74**	**0,81**
Tres etapas: descripción, out-links y título	0,72	0,72	**0,73**	**0,76**	0,60	0,66
Tres etapas: título, out-links y descripción	0,78	0,77	**0,71**	**0,78**	0,49	0,56
Tres etapas: descripción, título y out-links	0,74	0,73	0,65	0,74	0,50	0,59
Tres etapas: out-links, descripción y título	0,75	0,75	0,72	0,76	0,55	0,65
Tres etapas: out-links, título y descripción	0,76	0,76	0,65	0,71	0,60	0,65
Resultado más frecuente	**0,81**	**0,81**	0,71	0,74	0,63	0,64
Vector combinado	**0,79**	**0,78**	0,71	0,75	**0,68**	**0,69**

Tabla 4. Valores de I1 e I2 para los motores *Google*, Altavista y *Yahoo*.

Como se observa en la Tabla 4, los valores más altos de I1 e I2 se presentan para *Google* y los más bajos para *Yahoo*!. Para *Google* la mejor performance está dada por el resultado más frecuente y por el vector combinado de título, descripción y out-links. Para Altavista, los mayores valores se dan para las clasificaciones en tres etapas: título/descripción/out-links, descripción/out-links/título y título/out-links/descripción. Para *Yahoo*! los mejores resultados se obtuvieron para la clasificación en tres etapas título/descripción/out-links y para el vector combinado de título, descripción y out-links.

5. Conclusiones

El análisis presentado en este trabajo puede ser aplicado para mejorar la búsqueda en la *web*. En los resultados de una búsqueda suelen aparecer páginas comerciales, proyectos, noticias, presentaciones, etc. Estas categorías pueden resultar muy útiles al usuario que usualmente está interesado en una sola de ellas. Las técnicas de clasificación existentes usadas en forma independiente no brindan buenas soluciones, ya que quedan muchas páginas sin clasificar. Se observó que la clasificación por títulos es la que deja la mayor cantidad de páginas sin clasificar. En cambio, la clasificación por enlaces salientes es la que clasifica la mayor cantidad de páginas; no obstante, es la que produce mayor porcentaje de error. Excepto para *Yahoo*!, la clasificación por títulos es la más precisa, es decir, es la que clasifica menor cantidad de páginas pero comete la menor cantidad de errores.

Se combinaron las técnicas buscando obtener mejores resultados. Las clasificaciones alcanzadas con el título, enlaces salientes y descripción en forma independiente, que habían dejado a gran parte de las páginas sin asignarle una categoría, fueron primeramente reunidas de forma que las tres aportaran su resultado para otorgar una única categoría a cada página,

tomándose la clasificación más frecuente. Así, la cantidad de páginas en la categoría *Otros* llegó a bajar entre 20 y 40%, con aproximadamente un 15% de error para *Google* y Altavista.

La información lograda a partir del título, descripción y enlaces salientes se combinó luego para constituir un único vector. Este método obtuvo buenos resultados en especial para Altavista y *Yahoo*!, donde se elevó considerablemente la cantidad de páginas categorizadas.

El segundo modo de combinar las tres clasificaciones individuales fue una clasificación en etapas sucesivas, donde a aquellas páginas que no caían en ninguna de las cuatro categorías se les aplicaba un segundo y hasta tercer método, quedando así, en la categoría *Otros* sólo aquellas páginas que ninguna de las tres formas de clasificación le habían otorgado una clase. De entre todas las clasificaciones en dos etapas, las mejores fueron las que combinaban descripción con enlaces salientes, y con respecto a las clasificaciones en tres etapas, las más altas fueron las que empezaban con la clasificación por título seguida por las otras dos en ambos órdenes.

Si bien se observaron algunas diferencias en los resultados de los diferentes métodos y para los distintos motores, en todos los casos presentados, las clasificaciones combinadas fueron muy superiores en cuanto a la precisión a las efectuadas individualmente y, si bien estas formas combinadas presentan más carga computacional, el tiempo de respuesta no varía en forma significativa con respecto a las formas individuales. Se observó que para *Google* y Altavista las clasificaciones fueron bastante precisas, teniéndose un pequeño margen de error. Las mayores diferencias de valores entre los obtenidos por las técnicas automáticas y la clasificación manual se dieron para *Yahoo*!, pero sólo para las categorías *Comercial* y *Publicaciones*, ya que para el resto de las categorías no se observaron diferencias de importancia.

Estos resultados muestran que, dependiendo del interés del usuario, se puede establecer un modo de determinar qué combinación de técnicas utilizar si se desea recuperar sólo una categoría de páginas, así como determinar qué buscador es el más adecuado en cada caso. En este trabajo, se usaron consultas donde sólo se consideraban los resultados en el idioma español. Las conclusiones obtenidas probablemente varíen si se hacen en otros idiomas, lo que sería valioso analizar. Para efectuar las clasificaciones se usaron cuatro categorías fijas. Una interesante cuestión es determinar cómo se adaptarían los algoritmos usados para la determinación de las clases en forma dinámica, de acuerdo a las características del grupo de documentos retornados por los distintos motores.

Referencias

Carlos Figuerola, Raquel Gómez, Ángel Rodríguez y José Luis Berrocal. 2001. Stemming in Spanish: A First Approach to its Impact on Information Retrieval. CLEF 01. *Workshop Cross-Language System Evaluation Campaign*. CLEF2. www.ercim.org/publication/ ws-proceedings/CLEF2/figuerola.pdf.

G. Huck, P. Fankhauser, K. Aberer y E. Neuhold. 1998. *JEDI: Extracting and Synthesizing Information from the Web*, COOPIS 98. IEEE Computer Society Press, New York.

J. Farkas. 1994. Generating Document Clusters Using Thesauri and Neural Networks. *Canadian Conference on Electrical and Computer Engineering*, Vol. 2: 710-713.

Marti A. Hearst. 1999. The use of categories and clusters in organizing retrieval results. En Tomek Strzalkowski (1999: 333-374).

R. Baeza-Yates y B. Ribeiro-Neto (eds.). 1999. *Modern Information Retrieval*. ACM Press, New York.

R. Motz, C. Deco, C. Bender, C. Manzino, L. Perlo, E. Ruiz, y A. von Furt. 2003. La clasificación en la carga de *web data warehouse*. En *II Workshop Chileno de Bases de Datos*, Jornadas Chilenas de Computación, Chillan, Chile.

Thorsten Joachims. 1998. Text Categorization with Support Vector Machines: Learning with Many Relevant Features. Machine Learning. *10th European Conference on Machine Learning*, 137-42.

Tom M. Mitchell. 1997. *Machine Learning.* McGraw-Hill.

W. Lam y K. Low. 1997. Automatic document classification based on probabilistic reasoning: Model and performance analysis. *Proc.IEEE International Conference on Systems*, Vol. 3: 2719-2723.

Walter Panessi y Fernando Bordignon. 2001. Procesamiento de Variantes Morfológicas en Búsquedas de Textos en Castellano. En *Revista Interamericana de Bibliotecología*, 24(1): 69-88.

Tomek Strzalkowski (ed.). 1999. *Natural Language Information Retrieval*. Kluwer Academic Publishers. www.sims.berkeley.edu/~hearst/papers/cats-and-clusters.pdf.

Yiming Yang y Xin Liu. 1999. A re-examination of text categorization methods. *22nd International Conference on Research and Development in Information Retrieval*, 42-49, Berkeley.

Capítulo 2

CONOCIMIENTO DE LA LENGUA Y TÉCNICAS ESTADÍSTICAS EN EL ANÁLISIS LINGÜÍSTICO

Gabriel G. Bés, Zulema Solana y Celina Beltrán

En Víctor M. Castel, Comp. (2005) *Desarrollo, implementación y utilización de modelos para el procesamiento automático de textos*. Mendoza: Editorial de la Facultad de Filosofía y Letras, UNCuyo: 13-22.
ISBN 987-575-010-0 del soporte Internet

Conocimiento de la lengua y técnicas estadísticas en el análisis lingüístico

Gabriel G. Bès
Universidad Blaise-Pascal
Groupe de Recherche dans les Industries de la Langue (GRIL)
Clermont-Fd., Francia
Gabriel.Bes@univ-bpclermont.fr

Zulema Solana
Universidad Nacional de Rosario
Facultad de Humanidades y Artes
Rosario, Argentina
zsolana@arnet.com.ar

Celina Beltrán
Universidad Nacional de Rosario/INDEC
Facultad de Ciencias Agrarias
Rosario, Argentina
beltranc@dat1.net.ar

Resumen

Son comparados los resultados obtenidos sobre un mismo corpus en la tarea del *POS tagging* por dos sistemas orientados por enfoques diferentes en lingüística computacional, el uno orientado por el Conocimiento de la lengua (sistema CL) y el otro por Técnicas estadísticas (sistema EST). Se trata de no limitarse a consideraciones globales sobre el « costo » de obtención de los dos tipos de resultados, noción mal definida, ni a cotejar resultados globales, sino de poner en relación los resultados obtenidos con las características lingüísticas involucradas. La problemática de la comparación es clarificada, los sistemas CL y EST presentados, la metodología de la comparación definida y los resultados obtenidos presentados. En el caso comparado, el sistema CL ofrece mejores resultados, pero la conclusión más interesante es la posibilidad de establecer correlaciones entre aspectos de la estructura lingüística y resultados obtenidos por técnicas estadísticas.

1 Clarificación del problema de la comparación

En la comunidad de estudiosos de las lenguas naturales, a pesar de su división en capillas y escuelas, se ha generalizado la conciencia de que el conocimiento de la lengua debe integrar, de una manera u otra, el trabajo sobre corpus efectivos. Esto es particularmente cierto entre aquellos que practican el tratamiento automático del lenguaje, campo en el cual no se puede continuar ignorando los escollos y problemas irritantes que nos desafían en los textos efectivos. La – mala – tradición chomskiana en sentido contrario está siendo así felizmente olvidada.

Pero los textos reunidos en corpora pueden ser utilizados de manera muy diferente. Una utilización, quizás mínima, es la de explotarlos para evaluar resultados de descripciones o de análisis o generaciones automáticos. Para la evaluación se usa cierto nivel de estadística utilizando contajes que finalmente darán medidas como las de precisión y cobertura.

No hacemos alusión a este tipo de estadística en el título de este trabajo. Se trata de un enfoque más profundo de la estadística que puede caracterizarse como sigue. Subyacente a un texto, existe un flujo de entidades que se caracterizan por sus frecuencias propias, adicionales y/o condicionales. Estas frecuencias o parámetros se pueden extraer inductivamente mediante algoritmos apropiados aplicados a corpus de entrenamiento (en general previamente etiquetados) y, una vez extraídas, a partir de ellas, mediante cálculos adicionales que utilizan operaciones algebraicas (por ejemplo funciones logarítmicas y/o de comparaciones vectoriales) se propone un sistema subyacente estadístico que se pretende válido para una categoría de textos. El sistema subyacente estadístico propuesto será aplicado a otros textos que el texto de entrenamiento, pero originados en la misma fuente (por ejemplo, textos de tal diario con tal tipo de noticias). Al ser aplicados de esta manera, se deben obtener los resultados esperados: desambiguación de categorías morfosintácticas, desambiguación de interpretaciones semánticas, identificación de colocaciones, etc. Las funcionalidades del sistema subyacente estadístico están evidentemente condicionadas por las informaciones estadísticas extraídas inductivamente de los textos de entrenamiento, convertidas en los parámetros utilizados por el sistema.

Un ejemplo bien caracterizado de utilización de estas técnicas estadísticas es la resolución de las ambigüedades de asignación categorial de las ocurrencias de expresiones en los textos (*POS tagging*).

Que se sepa, en todas las lenguas relativamente bien descriptas, existen expresiones que lexicalmente pueden ser asignadas a más de un categoría morfosintáctica, como *lo*, y *la* en español, clíticos o artículos, como *nota* en español, *note* en francés y *book* en inglés, verbos o nombres. Tal fenómeno no es ni aislado ni marginal. Es sistemático y caracteriza profundamente las lenguas naturales. Es prácticamente inexistente en los lenguajes formales de la lógica, la matemática y la informática. Pero todas las expresiones léxicas no son ambiguas, y se usan prácticamente siempre enunciados en donde la ambigüedad de asignación de las ocurrencias en los textos (*tokens*) pasa desapercibida en el uso. Si ello es así, es porque existen mecanismos en el lenguaje, no bien identificados hoy, que permiten resolver el escollo de la ambigüedad.

Existen al menos dos maneras de desambiguar las asignaciones categoriales: por técnicas estadísticas y por conocimiento de la lengua. Las primeras utilizan en general modelos de Markov, o modelos de Markov ocultos (HMM, *Hidden Markov Models*), los que van a observar transiciones en n-gramas entre categorías en un corpus de entrenamiento para deducir el sistema subyacente estadístico. El conocimiento de la lengua, en cambio, es utilizado cuando se ha logrado especificar un sistema subyacente expresado en reglas formales y explícitas, las cuales describen las representaciones que deben asignarse a los enunciados de una lengua, que hayan o no sido previamente observados en un corpus. Conocimiento de la lengua es una noción que se inscribe, pero sin las connotaciones mentalistas, en la noción chomskiana de gramática. Teniendo en cuenta ambas posiciones, Manning y Schutze (1999: 373) opinan así:

> The claim has been made that for somebody who is familiar with the methodology [fundada en el conocimiento de la lengua], writing this type of tagger takes no more effort than building an HMM tagger [...] though it could be argued that the methodology for HMM tagging is more easily accessible.

La última frase de la cita debería cambiarse por:

> La metodología HMM es más accesible para los informáticos, que en su gran mayoría ignoran la rica herencia formal y descriptiva de la lingüística, y la metodología fundada en el conocimiento de la lengua es más accesible para los lingüistas, que tardan en aceptar que las lenguas naturales no pueden ser hoy exploradas sin herramientas informáticas y sin exigencias de formalización.

El objetivo de la comunicación es comparar técnicas fundadas en el conocimiento de la lengua y técnicas estadísticas para dar solución al problema clave de la desambiguación de las asignaciones categoriales. La comparación se hará sobre los resultados efectivos obtenidos, sobre el mismo corpus, por dos sistemas diferentes, el uno orientado conocimiento de la lengua y el otro orientado técnica estadística, identificados respectivamente, de aquí en más, como *sistema CL* y *sistema EST*. Se utilizará un corpus francés, pero ambas técnicas pueden aplicarse a otras lenguas, y se aportarán datos sobre el español. La comparación no se limitará a cotejar cifras de resultados globales o parciales, pero tratará de comprender el funcionamiento de cada sistema, que permite o no programar su evolución controlada. Ninguno de los dos sistemas es ni puramente conocimiento de la lengua ni puramente estadístico. Este último no trabaja exclusivamente sobre las solas secuencias de caracteres, sino sobre secuencias de caracteres asociadas a categorías, y el sistema CL que se va a comparar, deja un residuo de casos para ser tratados por datos heurísticos, de tipo estadístico, pero que no necesitan de corpus de entrenamiento. Pero cada sistema da prioridades opuestas a cada tipo de información.

2 Etiquetado y extracción de los sintagmas núcleo

La noción de los sintagma núcleo – verbal, nominal, adjetival ... – retoma la noción de *chunk* y el análisis automático de sintagmas núcleo se inscribe en los análisis parciales de superficie o

shallow parsing. Si tenemos la expresión que sigue en (i) se la analiza como en (ii) en una sucesión de sintagmas núcleo (*snn*, nominal; *san*, adjetival; *svn*, verbal; *sadvn*, adverbial).

(i) Todas las noticias tan preocupantes no fueron recibidas muy rápidamente.
(ii) (Todas las noticias)snn (tan preocupantes)san (no fueron recibidas)svn (muy rápidamente)sadvn.

Quedará para otra etapa del análisis obtener el sn (sintagma nominal) completo a partir de la sucesión inicial snn + sadjn, caracterizar las dependencias en la oración para completar el análisis sintáctico y especificar la representación semántica, cf. (Aït-Mokhtar et al. 2002).

Los análisis automáticos de superficie no agotan entonces la temática del análisis automático de la oración, pero constituyen una parte importante, una especie de zócalo a partir del cual se presume que es posible completar el análisis oracional. Se inscriben en las exigencias del análisis robusto: se deben analizar textos efectivos, es decir, no sólo expresiones aisladas, y los resultados deben ser evaluables.

2.1 La problemática del sintagma verbal núcleo

En general, un sintagma núcleo se presenta como una sucesión de categorías bien identificables, con relaciones estrictas de linealidad entre ellas, de manera que es posible identificarlas en el análisis y determinar en dónde comienza y termina cada sintagma núcleo. Los que siguen son sintagmas verbales núcleo (en itálicas), tanto en francés como en sus traducciones en español.

(Il) *achète* (beaucoup).
(Él) *compra* (mucho).
(Il) *ne les a pas ac*hetées (hier).
(Él) *no las ha comprado* (ayer).
(Les fleurs) *ont été achet*és (hier).
(Las flores) *han sido compradas* (ayer).

Un svn en francés o en español, en estructuras activas o pasivas, puede comenzar por una forma verbal simple, o por la negación, o por el auxiliar, o por un clítico, y termina en una forma verbal flexiva o participial, o, en francés, por el clítico de nominativo. Para identificar los svn es necesario afrontar dos desafíos mayores:

- resolver las ambigüedades de categorización de las ocurrencias en los textos
- identificar las expresiones incisas incorporadas, encerradas o no entre comas

Ciñéndonos al francés (pero observaciones parecidas son válidas para el español) se observa que la ambigüedad de categorización es sistemática, tanto para las formas morfológicas como para las lexemáticas: *le, les, la, l'*, clíticos o artículos; *lui*, clítico o pronombre; *leur*, clítico o posesivo; *pas*, adverbio forclusivo o nombre, formas de *avoir* y *être* como auxiliares o verbos principales; formas verbales que también pueden ser sustantivos (*note, juge, empire*...); formas verbales que pueden ser participiales o personales como *interdit*, participio pasado o tercera persona del singular del presente de indicativo.

El otro desafío mayor, si se quiere satisfacer la exigencia de robustez del análisis, es decir, operar sobre textos efectivos, es ser capaz de identificar las expresiones incisas, abundantes en los textos, y que se presentan en una diversidad de formas (subrayadas en los ejemplos que siguen).

Il *est, sans le savoir, allé* dans la mauvaise direction.
Il *a, avec obstination, traité* ce problème.

2.2 Modelización, herramientas utilizadas y estrategias de análisis

Las Propiedades de Existencia y de Linealidad (Bès 1999) del Paradigma 5P, permiten describir las secuencias de categorías de los sintagmas verbales núcleo y caracterizar los puntos de inserción posibles de las incisas. Las ocurrencias en los textos son analizadas asociándolas a categorías, que serán subsumidas por las categorías especificadas en las secuencias definidas en 5P. El sistema CL de análisis automático (Bès et al. 2004) comporta dos módulos, con las herramientas siguientes:

- Smorph, (Aït-Mokhtar 1998), que tokeniza y efectúa un primer análisis morfológico, sin resolver las ambigüedades;
- Pasmo, (Paulo et al. 2001), para el español (Abbaci 1999), que desambigua e identifica las secuencias de categorías de los sintagmas verbales núcleo.

La evaluación de los resultados se hace mediante Censio[1] (Trouilleux 2005).

La estrategia de modelización de los recursos lingüísticos se sintetiza en los puntos siguientes:

- reducción drástica de las categorías léxicas
- no utilización de corpus etiquetados de entrenamiento
- explotación al máximo de las propiedades estructurales de la lengua, expresadas mediante reglas estructurales, dándoles prioridad sobre el residuo de datos estadísticos, tratados mediante reglas heurísticas.

La estrategia del análisis se resume a su vez en los puntos siguientes:

- tratar de hacer lo máximo con lo mínimo
- ir de lo seguro a lo inseguro
- utilizar la recursión para hacer idas y vueltas entre « piso parcial » y « techo parcial »

2.3 Resultados obtenidos

Para el francés[2] sobre un corpus de 10.400 palabras, los resultados obtenidos aparecen en la Tabla 1, en donde *N* : Número (cardinalidad), *N(svn)* : N de svn efectivos, *N1* : N de svn correctamente analizados, *N2* : N de expresiones analizadas como svn; y las fórmulas de Precisión: N1/N2 x 100 y de Cobertura: N1/N(svn) x 100.

N(sv)	N1	N2	Precisión	Cobertura
953	945	957	98,75	99,16

Tabla 1. Resultados sobre el corpus del francés.

Utilizando *Smorph* y MPS, en un sondeo (Solana y Bès 2005) sobre textos periodísticos en español (1.211 palabras), los resultados obtenidos son dados en la Tabla 2.

[1]Censio compara los resultados obtenidos por el sistema de análisis con los resultados especificados en el corpus de referencia, indicando, mediante tres etiquetas diferentes, si el resultado es correcto, incorrecto o inexistente, y permite diferenciar los resultados obtenidos por el sistema de análisis, lo que a su vez permite obtener evaluaciones « transparentes » en que se discriminan los resultados según las hipótesis utilizadas.

[2]Cf. en (Bès et al. 2004) un análisis detallado de los resultados obtenidos tanto sobre el sintagma verbal flexionado – el único aquí retenido – como sobre el sintagma verbal infinitivo, y la fórmula utilizada para determinar los límites del sistema.

N(sv)	N1	N2	Precisión	Cobertura
128	126	126	100	99,4

Tabla 2. Resultados sobre el corpus del español.

La utilización de Censio permite un análisis de los resultados, y, muy especialmente detectar aquellas expresiones analizadas erróneamente (en número de 12) que son imposibles de subsanar dentro del alcance de las hipótesis básicas del sistema CL: 3 de ellas se originan en la falta de poder expresivo para tratar las incisas verbales y 9 en malos resultados de las reglas heurísticas utilizadas.

2.4 Etiquetado estadístico del francés

En la página http://www.xrce.xerox.com/cgi-bon/mltt/demos/french.cgi de *Xerox Research Center* es posible acceder a un analizador morfológico y desambiguador de varias lenguas, entre las cuales el francés. Esta herramienta está fundada en la técnica estadística de los modelos de Markov ocultos o HMM (*Hiden Markov Models*); sobre los HMM en general cf. (Manning y Schütze 1999, capítulos 9 y 10), sobre el HMM utilizado por Xerox, cf. (Cutting et al. 1992), su discusión en (Chanod y Tapanainen 1995) y su utilización en un analizador oracional en (Aït-Mokhtar et al. 2002). En un modelo de Markov se conocen las transiciones entre categorías, en un modelo de Markov oculto se conocen las probabilidades de transición entre categorías. Un sistema de etiquetado de ocurrencias que utiliza la técnica de los HMM ha calculado las probabilidades de transición a partir de un corpus de entrenamiento, supervisado o no. Para nosotros, el analizador y desambiguador del francés consultado – que identificamos como sistema EST– actúa como una caja negra: se le someten expresiones en entrada y se observan las respuestas que da en salida. El problema es entonces comparar los resultados obtenidos por CL, orientado por Conocimiento de la Lengua, con EST, el etiquetador de Xerox, orientado por estadistica.

2.5 Comparación de los resultados

Los objetos en la salida de CL son formalmente muy diferentes a los objetos en la salida de EST. Los primeros se presentan simplemente impresos en negrita en el texto de salida, precedidos por la etiqueta que les habrá asociado Censio. En cambio la salida de EST es una tokenización que indica para cada ocurrencia (*token*) sus rasgos. Así a la entrada que sigue en (i), CL le asociará (ii) y EST le asociará (iii), aquí presentado en forma resumida.

i Marie ne la note pas souvent.
(María no la nota a menudo)
ii Marie **[C. vnfl-I ne la note pas]** souvent.
iii
Marie	+NOUN_INV
ne	+NEG
la	+PC
note	+VERB_P3SG
pas	+ADV
souvent	+ADV
.	+SENT

Los criterios utilizados para comparar los dos tipos de salida son los siguientes. Los resultados de salida en CL y EST son considerados equivalentes ssi:

- a partir de reglas generales lingüísticamente válidas se obtiene (iii) a partir de (ii)
- dado (ii) y la salida intermedia obtenida por Smorph sobre las ocurrencias de (ii), y a partir de reglas generales, se obtiene el resultado en (iii).

Notemos que en la salida (ii) se expresa en general: la expresión en negrita es la secuencia terminal asociable a un nudo *vnfl*. En el ejemplo anterior, a partir de las reglas sobre el sintagma verbal núcleo en francés, cf. (Bès 99), es posible asociar *vnfl* a la cadena *[ne, la, note, pas*]. Para obtener (iii) a partir de (ii) se debe recurrir a la salida de Smorph, en la que tendremos las informaciones siguientes aquí resumidas:

ne[negación] la[clítico ambiguo] note[verbo ambiguo] pas[forclusivo]

Las reglas generales nos dirán que un *[clítico ambiguo]* y un *[verbo ambiguo]* en una cadena a la que se le ha asociado el nudo *vnfl* deben ser considerados un clítico y un verbo no ambiguos. Podemos decir, en este tipo de casos, que los resultados de CL y EST son equivalentes.

Supongamos una entrada como la que sigue en (i) con las salida (ii) por CL y (iii) por EST.

i ... la note ...
ii ... la note ...
iii ... la +PC
note +VERB_P3SG

En este caso, los resultados en CL y en EST no son equivalentes. La ausencia de negrita en (ii) con la correspondiente etiqueta de Censio indica que CL no ha reconocido la expresión en (i) como siendo un *vnfl*, contrariamente a (iii), que etiqueta *la* y *note* como terminales de un *vnfl*. Es entonces imposible, en esta situación, pasar de (ii) a (iii) y viceversa. La discriminación entre resultados equivalentes y no equivalentes permite comparar los resultados obtenidos por ambos sistemas.

2.5.1 Metodología e implementación de la comparación

EST es consultable por Internet dentro de condiciones, perfectamente razonables, que hubieran hecho engorroso el análisis completo del corpus *test* analizado por CL. Se usa entonces una metodología de análisis selectivo que, sin embargo, debe dar pautas seguras de interpretación.

La evaluación discriminada de los resultados obtenidos por CL (cf. § 2.3) permite detectar 12 expresiones erradas que se discriminan en 3 originadas en carencias expresivas de reglas estructurales de CL, y 9 en carencias de reglas heurísticas, ambos tipos de carencias no solucionables en los límites del sistema. No hay otros errores en la salida de CL que los originados en estas 12 expresiones.

La metodología de análisis selectivo de EST consiste en responder a dos preguntas:

I ¿Cómo trata EST las 12 expresiones erradas en CL?
II ¿Cómo trata EST sub-conjuntos de expresiones correctamente analizadas por CL?

La implantación de la comparación para responder a (I) es directa: las 12 expresiones fueron sometidas a EST vía *Internet*. La implantación para responder a (II) requiere la utilización de la capacidad expresiva y declarativa de Censio, que permite discriminar entre tres tipos de etiquetas y detectar las expresiones que hubieran sido erradas de no mediar tal tipo de reglas (cf. § 2.2, nota 1).

2.5.2 Resultados obtenidos

Respuesta a la Pregunta I: EST trata correctamente 6 de las 12 expresiones tratadas incorrectamente en CL e incorrectamente las 6 restantes. Respuesta a la Pregunta II: sobre 98 expresiones testeadas en EST, todas tratadas correctamente por CL, EST trata incorrectamente 22. Resultados globales de expresiones erradas:

CL: 12
EST: ≥ 28

Es decir, EST multiplica en por lo menos 2.5 el número de expresiones erradas en CL (*por lo menos* ya que no todas las expresiones correctamente analizadas por CL fueron testeadas sobre EST). En la Tabla 3 que sigue más abajo se discrimina el origen de los errores en EST, que corresponden a la caracterización que sigue.

Línea	Caracterización
1	*vnfl* ambiguo con un nombre comienzo de oración
2	incisas delimitadas por comas
3	incisas no delimitadas por comas
4	*vnfl* ambiguo con un sintagma nominal de tipo *art + nombre*
5	*vnfl* (formas simples, ex. *compte*) no reconocidos como tales
6	incisas delimitadas por comas, también errores en CL
7	*vnfl* ambiguos, también errores en CL

Línea	N expresiones	Estructural	N errores	Precisión	Cobertura
1	4	+	2	1	1
2	2	+	2		2
3	23	+	3		3
4	27	-	1	1	
5	42	-	14		14
6	3	+	3		3
7	9	-	3	2	1

Tabla 3. Origen discriminado de los errores en EST.

Los resultados precedentes dan una radiografía en números de la comparación de los resultados obtenidos sobre un mismo corpus por dos sistemas inspirados en metodologías diferentes. Más allá de los números, parece importante señalar otras enseñanzas.

CL utiliza herramientas declarativas de análisis y una herramienta declarativa para la evaluación transparente de los resultados. Es entonces posible, por un lado, conocer los límites de CL y, por otro, desarrollarlo, modificando la especificación de sus fuentes sin tocar a la maquinaria algorítmica asociada. Por ejemplo, una de las reglas de CL permite desambiguar los *vnfl* que siguen inmediatamente a un pronombre nominativo como *il*, el que es declarado como *pnom* en Smorph. Se tiene así (de manera simplificada) la regla:

X[pnom] Y[vnfla] -->X[pnom] Y[vnfl]

En la regla, las mayúsculas *X, Y* notan variables y *vnfla* un *vnfl* potencialmente ambiguo. En un primer ciclo de aplicación de las reglas se especifican los *vnfl* estructuralmente no ambiguos y se los discrimina con respecto a los otros (es decir a los de tipo *vnfla*), los que pueden, en ciertos casos, ser desambiguados ulteriormente de manera contextual.

El pronombre *il* (*él*) en Smorph, es declarado como *pnom*. Tal no puede ser el caso de *elle* (*ella*), ya que no lo es siempre, cf. *avec elle* (*con ella*), pero *elle*, en posición inicial absoluta de oración sí lo es, de manera que *elle* puede ser especificado como *pnom*, por regla, en el contexto apropiado, y en ese contexto va a actuar como *il*.

Observemos también que un *pnom* debe también desambiguar un *vnfla* si entre ambos ocurre un adverbio, delimitado o no por comas (Cf. *Il, souvent, note les résultat* (*Él, a menudo, nota los resultados*). Todas estas extensiones son directas en CL y dan los resultados buscados.

Nada de todo esto ocurre en EST, caja negra no declarativa, la que, sometida a tests previamente seleccionados da los resultados siguientes (*ok:* resultado correcto; *k:* resultado incorrecto).

Test	Expresión	Resultado
i	Il/Elle *note/la note/en note* souvent.	ok
ii	Il/Elle souvent *note/la note/en note*.	k
iii	Il/Elle, souvent, *note/la note/en note.*	k
iv	avec elle *note/la note/en not souvent*	k

El testeo de EST por prueba y error muestra que el pronombre nominativo desambigua en el contexto inmediato (cf. (i), en donde se obtiene lo equivalente a *vnfl*), pero que el efecto se pierde cuando se pierde la transición inmediata, aunque la pérdida esté motivada por elementos como el adverbio que no perturban la acción desambiguante a distancia (cf. (ii) y (iii) en donde se obtienen sintagmas nominales para los *vnfla*). Pero el testeo muestra otro efecto indeseable: *elle* desambigua incorrectamente cuando no es *pnom*, cf. (iv), ya que en este caso se obtiene *vnfl* para las formas ambiguas, lo que puede o no ser cierto.

Una desambiguación estructural es una mini-proyección: mediante ella se está diciendo que satisfechas tales condiciones se obtienen (casi!) siempre tales resultados. Aumentar la proporción de resultados estructurales en detrimento de los resultados heurísticos consolida un sistema. Esto es posible en un sistema tal que CL pero no en SL.

La desambiguación de los *vnfla* potencialmente ambiguos con respecto a sintagmas nominales núcleos del tipo *art + nombre* tanto en CL (resultados calculados por heurística) como en EST (resultados calculados por transiciones) son satisfactorios; cf. la línea 4 de la Tabla 3. Parte de los resultados obtenidos por heurística en CL pueden transformarse en resultados estructurales implementando en reglas de Cl la restricción siguiente: *dos vnfl no se siguen inmediatamente.* En CL, obtenidos los *vnfl* estructurales seguros, la restricción precedente se implementa declarando en reglas: *un vnfla no puede seguir o preceder inmediatamente a un vnfl.* Gracias a la evaluación transparente de CL, se sabe que 74 sobre 75 ocurrencias de *vnfla* se desambiguan correctamente. Implementadas las restricciones de co-ocurrencia entre *vnfla* y *vnfl*, los casos resueltos heuristicamente se reducen a 57. Es imposible obtener este tipo de resultados en EST. Un testeo con expresiones como *Nous avec elle en note[vnfla] partirons-nous/partirions-nous[vnfl]?*, muestra que *en note* es analizado como *vnfl*, lo que es *k* (restricción no siempre válida).

3 Balance y perspectivas

Los resultados obtenidos favorecen al sistema orientado por el conocimiento de la lengua sobre el sistema orientado por técnicas estadísticas. Pero quizás la perspectiva más interesante abierta por la metodología de la comparación sea la de relacionar características lingüísticas con resultados obtenidos por técnicas estadísticas. Los sistema estadísticos, contrariamente a los fundados en el conocimiento de la lengua, no son pensados – al menos explícitamente – en función de rasgos específicos de una estructura lingüística, actúan como cajas negras, no declarativas, los resultados globales que se obtienen no permiten establecer relaciones con lo estructural de la lengua y es difícil imaginar para ellos un proceso controlado de desarrollo. Pero si se los interroga desde sistemas de tipo CL utilizando evaluaciones transparentes, parece posible llegar a determinar qué características lingüísticas se correlacionan con qué resultados obtenidos por técnicas estadísticas.[3]

[3]En el Taller sobre Límites en estas mismas jornadas JALIMI 2005 se presenta y utiliza un sistema de detección de límites oracionales fundado en la Técnica estadística de ME (Máxima Entropía), y se evalúan los resultados en función de la metodología estadística utilizada y según los resultados obtenidos, los que son analizados con la misma metodología que la utilizada en este trabajo.

Referencias

Faiza Abbaci. 1999. *Développement du Module Post-Smorph*. 1999. Memoria del DEA de Linguistique et Informatique. Universidad Blaise-Pascal/GRIL, Clermont-Fd.

Salah Aït-Mokhtar. 1998. *L'analyse présyntaxique en une seule étape*. Tesis de doctorado. Universidad Blaise-Pascal/GRIL, Clermont-Fd.

Salah Aït-Mokhtar, Jean-Pierre Chanod y Claude Roux. 2002. Robustness beyond shallowness: incremental deep parsing. En *Natural Language Engineering*, 8(2/3): 121-144.

Gabriel G. Bès, Lionel Lamadon y François Trouilleux. 2004. Verbal chunk extraction in French using limited resources. http://www.arxiv.org/pdf/cs.CL/0408060.

Gabriel G. Bès. 1999. La phrase verbale noyau. En *Recherches sur le français parlé*, 15: 273-358.

Jean-Pierre Chanod y Pasi Tapanainen. 1995. Tagging French – comparing a statistical and a constraint-based method. En *Proceedings of the Seventh Conference of the European Chapter of the Association for Computational Linguistics (EACL'95),* 149-156.

Doug Cutting, Julian Kupiec, Jan Pedersen y Penelope Sibun. 1992. A Practical Part-of-Speech Tagger. En *Third Conference on Applied Natural Language Processing*, Trento, 133-140.

Christopher. D. Manning y Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing*. Cambridge Mass., The MIT Press.

Joana L. Paulo, Nuno Mamede y Caroline Hagège. 2001. *PasMO – Pos-AnaliSe* MOrfologica. Technical report, L2F-INESC-ID, Lisboa.

Zulema Solana y Gabriel G. Bès. 2005. Extracción del sintagma verbal núcleo y resolución de ambigüedades en la asignación categorial. En *Revista de Letras*: 157-171, UNR / Facultad de Humanidades y Artes.

François Trouilleux. 2005. *Censio*. Rapport technique. Université Blaise-Pascal/GRIL, Clermont-Fd.

Capítulo 3

DETERMINACIÓN DE VALORES DE VERDAD DE CONDICIONES DE REGLAS DE GENERACIÓN DE TEXTOS

Víctor M. Castel

En Víctor M. Castel, Comp. (2005) *Desarrollo, implementación y utilización de modelos para el procesamiento automático de textos*. Mendoza: Editorial de la Facultad de Filosofía y Letras, UNCuyo: 23-33.
ISBN 987-575-019-0 del soporte Internet

Determinación dinámica de valores de verdad de condiciones de reglas de generación de textos

Víctor M. Castel

Consejo Nacional de Investigaciones Científicas y Técnicas
Universidad Nacional de Cuyo
Mendoza, Argentina
vcastel@lab.cricyt.edu.ar

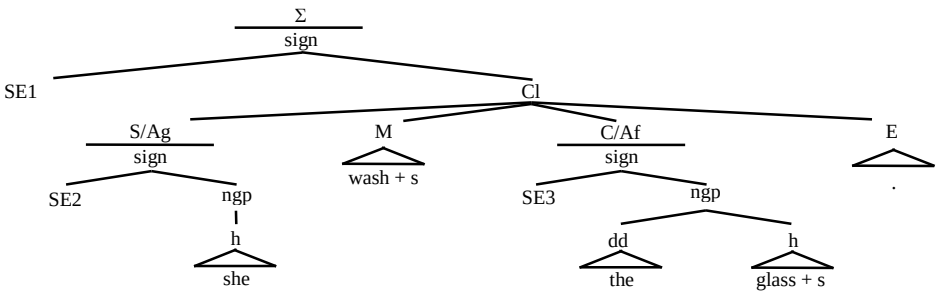
Resumen

La Léxico-Gramática de Cardiff (LGC; Fawcett 2000, Fawcett y otros 1993) es una gramática sistémica funcional orientada a la generación de textos en inglés. Las reglas semánticas y sintácticas de la LGC son implicaciones materiales, $p \rightarrow q$, en las que la condición p puede ser un rasgo, por ejemplo, *congruent situation* $\rightarrow q$, o una disyunción de rasgos, por ejemplo, *giver* $\vee$ *seeker* $\rightarrow q$, o una conjunción de rasgos, por ejemplo, *information* $\wedge \neg$ *future_trp* $\rightarrow q$. Los términos de las condiciones disyuntivas pueden ser también conjunciones, y los términos de las condiciones conjuntivas pueden ser también disyunciones. Algunas de las reglas de la LGC son muy complejas porque (i) contienen condiciones con términos que son conjunciones con términos que son disyunciones, y (b) tanto los términos como las condiciones pueden estar negados. Las reglas de la LGC realizan las operaciones definidas en q (inserción de rasgos semánticos, composición, llenado, exponencia, etc.), si la condición p es verdadera en relación con una representación semántica dada. Este trabajo presenta un procedimiento computacional que permite determinar el valor de verdad de p, verdadero o falso, de manera dinámica a partir de los rasgos semánticos elegidos en el proceso de generación, y del subconjunto relevante de ausencias de rasgos que resulta de esas elecciones.

1 Introducción

La LGC es una gramática orientada a la generación de textos capaz de definir representaciones lingüísticas del tipo (1) para oraciones como (2):

(1)



(2) She washes the glasses.

Abreviaturas: Σ = variable que abarca categorías de género; Cl = clause; S = subject; Ag = agent; ngp = nominal group; h = head; M = main verb; C = complement; Af = affected; dd = deictic determiner; E = ender; SE = selection expression; SE1 = [entity, situation, ..., present trp, ..., washing, ..., outsider sth, count sth, singular sth, ...]; SE2 = [entity, thing, outsider, recoverable thing, ..., singular tc, human tc, female tc, ...]; SE3 = [entity, thing, ..., outsider, ..., artefact, container, glass c, count cc, plural cc, ...], donde *situation*, *thing*, etc. son rasgos semánticos.

Las reglas semánticas y sintácticas involucradas en la generación de (1-2) son todas implicaciones que pueden representarse, leerse e interpretarse como en (3i-iii), respectivamente:

(3i) $p \rightarrow q$
(3ii) si *p*, entonces *q*
(3iii) si *p* is verdadera, entonces ejecute *q*,

donde *p* y *q* son variables que abarcan condiciones y consecuencias, respectivamente. Sirvan los ejemplos de (4) para ilustrar valores posibles de la condición *p*:

(4i) r1 $\rightarrow q$
(4ii) r2 / r3 / r4 $\rightarrow q$
(4iii) r5 / r6 / (r7 & r8) $\rightarrow q$
(4iv) r9 & r10 & r11 $\rightarrow q$
(4v) r12 & (r13 / r14) & r15 $\rightarrow q$
(4vi) r16 / r17 / (r18 & not r19) $\rightarrow q$
(4vii) r20 & not (r21 / r22 / r23) $\rightarrow q$
(4viii) not (r24 / r25) $\rightarrow q$,

donde "r1", "r2", etc. son rasgos semánticos, y "/" y "&" son los símbolos utilizados por la LGC para representar los operadores lógicos "$\vee$" y "$\wedge$", respectivamente. La condición *p* puede ser simple, como en (4i), disyuntiva, como en (4ii-iii, vi, viii) o conjuntiva, como en (4iv-v, vii). Observemos que las condiciones disyuntivas y conjuntivas pueden contener términos que son conjunciones y disyunciones, respectivamente. Así, por ejemplo, la condición disyuntiva (4iii) contiene el término "(r7 & r8)", que es una conjunción de términos. Por su parte, la condición conjuntiva (4v) contiene el término "(r13 / r14)", que es una disyunción de términos. Los ejemplos (4vi-viii) nos muestran que tanto los términos como las condiciones pueden estar negados. En la LGC, existen reglas sintácticas, pero no reglas semánticas, con términos o condiciones negados.

Razones de espacio nos impiden desplegar las ideas fundamentales sobre los valores posibles de la consecuencia *q* en la LGC. Baste decir que, en las reglas semánticas, la consecuencia *q* es la operación que permite introducir rasgos semánticos en la representación lingüística de modo de obtener representaciones semánticas del tipo ilustrado por SE1-SE3 en (1). En las reglas sintácticas, la consecuencia *q* puede ser una operación de llenado, composición o exposición, entre otras operaciones posibles; así, por ejemplo, en (1), las reglas sintácticas han permitido que tres ocurrencias de la unidad *sign* llenen los elementos *Σ*, *S/Ag*, y *C/Af*; que los elementos *dd* y *h* compongan la unidad *ngp*; y que los items *she*, *the*, y "." expongan los elementos *h*, *dd* y *E*, respectivamente.

Lo esencial para los fines de este trabajo es mostrar cómo se determina el valor de verdad, verdadero o falso, de la condición *p*. Para ello, presentamos a continuación los aspectos centrales de un algoritmo que permite hacer esa determinación en el contexto de la LGC.

2 Algoritmo para la determinación del valor de verdad de condiciones de reglas

Esta sección está dedicada a presentar de manera informal las propiedades distintivas de los procedimientos computacionales involucrados en la determinación del valor de verdad de las condiciones de las reglas semánticas y sintácticas de la LGC.

2.1 Representaciones semánticas y condiciones de reglas

El efecto conjunto de las reglas semánticas y sintácticas es una representación lingüística del tipo ilustrado en (1). La tarea específica de las reglas semánticas es construir representaciones semánticas como SE1, SE2 y SE3. La tarea específica de las reglas sintácticas es construir uni-

dades sintácticas del tipo *Cl*, *ngp*, etc. La(s) operación(es) definida(s) en la consecuencia *q* se ejecuta(n) si la condición *p* es verdadera con respecto a una representación semántica. La representación lingüística (1) contiene las abreviaturas SE1, SE2 y SE3 de las representaciones semánticas asociadas a la unidad *Cl*, la unidad *ngp* sujeto y la unidad *ngp* complemento.

Lo fundamental a los fines de este trabajo es que el valor de verdad de la condición *p* se determina sobre la base de representaciones semánticas, es decir, conjuntos de rasgos semánticos como los ejemplificados en (5):[1]

(5i) [r1, r3, r9, r10, r11, r12, r28, r31]
(5ii) [r7, r8, r329, r74]
(5iii) [r12, r15, r77, r128]
(5iv) [r18, r523, r276]
(5v) [r20, r48, r134]
(5vi) [r20, r21, r134]

Observemos que algunos de los rasgos semánticos de las representaciones en (5) son idénticos a términos de las condiciones de las reglas en (4), y que otros no lo son. Esta distinción es crucial para determinar el valor de verdad de las condiciones de las reglas en (4). Veamos algunos casos.

Diremos, por ejemplo, que las condiciones de las reglas (4i-ii; iv) son verdaderas con respecto a (5i), y falsas con respecto a (5ii-vi). Veamos por qué. La condición de (4i) es verdadera con respecto a la representación (5i) porque consta de un término, "r1", que corresponde a un rasgo semántico presente en la representación (5i) que es idéntico a ese término. Por su parte, la condición disyuntiva de (4ii) es verdadera con respecto a la representación (5i) porque uno de sus términos, "r3", es idéntico a un rasgo semántico presente en (5i). En cuanto a la condición conjuntiva de (4iv), diremos que es verdadera con respecto a (5i) porque para cada uno de sus términos, "r9", "r10" y "r11", hay en (5i) un rasgo semántico idéntico a él. La condición de (4i) es falsa con respecto a (5ii-vi) porque en ninguna de estas representaciones semánticas hay un rasgo que sea idéntico al término "r1". La condición disyuntiva de (4ii) es falsa con respecto a (5ii-vi) porque en ninguna de estas representaciones hay un rasgo que sea idéntico a alguno de los términos de la condición disyuntiva. La condición conjuntiva de (4iv) es falsa con respecto a (5ii-vi) porque ninguna de estas representaciones contiene tres rasgos semánticos que sean idénticos a los términos de la condición conjuntiva. La condición diyuntiva de (4iii) es verdadera con respecto a la representación semántica (5ii) porque uno de sus términos, la conjunción "r7 & r8", es verdadero con respecto a (5ii); en efecto, la conjunción "r7 & r8" es verdadera con respecto a (5ii) porque para cada uno de sus términos hay un rasgo en (5ii) que es idéntico a él. La condición de (4vii) es verdadera con respecto a (5v) y falsa con respecto a todas las demas representaciones semánticas de (5). La condición conjuntiva de (4vii) es verdadera con respecto a (5v) porque sus dos términos los son. Por un lado, el término "r20" es idéntico a un rasgo semántico de (5v). Por otro lado, el término disyuntivo "not (r21 / r22 / r23)" es verdadero porque (i) la disyunción "(r21 / r22 / r23)" es falsa con respecto a (5v), pero (ii) al estar negada se "vuelve" verdadera; la intuición es que es verdad que no hay ningún rasgo semántico en (5v) que sea idéntico a alguno de los términos de la disyunción "(r21 / r22 / r23)". Dejamos como ejercicio para el lector la determinación del valor de verdad del resto de las condiciones en relación con las representaciones semánticas de (5).

[1]Usamos ejemplos abstractos porque en la LGC los rasgos semánticos son largos y las reglas completas son muy complejas, lo cual atenta contra el espacio disponible. Lo importante, sin embargo, es que la discusión presentada y los procedimientos definidos valen independientemente de la extensión y complejidad de reglas reales.

2.2 Objetivo del algoritmo de determinación de valores de verdad

El objetivo del algoritmo es determinar el valor de verdad, verdadero o falso, de las condiciones de reglas semánticas y sintácticas de la LGC. Esta determinación puede hacerse de manera dinámica en el proceso de generación, a saber: a medida que se va construyendo la representación semántica. El algoritmo se basa en las elecciones de rasgos semánticos realizadas por el usuario del generador o por el generador mismo, lo que nos permite distinguir dos tipos de rasgos: presentes y ausentes. Un rasgo semántico presente es un rasgo que ha sido elegido en el proceso de generación de una representación semántica y, por tanto, es parte de esa representación semántica. Un rasgo ausente es todo rasgo que no ha sido elegido en el proceso de generación y, por tanto, no es parte de la representación semántica.

2.3 Determinación de valores de verdad por presencia de rasgos

El algoritmo utiliza la presencia de rasgos para asignar valores de verdad a las condiciones de reglas semánticas y sintácticas. Esta asignación se hace a medida que se eligen los rasgos en el proceso de generación de la representación semántica. La idea básica es determinar el impacto que tiene la elección de un rasgo sobre el valor de verdad de todas las condiciones en las que el rasgo elegido es idéntico a uno de los términos que la componen.

El algoritmo de determinación dinámica de valores de verdad a partir de rasgos elegidos se funda en la observación de que, al no existir llamado explícito de reglas semánticas, es, en principio, ineficiente evaluar las condiciones de **todas** las reglas semánticas, ya que la mayoría de ellas resultarán falsas. Pero, dado que las condiciones de las reglas semánticas no contienen términos ni condiciones negados, es posible calcular la verdad de esas condiciones composicionalmente a partir de los rasgos elegidos en el proceso de generacion. Se evita así evaluar las condiciones de todas las reglas semánticas, ya que al predeterminar cuáles son verdaderas es innecesario manipular las falsas.

El rasgo elegido puede ser idéntico o no al término de una condición simple y/o una condición conjuntiva y/o una condición disyuntiva. Puesto que tanto los términos como las condiciones de regla pueden estar negados, el impacto del valor de verdad de un término sobre las condiciones de la que es parte depende de tres factores: (i) el valor de verdad "inherente" del término, (ii) el carácter negado o no del término, y (iii) el carácter negado o no de la condición de la que es parte el término. Los tres factores permiten definir una función como la siguiente, capaz de devolver el valor de verdad aportado por un término a la condición de la que es parte:

TermTV (TNeg, CNeg, InTV),

donde *TNeg = TermNegation*, *Cneg = ConditionNegation* e *InTV = InherentTruthValue*. Los valores posibles de las variables *Tneg* y *Cneg* son "sí" y "no". Los valores posibles de *InTV* son *Verdadero* y *Falso*. Los valores devueltos por la función son *Verdadero* o *Falso*, según resulte el cálculo. Véase en el Anexo el procedimiento que implementa la función.

Los procedimientos involucrados en la determinación de valores de verdad de condiciones a partir de la presencia de rasgos son los siguientes:

EvaluatePresentTermTVImpactOnConjunction strFeature, InTV
EvaluatePresentTermTVImpactOnDisjunction strFeature, InTV
EvaluatePresentTermTVImpacToSimpleCondition strFeature, InTV

El valor inicial de la variable *strFeature*, es decir, el valor con el que se inicia la aplicación de estos tres procedimientos, es siempre el rasgo elegido. Los valores sucesivos de la variable *strFeature* en el ciclo de aplicación basado en ese valor inicial son términos complejos, a saber:

disyunciones en el procedimiento *EvaluatePresentTermTVImpactOnConjunction* y conjunciones en en el procedimiento *EvaluatePresentTermTVImpactOnDisjunction*.

Supondremos que el valor inicial de la variable *InTV*, esto es, el valor con el que se inicia la aplicación de estos tres procedimientos, es *inherentemente verdadero*, en el sentido de que es verdad que ese término que es idéntico al rasgo elegido es parte de la representación semántica en construcción. Los valores sucesivos de *InTV* en el ciclo de aplicación de los procedimientos de determinación por presencia serán *verdadero* o *falso* según corresponda de acuerdo con el valor de verdad que asigne la función *TermTV (Tneg, Cneg, InTV)* a los términos conjuntivos y disyuntivos. Veamos ahora cada uno de estos procedimientos por separado.

2.3.1 EvaluatePresentTermTVImpactOnConjunction (strFeature, InTV)

El procedimiento busca en la tabla correspondiente todas los condiciones conjuntivas de reglas semánticas y sintácticas en las que *strFeature* es un término. Observemos que la LGC puede contener muchas condiciones conjuntivas diferentes con términos idénticos a *strFeature*.

Los valores de *strFeature* pueden ser un término simple o un término disyuntivo. Un término simple es un rasgo semántico. Un término disyuntivo es una condición disyuntiva. Los valores de *InTV*, verdadero o falso, se utilizan para determinar el aporte en valor de verdad del término *strFeature* al valor de verdad de la condición conjuntiva de la que es parte. Este aporte se calcula mediante la función *TermTV (TNeg, CNeg, InTV)*, que devuelve verdadero o falso, según corresponda (véase el Anexo).

El procedimiento se activa siempre a partir de la elección de un rasgo semántico por parte del usuario del generador o por el generador mismo. Así, el valor inicial de *strFeature* es un término simple y el valor inicial de *InTV* es verdadero si aceptamos la intuición básica expuesta en §2.3 de que es verdad que el rasgo elegido forma parte de la representación semántica.

En las sucesivas aplicaciones del procedimiento que sean consecuencia del ciclo de aplicación activado por el rasgo elegido, el valor de *strFeature* será siempre una término disyuntivo pasado mediante la variable correspondiente de este mismo procedimiento llamado en *DetermineCondTV* (véase en el Anexo el procedimiento implementado); por su parte, el valor de *InTV* de ese término disyuntivo será verdadero o falso según se calcule también al interior de *DetermineCondTV*.

2.3.2 EvaluatePresentTermTVImpactOnDisjunction (strFeature, InTV)

El procedimiento busca en la tabla correspondiente todas los condiciones disyuntivas de reglas semánticas y sintácticas en las que *strFeature* es un término. La LGC puede contener muchas condiciones disyuntivas diferentes con términos idénticos a *strFeature*.

Los valores de *strFeature* pueden ser un término simple o un término conjuntivo. Un término conjuntivo es una condición conjuntiva. Los valores de *InTV*, verdadero o falso, se utilizan para determinar el aporte en valor de verdad del término *strFeature* al valor de verdad de la condición disyuntiva de la que es parte. Este aporte se calcula mediante la función *TermTV (TNeg, CNeg, InTV)*, que devuelve verdadero o falso, según corresponda (véase el Anexo).

El procedimiento se activa siempre a partir de la elección de un rasgo semántico. Así, el valor inicial de *strFeature* es un término simple y el valor inicial de *InTV* es verdadero si aceptamos la idea básica expuesta en 2.3 de que es verdad que el rasgo elegido forma parte de la representación semántica.

En las sucesivas aplicaciones del procedimiento que sean consecuencia del ciclo de aplicación activado por el rasgo elegido, el valor de *strFeature* será siempre un término conjuntivo pasado mediante la variable correspondiente de este mismo procedimiento llamado en *DetermineCondTV* (véase el Anexo); por su parte, el valor de *InTV* de ese término conjuntivo será verdadero o falso según se calcule también al interior de *DetermineCondTV*.

2.3.3 EvaluatePresentTermTVImpacToSimpleCondition strFeature, InTV

Este procedimiento es muy sencillo y podemos, dadas las limitaciones de espacio, ignorarlo.

2.4 Determinación de valores de verdad por ausencia de rasgos

El algoritmo utiliza la ausencia de rasgos para asignar valores de verdad a las condiciones de reglas sintácticas explicitamente llamadas para aplicación por el generador. La restricción de la determinación por ausencia a sólo reglas sintácticas llamadas explicitamente obedece a la necesidad de no evaluar innecesariamente el gran número de reglas sintácticas de la LGC que no son relevantes en relación con una representación semántica determinada. Es claramente más eficiente un procedimiento de determinación que se aplica solamente a reglas llamadas explicitamente que uno que se aplica a todas las reglas sintácticas de la LGC.

La determinación de valores de verdad por ausencia es necesaria para las condiciones de reglas sintácticas, pero no para las condiciones de reglas semánticas. Si bien tanto las operaciones de las reglas sintácticas como las operaciones de las reglas semánticas se ejecutan cuando las condiciones asociadas son verdaderas, hay dos razones esenciales para esta distinción en el alcance de la determinación por ausencia.

En primer lugar, las condiciones de reglas sintácticas, pero no las condiciones de reglas semánticas, pueden estar negadas y/o pueden contener términos negados. Un término ausente negado puede hacer que una condición de regla sintáctica sea verdadera y por tanto aplicable en el proceso de generación. Al carecer de términos y condiciones negados, las condiciones de reglas semánticas nunca pueden resultar verdaderas sobre la base de la ausencia de rasgos en la representación semántica en construcción. Toda condición de regla semántica que contenga términos idénticos a rasgos ausentes de la representación semántica es falsa. Este carácter de condición falsa se sigue, digamos, por omisión: las condiciones de reglas semánticas no manipuladas por la determinación de valores de verdad por presencia son todas falsas; nuestro procedimiento de determinación por presencia simplemente no hace nada con ellas, y no es necesario aplicarles el procedimiento de determinación por ausencia.

En segundo lugar, las reglas sintácticas, pero no las reglas semánticas, pueden ser de la forma siguiente:

(6)
cond1 → oper1, *Else* oper2

En una regla sintáctica como (6), si la condición *cond1* es verdadera en relación con una representación semántica determinada, entonces debe llevarse a cabo la operación *oper1*; pero si la condición *cond1* es falsa en relación con una representación semántica determinada, entonces es verdadera la condición *Else* y, por tanto, debe llevarse a cabo la operación *oper2*. Luego, está claro que es imprescindible definir explicitamente también las condiciones falsas de las reglas sintácticas para que la condición *Else* funcione apropiadamente.

Un término de una condición simple, disyuntiva o conjuntiva de una regla sintáctica llamada explicitamente puede ser idéntico a un rasgo ausente de la representación semántica construida, esto es, a un rasgo que no ha sido evaluado por los procedimientos de determinación por presencia. El objetivo del algoritmo de determinación de valores de verdad por ausencia es precisamente determinar el impacto que tiene un término ausente sobre el valor de verdad de todas las condiciones de la que es parte. Puesto que tanto los términos ausentes como las condiciones de las que son parte pueden estar negados, el impacto del valor de verdad de un término ausente sobre las condiciones de la que es parte depende también de los tres factores señalados arriba en relación con la determinación por presencia: (i) el valor de verdad "inherente" del término ausente, (ii) el carácter negado o no del término ausente, y (iii) el carácter negado o no de la condición de la que es parte el término ausente. De modo entonces que podemos utilizar la misma

función *TermTV (TNeg, CNeg, InTV)* definida arriba (véase el Anexo). La diferencia radica en que ahora la idea básica es que el término ausente es *inherentemente falso* y, por tanto, el valor inicial de *InTV* es también *Falso*, en el sentido de que es falso que haya un rasgo en la representación semántica construida que sea idéntico al término.

Los procedimientos utilizados para la determinación de valores de verdad por ausencia de rasgos en la representación semántica son los siguientes:

EvaluateAbsentTermTVImpactOnConjunction Cond
EvaluateAbsentTermTVImpactOnDisjunction Cond
EvaluateAbsentTermTVImpactOnSimpleCondition Cond

2.4.1 EvaluateAbsentTermTVImpactOnConjunction (Cond)

El valor de *Cond* es una condición conjuntiva de una regla sintáctica llamada explicitamente por el generador. La tarea del procedimiento es determinar la contribución al valor de verdad de la condición *Cond* por parte de cada uno de los términos que la componen. Un término de una condición conjuntiva puede ser simple o disyuntivo. Un término simple ausente es un rasgo semántico que no ha sido elegido en el proceso de generación de la representación semántica (y por tanto no ha sido evaluado por el procedimiento de determinación de valores de verdad por presencia). Los términos simples ausentes y los términos disyuntivos son evaluados al interior del procedimiento mediante el llamado a los procedimientos *DetermineCondTV* y *EvaluateAbsentTermTVImpactOnDisjunction*, respectivamente:

DetermineCondTV Cond, "conjunctive", TermTV (TNeg, CNeg, False), Term, Cneg
EvaluateAbsentTermTVImpactOnDisjunction Term

2.4.1.1 DetermineCondTV (Cond, CondType, TruthValue, Term, CondNeg)

Este procedimiento se ocupa de determinar el valor de verdad de la condición conjuntiva *Cond* sobre la base del valor de verdad aportado por cada uno de los términos simples ausentes que la componen. El valor aportado por cada término simple ausente lo provee la función *TermTV (TNeg, CNeg, Falso)*. Como se puede apreciar en el Anexo, de manera general, es suficiente que uno de los términos de la condición conjuntiva *Cond* aporte el valor *Falso* para que la *Cond* sea Falsa. Por otra parte, *Cond* es verdadera si todos sus términos aportan el valor *Verdadero*.

2.4.1.2 EvaluateAbsentTermTVImpactOnDisjunction Term

Term es una variable que abarca los términos disyuntivos de la condición conjuntiva *Cond*. El objetivo de este procedimiento se define abajo en §2.4.2. Digamos aquí simplemente que por tratarse de un término que es una condición disyuntiva es necesario acceder a los términos que componen dicha condición disyuntiva para determinar su aporte al valor de verdad de la misma, de manera que una vez que éste haya sido determinado pueda utilizarse a su vez para determinar su aporte al valor de verdad de la condición conjuntiva *Cond*.

2.4.2 EvaluateAbsentTermTVImpactOnDisjunction (Cond)

El valor de *Cond* es una condición disyuntiva de una regla sintáctica llamada explicitamente por el generador. La tarea del procedimiento es determinar la contribución al valor de verdad de la condición *Cond* por parte de cada uno de los términos que la componen. Un término de una condición disyuntiva puede ser simple o conjuntivo. Un término simple ausente es un rasgo semántico que no ha sido elegido en el proceso de generación de la representación semántica (y por tanto no ha sido evaluado por el procedimiento de determinación de valores de verdad por presencia). Los términos simples ausentes y los términos conjuntivos son evaluados al interior

del procedimiento mediante el llamado a los procedimientos *DetermineCondTV* y *EvaluateAbsentTermTVImpactOnConjunction*, respectivamente:

DetermineCondTV Cond, "disjunctive", TermTV (TNeg, CNeg, False), Term, Cneg
EvaluateAbsentTermTVImpactOnConjunction Term

2.4.2.1 DetermineCondTV(Cond, CondType, TruthValue, Term, CondNeg)

Este procedimiento se ocupa de determinar el valor de verdad de la condición disyuntiva *Cond* sobre la base del valor de verdad aportado por cada uno de los términos simples ausentes que la componen. El valor aportado por cada término simple ausente lo provee la función *TermTV (TNeg, CNeg, False)*. De acuerdo con la implementación (véase el Anexo), si el valor de *TruthValue* es *Verdadero* y la condición disyuntiva no está negada ("a/b/c"), la condición *Cond* resulta verdadera; la determinación concluye, ya que incluso si los demás términos fueran falsos, la *Cond* seguiría siendo verdadera. Una vez que una condición disyuntiva no negada ha recibido una asignación verdadera, la condición permanece como tal, esto es, un término falso no puede volverla falsa.

Si el valor de *TruthValue* es verdadero y la condición disyuntiva *Cond* está negada ("not (a/b/c)"), debe agregarse 1 al campo que almacena la cantidad de valores verdaderos de la condición disyuntiva *Cond*. La condición disyuntiva *Cond* es verdadera si todos los términos disyuntivos que la componen aportan el valor verdadero.

Si el valor de *TruthValue* es falso y la condición disyuntiva *Cond* no está negada ("a/b/c"), debe asignarse el valor 0 al campo que almacena la cantidad de valores verdaderos de la condición disyuntiva *Cond*. Esto garantiza que el número de términos de *Cond* nunca será idéntico al número de elementos verdaderos y en consecuencia la condición *Cond* resulta falsa o bien es evaluada nuevamente por el código precedente, en cuyo caso *Cond* resulta verdadera.

Si el valor de *TruthValue* es falso y la condición disyuntiva *Cond* está negada ("not (a/b/c)"), *Cond* resulta falsa. La evaluación puede concluir aquí, ya que incluso si los demás términos de la disyunción *Cond* son verdaderos, la condición *Cond* seguirá siendo falsa; una vez que una disyunción negada ha sido evaluado como falsa, permance como tal, esto es, un término verdadero no puede volverla verdadera. Dicho de otra manera, es suficiente que un término pase el valor falso a una condición disyuntiva para que la condición sea falsa.

2.4.2.2 EvaluateAbsentTermTVImpactOnConjunction Term

Term es una variable que abarca los términos conjuntivos de la condición disyuntiva *Cond*. El objetivo de este procedimiento se definió arriba en §2.4.1. Agreguemos aquí que por tratarse de un término que es una condición conjuntiva es necesario acceder a los términos que componen dicha condición conjuntiva para determinar su aporte al valor de verdad de la misma, de manera que una vez que éste haya sido determinado pueda utilizarse a su vez para determinar su aporte al valor de verdad de la condición disyuntiva *Cond*.

2.4.3 EvaluateAbsentTermTVImpactOnSimpleCondition (Cond)

Por razones de espacio ignoramos este procedimiento que, de todos modos, es muy sencillo.

3 Conclusión y perspectivas

El trabajo ha presentado los aspectos esenciales de un algoritmo de determinación de valores de verdad de condiciones de reglas semánticas y sintácticas de la LGC. La propiedad distintiva del algoritmo es la manera dinámica en que se lleva a cabo esa determinación: a medida que el usuario del generador o el generador mismo elige un rasgo semántico en el proceso de construcción de la representación semántica, se evalúa toda condición de regla semántica y sintáctica que tenga un término idéntico al rasgo elegido. Los rasgos ausentes, esto es, los no elegidos, son

también relevantes para la determinación del valor de verdad de las reglas sintácticas. El objetivo es establecer el aporte en valor de verdad del término, presente o ausente, al valor de verdad de la condición de la que es parte, de manera que finalmente, en el caso de condiciones disyuntivas y conjuntivas, se pueda determinar "composicionalmente" el valor de verdad de la condición. El algoritmo presentado es una simplificación conceptual de los procedimientos que hemos implementado en Visual Basic 6.0 para la versión *mini* de la LGC.

En Castel y Diblasi (en preparación) se compara la eficiencia de este algoritmo con otro cuya determinación de los valores de verdad se hace de manera no-dinámica, a saber: (i) para cada regla semántica de condición disyuntiva o conjuntiva se debe determinar el carácter verdadero o falso de la misma en relación con la representación semántica en construcción (para lo cual es necesario evaluar todos los términos que componen la condición), y (ii) para cada regla sintáctica llamada se debe determinar el valor de verdad de su(s) condición(es) en relación con la representación semántica construida (para lo cual también es necesario evaluar todos los términos que componen las condiciones).

Referencias

Víctor M. Castel y Ángela Diblasi. En preparación. Algoritmos alternativos para la determinación de valores de verdad de condiciones de reglas en una gramática sistémica funcional: del rasgo a la condición vs. de la condición al rasgo. InCiHuSA, CONICET, Mendoza.

Robin P. Fawcett. 2000. *A theory of syntax for systemic functional linguistics*. John Benjamins, Amsterdam.

Robin P. Fawcett, Gordon H. Tucker, y Yuen Q. Lin. 1993. How a systemic functional grammar works: The role of realization in realization. In Horacek y Zock (1993: 114-86).

Helmut Horacek y Michael Zock. Eds. 1993. *New concepts in natural language generation*. Pinter, London.

Anexo

```
Function TermTV (TNeg, CNeg, InTV)
    Select Case InTV
        Case Is = False
            If TNeg = "no" Then
                If CNeg = "no" Then
                    TermTV = False
                Else
                    TermTV = True
                End If
            Else
                If CNeg = "no" Then
                    TermTV = True
                Else
                    TermTV = False
                End If
            End If
        Case Is = True
            If TNeg = "no" Then
                If CNeg = "no" Then
                    TermTV = True
                Else
                    TermTV = False
                End If
            Else
                If CNeg = "no" Then
                    TermTV = False
                Else
                    TermTV = True
                End If
            End If
    End Select
End Function
Sub DetermineCondTV(strMatrixCondition, strTypeOfCondition, intTrueElement, strChosenFeature, strConditionNegation)
  With rsOstensiveListOfConditions
    If .RecordCount > 0 Then
      .MoveFirst
      .Find ("Condition = '" & strMatrixCondition & "'")
      If .EOF() = False Then
        If Trim(.Fields.Item("Already_Evaluated").Value) = "yes" Then
          Exit Sub
```

```
            Else
               Select Case strTypeOfCondition
                  Case Is = "simple"
                     .Fields.Item("Number_Of_True_Elements").Value = intTrueElement
                     If intTrueElement = 1 Then
                        MarkComplexCondSemanticRuleForApplication strMatrixCondition
                     End If
                  Case Else
                     Select Case strTypeOfCondition
                        Case Is = "disjunctive"
                           Select Case intTrueElement
                              Case Is = True
                                 If strConditionNegation = "no" Then
                                    .Fields.Item("Already_Evaluated").Value = "yes"
                                    .Fields.Item("Number_Of_True_Elements").Value = 1
                                    MarkComplexCondSemanticRuleForApplication strMatrixCondition
                                    EvaluatePresentTermTVImpactOnConjunction strDisjunctiveTerm, "true"
                                 Else
                                    .Fields.Item("Number_Of_True_Elements").Value = _
                                                                  .Fields.Item("Number_Of_True_Elements").Value + 1
                                    .Fields.Item("Number_Of_Evaluated_Elements").Value = _
                                                                  .Fields.Item("Number_Of_Evaluated_Elements").Value + 1
                                    If Trim(.Fields.Item("Number_Of_Elements").Value) = _
                                                                  Trim(.Fields.Item("Number_Of_Evaluated_Elements").Value) Then
                                       If Trim(.Fields.Item("Number_Of_Elements").Value) = _
                                                                  Trim(.Fields.Item("Number_Of_True_Elements").Value) Then
                                          .Fields.Item("Already_Evaluated").Value = "yes"
                                          MarkComplexCondSemanticRuleForApplication strMatrixCondition
                                          EvaluatePresentTermTVImpactOnConjunction strDisjunctiveTerm, "true"
                                       End If
                                    End If
                                 End If
                              Case Is = False
                                 If strConditionNegation = "no" Then
                                    .Fields.Item("Number_Of_True_Elements").Value = _
                                                                     .Fields.Item("Number_Of_True_Elements").Value + 0
                                    .Fields.Item("Number_Of_Evaluated_Elements").Value = _
                                                                  .Fields.Item("Number_Of_Evaluated_Elements").Value + 1
                                    If Trim(.Fields.Item("Number_Of_Elements").Value) = _
                                                                  Trim(.Fields.Item("Number_Of_Evaluated_Elements").Value) Then
                                       If Trim(.Fields.Item("Number_Of_Elements").Value) <> _
                                                                  Trim(.Fields.Item("Number_Of_True_Elements").Value) Then
                                          .Fields.Item("Already_Evaluated").Value = "yes"
                                          EvaluatePresentTermTVImpactOnConjunction strDisjunctiveTerm, "false"
                                       End If
                                    End If
                                 Else
                                    .Fields.Item("Already_Evaluated").Value = "yes"
                                    .Fields.Item("Number_Of_True_Elements").Value = 0
                                    EvaluatePresentTermTVImpactOnConjunction strDisjunctiveTerm, "false"
                                 End If
                           End Select
                        Case Is = "conjunctive"
                           Select Case intTrueElement
                              Case Is = True
                                 .Fields.Item("Number_Of_True_Elements").Value =
                                                                     Fields.Item("Number_Of_True_Elements").Value      + 1
                                 .Fields.Item("Number_Of_Evaluated_Elements").Value = _
                                                                  .Fields.Item("Number_Of_Evaluated_Elements").Value + 1
                                 If Trim(.Fields.Item("Number_Of_Elements").Value) = _
                                                                  Trim(.Fields.Item("Number_Of_Evaluated_Elements").Value) Then
                                    If Trim(.Fields.Item("Number_Of_Elements").Value) = _
                                                                  Trim(.Fields.Item("Number_Of_True_Elements").Value) Then
                                       .Fields.Item("Already_Evaluated").Value = "yes"
                                       MarkComplexCondSemanticRuleForApplication strMatrixCondition
                                       EvaluatePresentTermTVImpactOnDisjunction strConjunctiveTerm, "true"
                                    Else
                                       .Fields.Item("Already_Evaluated").Value = "yes"
                                       EvaluatePresentTermTVImpactOnDisjunction strConjunctiveTerm, "false"
                                       End If
                                 End If
                              Case Is = False
                                 .Fields.Item("Already_Evaluated").Value = "yes"
                                 .Fields.Item("Number_Of_True_Elements").Value = 0
                                 EvaluatePresentTermTVImpactOnDisjunction strConjunctiveTerm, "false"
                           End Select
                     End Select
               End Select
            End If
         End If
      End If
   End With
End Sub
```

Capítulo 4

EXPANSIÓN DE CONSULTAS UTILIZANDO RECURSOS LINGÜÍSTICOS PARA MEJORAR LA RECUPERACIÓN DE INFORMACIÓN EN LA *WEB*

Claudia Deco, Cristina Bender, Jorge Saer y Mario Chiari

En Víctor M. Castel, *Comp*. (2005) *Desarrollo, implementación y utilización de modelos para el procesamiento automático de textos*. Mendoza:
Editorial de la Facultad de Filosofía y Letras, UNCuyo: 35-46.
ISBN 987-575-019-0 del soporte Internet

Expansión de consultas utilizando recursos lingüísticos para mejorar la recuperación de información en la *web*

Claudia Deco, Cristina Bender, Jorge Saer y Mario Chiari

Universidad Católica Argentina
Facultad de Química e Ingeniería Fray R. Bacon
Rosario, Argentina
{cdeco, cbender, jsaer, mchiari}@bacon.org.ar

Resumen

En los últimos años, al convertirse la *web* en el mayor repositorio de conocimiento y en un medio de publicación fácilmente accesible para todos, la Recuperación de Información ha dejado de ser un campo exclusivo de los especialistas en ciencias de la información y ha pasado a ser un campo relacionado con cualquier persona. El maximizar la cantidad de documentos relevantes obtenidos para una consulta depende de la destreza del especialista en ciencias de la información para preparar una estrategia de búsqueda, que exprese la necesidad de información del usuario. Si bien los usuarios no tienen por qué conocer técnicas de recuperación de información, la propuesta de este trabajo es la de mejorar los resultados de su búsqueda por medio de un refinador semántico que actúa como lo haría el especialista preparando una estrategia adecuada. Este refinador utiliza recursos lingüísticos, tales como tesauros, diccionarios y ontologías, para la preparación de esta estrategia y así lograr una mejora en la recuperación de información de la *web*. El refinamiento semántico propuesto consiste en: guiar al usuario para desambiguar los conceptos ingresados por él, permitirle seleccionar conceptos jerárquicamente relacionados a fin de precisar los documentos a recuperar, y expandir semánticamente los conceptos a fin de aumentar la cantidad de documentos a recuperar. Además, se analizan los resultados obtenidos al utilizar *WordNet* como recurso lingüístico.

1 Introducción

En la década de los ’90, con la aparición de *Internet* y el abaratamiento de los costos de equipamiento, el usuario dejó de concurrir a las bibliotecas ó centros de información y comenzó a buscar información por sus propios medios. Por lo tanto dejó de utilizar el apoyo del bibliotecario ó del experto en ciencias de la información para expresar su necesidad de información. Como consecuencia, y agregando a ésto la explosión de información disponible en la *web*, resulta muy difícil para el usuario encontrar eficientemente información útil, dado que no es capaz de preparar una estrategia de búsqueda adecuada. Por ejemplo, no usar sinónimos puede reducir notoriamente la cantidad de documentos recuperados, ó una frase de búsqueda incompleta puede retornar muchos documentos irrelevantes. Es decir, el exponencial crecimiento de información disponible en la *web* lleva al problema que los usuarios no son capaces de encontrar la información que buscan en una forma eficiente y simple, y frecuentemente no ven satisfechas sus necesidades de información.

En el entorno de búsqueda tradicional, el usuario debe dividir su interés de búsqueda en distintos conceptos. Luego debe pensar en cómo los conceptos y los términos asociados con ellos corresponden a la representación de la información almacenada. Una vez que los términos han sido elegidos, pueden ser combinados para formar la consulta.

No siempre un término representa en forma adecuada un concepto de interés para el usuario. Encontrar otros términos equivalentes ó más adecuados para expresar un concepto es realizar una *expansión de consulta* (Efthimiadis 1996). Esta situación requiere un cambio en el pensa-

miento del proceso para elegir los términos de búsqueda. Podría ser necesario consultar recursos lingüísticos, tales como un tesauro o un diccionario para incorporar nuevos términos. Este esfuerzo usualmente requiere entrenamiento especializado o experiencia por parte de los usuarios, porque los usuarios inexpertos probablemente no obtendrán buenos resultados.

La expansión de consultas es el proceso de suplementar la consulta original con términos adicionales, y puede ser considerado como un método para mejorar el desempeño de la recuperación. La consulta inicial, tal como es provista por el usuario, puede ser una representación inadecuada o incompleta de la información que éste necesita, ya sea en sí misma o en relación a la representación de la información en los documentos.

Un método para seleccionar términos para la expansión de la consulta es la realimentación por relevancia. En este método, los documentos recuperados en una iteración anterior de la búsqueda, que han sido identificados como relevantes por el usuario, proveen los términos para la expansión de la consulta. Otro método es disponer de una estructura de conocimiento que sea independiente del proceso de búsqueda, tal como un tesauro ó un diccionario específico del dominio ó un diccionario global. Un ejemplo de un recurso específico del dominio es MeSH (www.ncbi.nlm.nih.gov/entrez/meshbrowser.cgi) en medicina, y un ejemplo de un recurso global es *WordNet* (http://wordnet.princeton.edu/).

La expansión de consultas puede ser desarrollada manual, automática o interactivamente. Un problema en cualquier tipo de expansión de consulta, es cómo definir cuáles términos están estrechamente asociados con los términos de la consulta. En una expansión de consulta interactiva se consideran métodos donde al usuario se le proponen términos de búsqueda como parte del proceso de reformulación de la consulta. En este tipo de expansión hay una responsabilidad conjunta entre el sistema y el usuario en la selección de términos para la expansión. Por un lado, el sistema sugiere términos y los presenta al usuario; y por el otro es el usuario quien toma la decisión final sobre la importancia relativa y la utilidad de un término.

Si bien los usuarios no tienen por qué conocer técnicas de recuperación y extracción de información, se mejorarían los resultados de su búsqueda si por medio de una interfaz que implemente estas técnicas se hiciera una expansión de su consulta y así lograr que en la respuesta los documentos recuperados sean los documentos relevantes. En este trabajo se presenta un refinamiento semántico para mejorar la precisión a través de la expansión semiautomática de la consulta, requiriendo una interacción mínima del usuario y no a través de la automatización completa de la búsqueda. Además, se analizan los resultados obtenidos con la utilización de *WordNet* como recurso lingüístico en la expansión de la consulta para la preparación de una estrategia de búsqueda adecuada a la necesidad del usuario.

El refinamiento semántico consiste en guiar al usuario para *desambiguar* los conceptos ingresados por él, permitirle *seleccionar* conceptos jerárquicamente relacionados a fin de precisar los documentos a recuperar y *expandir* semánticamente los conceptos a fin de aumentar la cantidad de documentos a recuperar.

Un problema que se presenta con respecto a la semántica es la *desambiguación* de conceptos. Basta un ejemplo muy simple como el hecho de buscar la palabra *cáncer* para comprobarlo. Cáncer puede referirse a la enfermedad, a la constelación de estrellas o al signo zodiacal. Esta desambiguación que realiza el usuario permite continuar el proceso, en las etapas siguientes, con sólo aquellos términos que están vinculados conceptualmente con la acepción de interés del usuario. La solución propuesta es utilizar recursos tales como diccionarios u ontologías, donde se pueda decidir dentro de qué contexto se está buscando el término ingresado por el usuario.

El objetivo de la *selección de conceptos jerárquicamente relacionados* es mostrarle al usuario una jerarquía de conceptos vinculados con el concepto ingresado por él, a fin de que éste se reubique, si es necesario, en la jerarquía conceptual para refocalizar su búsqueda y así aumentar la precisión en la recuperación. El objetivo de la *expansión semántica* es recuperar documentos que también sean relevantes aún cuando no respondan rigurosamente a los términos utilizados por el usuario. Es decir, la expansión semántica consiste en incorporar a la búsqueda términos que sean conceptualmente equivalentes. La expansión del concepto también puede hacerse desde el punto de vista multilingual, utilizando diccionarios multilinguales para obtener dichos

conceptos en otros idiomas de interés para el usuario.

El esfuerzo inicial que se pretende por parte del usuario en la desambiguación de términos y en la selección de conceptos relacionados sugeridos por el sistema, será recompensado evitándole a posteriori la lectura y el descarte de los documentos que no sean de su interés.

2 Conceptos básicos

La *Recuperación de Información* es la representación, almacenamiento, organización y acceso a ítems de información (Baeza et al. 1999). La meta principal de la Recuperación de Información es recuperar todos los documentos que sean relevantes a una consulta del usuario y recuperar la mínima cantidad de documentos no relevantes. En el modelo tradicional la información se organiza en documentos y se supone que existe un gran número de éstos. El proceso de recuperación consiste en localizar los documentos de importancia de acuerdo con la información aportada por el usuario. Un ejemplo típico de un sistema de recuperación de información son los catálogos de las bibliotecas, donde una entrada del catálogo es ejemplo de un documento. El usuario de este sistema puede desear recuperar un documento concreto ó un conjunto de éstos.

Un sistema de recuperación de *datos* sólo recupera datos que coinciden exactamente con el patrón a recuperar; mientras que un sistema de recuperación de *información* recupera datos que hagan la mejor coincidencia parcial con el patrón dado. Esto se debe a que la recuperación de información generalmente trata con texto de lenguaje natural, el cual no está siempre bien estructurado y podría ser semánticamente ambiguo. Por ejemplo, si se realiza una consulta por el término *cáncer*, además de obtener como resultado los documentos que contengan este término, se debería obtener también los documentos en que aparezca *neoplasma* o *carcinoma*.

El objetivo principal de la Recuperación de Información es satisfacer la necesidad de información planteada por un usuario en una consulta en lenguaje natural especificada a través de un conjunto de conceptos. Una *consulta* es una solicitud de documentos pertenecientes a algún tema. Dada una colección de documentos y una consulta, el objetivo de una estrategia de búsqueda es obtener todos y sólo los documentos relevantes a la consulta. Una *estrategia de búsqueda* es una expresión lógica compuesta por distintos conceptos combinados con los conectores lógicos de conjunción, disyunción y negación. En general, este proceso hacia la recuperación de documentos relevantes a la consulta presentada, no es simple debido a la complejidad semántica del vocabulario.

En la recuperación de información existe además la figura del *especialista en ciencias de la información* que es el encargado de expresar la necesidad de información del usuario en una estrategia de búsqueda. El maximizar la cantidad de documentos relevantes obtenidos para esta consulta depende de la correcta preparación de esta estrategia de búsqueda.

Al convertirse la *web* en el mayor repositorio de conocimiento fácilmente accesible para todos, resurge la Recuperación de Información que deja de ser un campo exclusivo de bibliotecarios y especialistas en ciencias de la información y pasa a ser un campo relacionado con cualquier persona. Sin embargo, los usuarios al buscar información en la *web* se enfrentan con varios problemas. Uno de ellos es cómo especificar la consulta (Baeza 1998).

Las estadísticas (Kobashayi et al. 2000) indican que el número promedio de palabras que se utilizan por consulta es de 2 palabras. El número de operadores lógicos por consulta es de 0,4. Las veces que se repiten las consultas es cuatro (en un rango de 1 a 1.5 millones). Por sesión, un usuario hace en promedio dos consultas. El 80% de los usuarios no modifica su consulta inicial y el 85% ve sólo la primera página de la respuesta. Esto indicaría que la gran mayoría de los usuarios desconoce las técnicas de recuperación de información, y tiene dificultad de expresar claramente su necesidad de información, y por lo tanto, no obtienen los resultados deseados. El 85% de los usuarios de Internet utiliza motores de búsqueda para encontrar información específica. El mismo estudio muestra que los usuarios no están conformes con la *performance* brindada por los motores de búsqueda ni con la calidad de los resultados obtenidos.

Las técnicas de recuperación de información no son triviales y si bien los usuarios no tienen

porqué conocerlas, ya que son propias de las ciencias de la información, se mejorarían los resultados de su búsqueda por medio de una interfaz que implemente estas técnicas. En este trabajo, se presenta un refinador semántico que actúa como especialista de ciencias de la información. Este refinador le sugiere al usuario la desambiguación del término a buscar, le permite la selección de un concepto jerárquicamente relacionado más cercano a su necesidad, y realiza la expansión semántica y multilingual de los términos, a fin de preparar una estrategia de búsqueda adecuada a su necesidad de información.

La Recuperación de Información tradicional utiliza los indicadores Precisión y *Recall* para evaluar los resultados de la búsqueda. La Precisión se define como el ratio de documentos relevantes sobre el número total de documentos recuperados y el *Recall* se define como la proporción de los documentos relevantes que son recuperados.

Los usuarios expertos en un área del conocimiento pueden trabajar con un *recall* alto y una precisión baja, porque son capaces de examinar la información y rechazar fácilmente la irrelevante. Los usuarios no expertos en un tema, por otro lado, necesitan más alta precisión porque les falta experiencia y conocimiento en el tema. Con respecto a los términos utilizados en la búsqueda, si son muy específicos aumenta la precisión y baja el *recall*. En cambio, si los términos son muy amplios ó generales aumenta el *recall* y baja la precisión.

Realizada una búsqueda en una colección de documentos, el conjunto de documentos recuperados no coincide totalmente con el conjunto de los relevantes sobre el tema de interés. Una búsqueda será óptima cuando estos dos conjuntos coincidan, es decir cuando todos los documentos recuperados sean relevantes y todos los documentos relevantes sean recuperados. Es decir, cuando tanto la precisión como el *recall* sean máximos. Estos indicadores se aplican también a la Recuperación de Información en la Web.

Los recursos lingüísticos que se pueden utilizar en la preparación de la estrategia de búsqueda son diccionarios, tesauros y ontologías.

Diccionarios: Indican las distintas acepciones de un término y permiten su expansión con sinónimos. Un diccionario muy utilizado como recurso es *WordNet* (Miller 1995), que es un sistema de referencia léxica y provee las distintas acepciones de un concepto, permitiendo además la expansión de éste con sinónimos, merónimos, hipónimos y otros tipos de términos relacionados a la acepción elegida.

Para aumentar el número de documentos a recuperar se puede ampliar cada concepto en los idiomas deseados por los usuarios mediante el uso de diccionarios multilinguales generales y especializados que permiten traducir un concepto a otros idiomas.

Tesauros: La flexibilidad y variedad del lenguaje natural crea serias dificultades para el manejo automatizado de la información. Para solucionar este problema, surgen los tesauros, que permiten el control del vocabulario para representar en forma unívoca cada concepto.

Según la definición de la Unesco, un tesauro es un instrumento de control terminológico utilizado para traducir a un lenguaje más estricto el idioma natural empleado en los documentos y por los indizadores, que son las personas que asignan las palabras claves a cada documento. Por su estructura, es un vocabulario controlado y dinámico de términos relacionados semántica y genéricamente, los cuales cubren un dominio específico del conocimiento.

El tesauro está estructurado formalmente para hacer explícitas las relaciones entre conceptos. Estas relaciones pueden ser jerárquicas, de afinidad, y preferenciales. Las relaciones jerárquicas indican términos más amplios ó más específicos de cada concepto. Las de afinidad muestran términos relacionados conceptualmente, pero que no están ni preferencial ni jerárquicamente relacionados. Las relaciones preferenciales se utilizan para indicar cuál es el término preferido o descriptor entre un grupo de sinónimos; y para la calificación de homónimos a fin de diferenciar su significado, eligiendo un significado preferido para cada término.

A diferencia de un diccionario, donde todos los sinónimos de un concepto son representativos y tratados por igual, en un tesauro se tiene una palabra clave preferida y representativa del conjunto de sinónimos para cada concepto.

Ontologías: Proporcionan una vía para representar el conocimiento y son un enfoque importante para capturar semántica (Studer et al. 1998). Consisten de términos, sus definiciones y

axiomas que los relacionan con otros términos. Los términos están organizados en una taxonomía y los axiomas permiten realizar búsquedas con inferencias.

Para poder darle semántica a la *web*, se necesitan lenguajes de marcado apropiados que representen el conocimiento. El lenguaje XML (*eXtensible Markup Language*) con sus respectivos DTD (*Document Type Definition*) no es suficiente para esto (Broekstra et al. 2002). Existen otros lenguajes de marcado como ser RDF (*Resource Description Framework*), que permite representar algunos aspectos sobre conceptos de un dominio y, mediante relaciones taxonómicas, crear una jerarquía de conceptos. RDF es recomendado por el consorcio W3C (www.w3.org) como estándar para los metadatos. Un lenguaje con gran capacidad expresiva que está emergiendo como un estándar para representar ontologías en la *web* es OWL (*Ontology Web Language*), y es desarrollado por el consorcio W3C.

A diferencia de los tesauros y de los diccionarios, en las ontologías se agregan axiomas que permiten realizar inferencias sobre los conceptos.

3 Trabajos relacionados

Se han realizado muchas experiencias para el aprovechamiento de los recursos lingüísticos en distintas áreas, entre ellas en la recuperación de información para la expansión de la consulta. Voorhees (1998) argumenta que las expansiones con recursos lingüísticos son efectivas para consultas con muy pocos conceptos, mientras que no trae mucha mejora para consultas con muchos conceptos. Mandala et al. (1998) concluyen que las expansiones de la consulta con *Wordnet* pueden mejorar la cantidad de documentos a recuperar pero decrece la precisión. En Martínez y García (2002) se presenta un sistema donde la consulta se ingresa en lenguaje natural, el sistema detecta las palabras de interés para la búsqueda y luego utiliza recursos lingüísticos para expandirla. Navigli y Velardi (2003) experimentan el uso de ontologías para extraer el dominio semántico de una palabra y expandir la consulta agregando términos que a menudo coocurren con las palabras de la consulta. Sangoi Pizzato y Strube (2003) expanden la consulta utilizando tesauros y muestran que esta propuesta mejora la recuperación en la *web*. En Carpineto et al. (2002) se propone realizar una realimentación por relevancia, incorporando a la consulta palabras de los documentos que el usuario marcó como de su interés. Por otra parte, Cui et al. (2000) proponen expandir la consulta con términos obtenidos de un perfil de usuario.

En general estos proyectos amplían la búsqueda en una sola dirección. Algunos lo hacen expandiendo los conceptos semánticamente. Muy pocos corrigen ortográficamente los términos sugiriéndole al usuario la forma ortográfica correcta. En general, no le permiten al usuario precisar su interés de búsqueda seleccionando un concepto jerárquicamente relacionado. Por otra parte, la mayoría intentan automatizar completamente todas las tareas. Aquí se propone mejorar la precisión a través de una interacción mínima del usuario. Este esfuerzo inicial que se pretende por parte del usuario será recompensado evitándole a posteriori la lectura y la clasificación manual de los documentos que no sean de su interés.

4 El refinamiento semántico

El *refinamiento semántico* que se propone consiste en guiar al usuario para *desambiguar* los términos ingresados por él, permitirle *seleccionar* conceptos jerárquicamente relacionados a fin de mejorar la precisión en los documentos a recuperar y *expandir* semánticamente los conceptos a fin de aumentar la cantidad de documentos a recuperar. La arquitectura del refinador semántico se presenta en la Figura 1, donde los módulos sombreados indican que se necesita la participación del usuario.

Para realizar una consulta, el usuario ingresa un conjunto de conceptos $\{C_i\}$ con $1 \leq i \leq n$, que representan su interés de búsqueda; y la salida del *Refinador Semántico* es una estrategia de búsqueda asociada a estos conceptos.

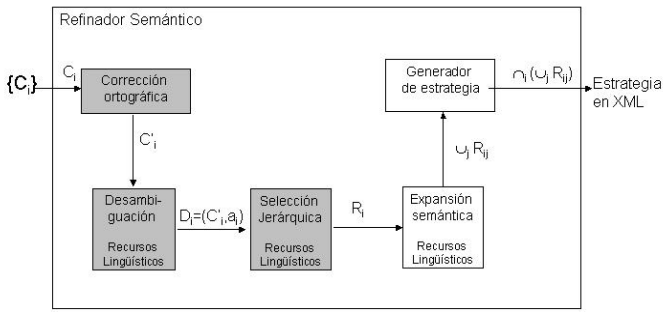


Figura 1. Arquitectura del Refinador Semántico.

Corrección ortográfica: Verifica que los términos que representan cada concepto, estén correctamente escritos. Por cada término C_i que ingresa, se obtiene como salida un término corregido C'_i. Si C_i está bien escrito, C'_i coincide con C_i. Si C_i estuviera incorrectamente escrito, entonces se lo reemplaza, previa aceptación del usuario, por C'_i.

Desambiguación: En este módulo, por cada término C'_i se muestra al usuario las distintas acepciones asociadas. El usuario selecciona la acepción que corresponde a su interés de búsqueda. Cada acepción de un término tiene una jerarquía conceptual asociada. La salida de este módulo es el término D_i desambiguado de la forma (C'_i, a_i), donde C'_i es el término ingresado y a_i es la acepción elegida por el usuario. La desambiguación que realiza el usuario permite continuar el proceso, en las etapas siguientes, con sólo aquellos términos que están vinculados conceptualmente con la acepción de interés.

Selección jerárquica: Muestra para cada concepto D_i los conceptos jerárquicamente relacionados con éste. Si existen conceptos relacionados para algún D_i entonces se le permite al usuario moverse en la jerarquía conceptual del mismo. Esto le permite ubicar un concepto más cercano a su necesidad, y *reemplazar* el de partida D_i por algún otro J_i que se encuentra jerárquicamente relacionado en un nivel superior ó inferior, ó eventualmente en otra rama del árbol de jerarquía, y así aumentar la precisión en la recuperación. Esta etapa es interactiva porque el usuario puede elegir estos conceptos relacionados provistos por el refinador a partir de los recursos lingüísticos. Si al usuario le interesa un conjunto de conceptos $J_{i,1}$, ..., $J_{i,s}$ de la jerarquía asociada al concepto D_i, la salida de este módulo es la unión de éstos. Entonces, la entrada a este módulo es D_i y la salida, que para simplificar la notación llamaremos R_i, puede ser:

- D_i si el usuario decidió no cambiar de nivel jerárquico;
- J_i si decidió reemplazar el concepto D_i, por otro jerárquicamente relacionado;
- $\{ J_{i,1}, ..., J_{i,s} \}$ si decidió reemplazar D_i por un conjunto de conceptos relacionados.

Generalmente, la tercera posibilidad se presenta cuando se ingresa por un término general e interesan dentro de éste varios hipónimos, es decir, varios términos específicos. En este recorrido conceptual puede ocurrir que el usuario decida seleccionar un concepto específico, el cual sea ambiguo. Para no volver a requerir su participación, se automatiza esta desambiguación arrastrando la acepción original elegida.

Expansión semántica: El objetivo de esta expansión es recuperar documentos que también sean relevantes aún cuando no respondan rigurosamente a los términos utilizados por el usuario. Consiste en incorporar a la búsqueda términos conceptualmente equivalentes. Por ejemplo, ante la búsqueda del término *padre*, se puede expandir agregando su sinónimo *papá* y su término relacionado *madre*. Esta expansión es automática y permite aumentar la cantidad de documentos a recuperar. La salida de este módulo es un conjunto de *r* términos relacionados semánticamente $\{R_{i1}, ..., R_{ik} ... R_{ir}\}$ asociados a cada concepto R_i, con $1 \leq i \leq n$.

Generación de estrategia: La salida de este módulo contiene la estrategia de búsqueda asociada al interés del usuario, representada en XML. Esta estrategia consiste en realizar en primer

lugar el OR lógico de las expansiones de cada concepto; y luego el AND lógico de estas expansiones. Si se desea hacer una búsqueda que *no* contenga un determinado concepto, este concepto a descartar se expande en la forma descrita a fin de considerar otros sinónimos a descartar también. Luego se realiza el NOT del OR obtenido para este concepto a negar y se lo agrega al AND lógico. Es decir, para una búsqueda que involucra los conceptos C_1 y .... y (no C_h) y ... y C_n se obtiene la estrategia siguiente:

$$(R_{11} \text{ OR } R_{12} \text{ OR } ... \text{ OR } R_{1\,r1}) \quad \text{AND} \quad \ldots \text{ AND} \quad (\text{NOT } (R_{h1} \text{ OR } R_{h2} \text{ OR } ... \text{ OR } R_{h\,rh})) \text{ AND } ... \text{ AND } (R_{n1} \text{ OR } R_{n2} \text{ OR } ... \text{ OR } R_{n\,rn})$$

donde: $(R_{i1} \text{ OR } R_{i2} \text{ OR } ... \text{ OR } R_{i\,ri})$ es la expansión del concepto C_i.

Esta estrategia representada en XML, es luego traducida a la sintaxis de las distintas herramientas de consulta, por ejemplo buscadores. Este refinador está inmerso dentro de una arquitectura general propuesta en (Motz et al. 2003).

En la preparación de una estrategia de búsqueda pueden presentarse contingencias: cómo reducir la cantidad de documentos si se recuperan demasiados, y cómo aumentar la cantidad si no se recupera información suficiente. En la recuperación de información tradicional, cuando un usuario recupera demasiados documentos como resultado de una consulta, pudo haber cometido errores de estrategia ó errores de entrada. Los errores de estrategia pueden provenir del uso de términos ambiguos o de términos no específicos, de la falta de conceptos, del uso de disyunción (OR) cuando debería haber usado conjunción (AND), del uso de truncamiento demasiado corto de términos, ó del uso incorrecto de paréntesis.

En el caso de que el usuario recupere pocos ó ningún documento como resultado de una consulta, pudo haber cometido también errores de estrategia ó errores de entrada. Los errores de estrategia en este caso pueden provenir del uso de demasiados conceptos, de no incluir sinónimos suficientes, de la utilización de términos demasiado específicos, del uso de operadores de proximidad sintáctica entre términos, del uso de conjunción (AND) cuando debe usarse disyunción (OR), del uso incorrecto de la negación (NOT), ó del uso incorrecto de paréntesis. Los errores de entrada en ambos casos pueden deberse a errores de tecleo, ó errores de deletreo (distintas formas de escribir una palabra, por ejemplo "*color*" y "*colour*").

El refinador semántico resuelve la mayoría de estos problemas: la desambiguación de términos ambiguos, la especificación de términos no específicos, el correcto uso de la disyunción y de la conjunción, el uso correcto de paréntesis, la inclusión de sinónimos y palabras con distintas formas de escritura, el uso correcto de la negación y los errores de tecleo.

La cantidad de documentos recuperados aumenta si se amplía en forma automática el criterio de búsqueda ingresado por el usuario, mediante el agregado de sinónimos y palabras relacionadas. La mejora en la precisión de los resultados se logra presentándole al usuario una estructura jerárquica de conceptos que le permite hacer un recorrido conceptual de su consulta. Es decir, moverse por jerarquías conceptuales, subiendo ó bajando de nivel conceptual, y seleccionando un término más preciso a su necesidad de información.

Ejemplo: Supongamos que un usuario desea obtener información sobre *cáncer de pulmón*, y decide ingresar en inglés el concepto más general *cancer*. El refinador semántico toma esta palabra y verifica que está correctamente escrita desde el punto de vista ortográfico. Si el usuario hubiera ingresado *canser*, el corrector le sugiere la palabra *cancer*, ortográficamente correcta. La palabra *cancer* ingresa al módulo Desambiguación, el cual a través de un recurso lingüístico le muestra las distintas acepciones de esa palabra. Si se utiliza *WordNet* como recurso, se observa que el sistema provee cinco acepciones distintas de esta palabra. El usuario decide que la acepción de interés es la primera: medicina. El módulo Selección de jerarquía expande entonces este concepto con sus hipónimos. El usuario se mueve en la jerarquía y se queda con la frase *lung cancer*, la cual ingresa al módulo Expansión semántica. Este módulo la expande e incorpora automáticamente el término: *carcinoma of the lungs*. Si en la expansión se utilizan otros recursos tales como un diccionario multilingual y un tesauro de medicina, se incorporan: *cáncer de pulmón* y *lung neoplasms*. El Generador de estrategia, toma este conjunto de términos y, en

forma automática, construye la siguiente estrategia de búsqueda:

lung cancer OR *carcinoma of the lungs* OR *cáncer de pulmón* OR *lung neoplasms*

Generalmente, una búsqueda involucra varios conceptos. En estos casos, el refinador semántico trata cada uno de éstos en forma independiente, y los combina en el módulo Generación de estrategia. Como resultado, la estrategia de búsqueda asociada consta de la disyunción de cada una de las expansiones y luego la conjunción de los conjuntos resultantes de las expansiones. Por ejemplo, si se desea saber la *relación de la aspirina en el tratamiento del cáncer de pulmón*. Los conceptos que podría ingresar el usuario son: *cáncer de pulmón, aspirina* y *tratamiento* y una estrategia resultante es:

(lung neoplasms OR *lung cancer* OR *cáncer de pulmón* OR *carcinoma of the lungs)*
AND *(aspirina* OR *aspirin* OR *ácido acetil salicílico)*
AND *(tratamiento* OR *treatment)*

Prototipo: Para su desarrollo se utilizaron estándares y recomendaciones del consorcio W3C (www.w3.org) así como lenguajes y recursos libres disponibles en la *web*. Para la corrección ortográfica se utiliza el método *Spelling Suggestion* del web service de *Google* (www.google.com/apis). Para la selección de jerarquía y la expansión semántica se utiliza el recurso lingüístico *WordNet*. Estos servicios fueron ensamblados y provistos de una interfaz *web* sencilla. Se adoptó PHP (www.php.net) como lenguaje para la implementación. Se analizaron distintos buscadores encontrando que *Google* tiene la limitación de 10 palabras por consulta, y una estrategia compleja puede llegar a tener muchas más. Se analizó *Yahoo*! y se observó que no tiene esta limitación. Por eso se utilizó este último en las experiencias.

5 Experimentación

Para probar el refinamiento semántico, se realizaron 24 consultas. Para cada consulta se solicitó al usuario que describiera su interés de búsqueda en sus propias palabras, y que luego realizara la consulta de dos formas: primero en el buscador *Yahoo*! y luego con el refinamiento semántico. Se registró la estrategia planteada por el usuario directamente a *Yahoo*! y se registró la estrategia generada por el refinador semántico, que luego se ejecutó en *Yahoo*!. Además, en cada prueba se registró la cantidad de documentos resultantes y la cantidad de documentos que respondían al interés del usuario en los primeros 50 documentos, a fin de medir luego la precisión en los primeros 50 documentos. Además se registró el tipo de usuario que realizaba la consulta. Se consideró de nivel Inexperto a aquel usuario que no estaba habituado al uso de un buscador. El nivel Medio corresponde a los usuarios que realizan consultas a través de buscadores con frecuencia. Un usuario de nivel Experto es aquel que utiliza las opciones de Búsqueda Avanzada en los buscadores.

El objetivo de las experiencias realizadas fue evaluar el refinamiento semántico, utilizando un recurso lingüístico particular, *WordNet*, para la preparación de la estrategia de búsqueda.

De los resultados obtenidos se puede observar que, en general, el usuario no utiliza la *búsqueda por frases*. Por ejemplo, Gabriel García Márquez son tres palabras que forman parte de un solo concepto y debería buscarse como una unidad: “Gabriel García Márquez”. El refinador genera automáticamente frases a partir de conceptos formados por más de una palabra, ya sea que estos conceptos estén en *WordNet* o no. El uso de frases en la estrategia de búsqueda aumenta la precisión de la recuperación, y disminuye la cantidad de documentos recuperados.

En el caso de *nombres propios* que no están en *WordNet* no varía la precisión con respecto a la búsqueda sin refinamiento, excepto que estos nombres propios sean frases, en cuyo caso la precisión mejora.

La estrategia generada con refinamiento semántico no difiere mucho de la planteada por un *usuario experto*. Por lo tanto, los resultados de la búsqueda con refinamiento son bastante similares a los resultados sin refinamiento. La estrategia generada con refinamiento semántico mejo-

ra la precisión en el caso de *usuarios inexpertos ó medios*.

En general, el usuario no ingresó términos con *errores ortográficos*, pero en las consultas donde ingresó términos con errores ortográficos, la corrección ortográfica realizada por el refinador aumentó la cantidad de documentos recuperados y la precisión de los mismos.

Mediante el refinamiento semántico se permite la *navegación por una jerarquía conceptual*, donde al poder seleccionar el usuario términos más específicos aumenta la precisión. La utilización de *sustantivos adjetivados*, como por ejemplo *"spanish civil war"*, como un solo concepto, aumenta la precisión y disminuye la cantidad de documentos recuperados. Este concepto es más específico que "*war*" y se obtiene moviéndose por la jerarquía conceptual.

Analizado el número de conceptos utilizados en cada consulta, en aquellas que involucran más de un concepto, el promedio de la cantidad de documentos recuperados aumenta luego del refinamiento semántico. En el caso de consultas que involucran un solo concepto, si el refinamiento consiste en sólo agregar sinónimos, aumenta la cantidad de documentos recuperados. Pero, si el refinamiento consiste en cambiar el concepto inicial por uno más específico, la cantidad de documentos recuperados disminuye.

Analizado el número de conceptos utilizados en cada consulta, el promedio de la precisión aumentó luego del refinamiento, en las consultas que involucran uno ó dos conceptos. Para las consultas que involucran más de dos conceptos, disminuyó la precisión en la experiencia realizada. Una posible causa es el agregado por parte del refinador de siglas cortas como sinónimos, por ejemplo, se agregan *US*, *USA* para *United States*.

Finalmente, se promediaron la cantidad de documentos recuperados y la precisión en los primeros 50 documentos sin y con refinamiento semántico. Los resultados se muestran en la Tabla 1.

	Recuperados	Precisión
Sin refinamiento	533811,67	0,46
Con refinamiento	651726,67	0,55
	22,09 %	19,03 %

Tabla 1. Promedios de cantidad de documentos recuperados y precisión en los primeros 50.

De los promedios se observa que el refinamiento semántico mejora la cantidad de documentos recuperados en un 22,09 % y mejora la precisión en un 19,03 %. Esto muestra que la propuesta de refinamiento semántico presentada mejora la recuperación de información en la *web* al utilizar *WordNet* para la preparación de la estrategia de búsqueda. Estos resultados no difieren mucho de los presentados por Sangoi Pizzato y Strube (2003) en un trabajo similar donde la expansión de la consulta se basa en un tesauro como recurso lingüístico.

6 Conclusiones

Al convertirse la *web* en el mayor repositorio de conocimiento y en un medio de publicación fácilmente accesible para todos, la Recuperación de Información ha dejado de ser un campo exclusivo de los especialistas en ciencias de la información y ha pasado a ser un campo relacionado con cualquier persona. Si bien los usuarios no tienen por qué conocer técnicas de recuperación de información, la propuesta es la de mejorar los resultados de su búsqueda por medio de un "especialista" que implementa estas técnicas. El refinador semántico propuesto es el que actúa como lo haría el especialista en ciencias de la información expandiendo los términos de la consulta para preparar una estrategia de búsqueda adecuada a la necesidad de información del usuario.

El refinamiento semántico propuesto consiste en: guiar al usuario para desambiguar los conceptos ingresados por él, permitirle seleccionar conceptos jerárquicamente relacionados a fin de precisar los documentos a recuperar y expandir semánticamente los conceptos a fin de aumentar la cantidad de documentos a recuperar. Los recursos lingüísticos que pueden utilizarse para el refinamiento semántico son tesauros, diccionarios y ontologías. Qué recurso ó recursos se pue-

den utilizar depende del área del conocimiento. Se propuso un refinamiento semiautomático pues se considera que el esfuerzo inicial que se pretende por parte del usuario en la desambiguación y en la selección jerárquica es recompensado evitándole a posteriori la lectura y el descarte de los documentos que no sean de su interés.

La cantidad de documentos recuperados aumenta mediante el agregado de sinónimos y palabras relacionadas. La mejora en la precisión de los resultados se logra presentándole al usuario una estructura jerárquica de conceptos que le permite hacer un recorrido conceptual de su consulta. Es decir, moverse por jerarquías conceptuales, subiendo ó bajando de nivel conceptual, y seleccionando un término más cercano a su necesidad de información.

Para la experimentación se utilizó el recurso lingüístico *WordNet*. Cabe destacar que en los resultados generales tanto el promedio de la cantidad de documentos recuperados como la precisión se incrementan en cerca de un 20%. Esto muestra que la propuesta de refinamiento semántico presentada mejora la recuperación de información de la *web* al utilizar *WordNet* como recurso lingüístico para la preparación de la estrategia de búsqueda.

El refinador semántico resuelve la mayoría de los problemas relacionados con las contingencias: la desambiguación de términos ambiguos, el correcto uso de la disyunción y la conjunción, el uso correcto de paréntesis, la inclusión de sinónimos y palabras con distintas formas de escritura, la utilización de términos específicos, el uso correcto de la negación y los errores de tecleo. Queda abierto el problema de, en caso de obtener como resultado pocos documentos porque se ingresaron demasiados conceptos, determinar qué concepto quitar de la estrategia ó detectar cuál es el término demasiado específico para realizar un nivel menos de especificación, a fin de aumentar la cantidad de documentos a recuperar. Entre otros temas abiertos están: la utilización de un perfil de usuario para la selección automática de los recursos lingüísticos más adecuados; la utilización de realimentación por relevancia para la expansión de la consulta y la utilización de ontologías con axiomas para incorporar a la estrategia nuevos conceptos obtenidos a través de la inferencia.

Referencias

R. Baeza-Yates. 1998. *Searching the Web: Challenges and Partial Solutions*. Depto. de Ciencias de la Computación. Universidad de Chile. Proyecto VII.13.AMYRI – CYTED.

R. Baeza-Yates y B. Ribeiro-Neto (eds.). 1999. *Modern Information Retrieval*. New York. ACM Press.

J. Broekstra, M. Klein, S. Decker, F. van Harmelen e I. Horrocks. 2002. Enabling knowledge representation on the Web by extending RDF Schema. *Computer Networks* 39: 609-634.

C. Carpineto, G. Romano, V. Giannini. 2002. Improving retrieval feedback with multiple term-ranking function combination. *TOIS* 20(3): 259-290.

H. Cui, J. Wen, J. NIE, W. Ma. 2002. Probabilistic Query expansion using Query logs. WWW202, may 7-11, Hawai, USA, *ACM* 1-58113-449.

E. N. Efthimiadis. 1996. Query Expansion. *Annual Review of Information Systems and Technology* (ARIST), 31: 121-187.

M. Kobayashi y K. Takeda. 2000. Information Retrieval on the Web. *IBM Research*, Tokyo, Japan.

R. Mandala, T. Takenobu and T. Hozumi. 1998. The use of Wordnet in information retrieval. *Proceedings of Coling - ACL*.

P. Martínez, A. García. 2002. Utilizando recursos lingüísticos para mejora de la recuperación de información en la Web. *Revista Iberoamericana de Inteligencia Artificial* 16: 55-64.

G. Miller. 1995. A lexical database for English. *Communication of the ACM* 38(11): 39-41.

R. Motz, C. Deco, C. Bender. 2003. Arquitectura de un asistente para la recuperación semántica de referencias bibliográficas en la Web. *Anales de la 32 Jornadas Argentinas de Informática e Investigación operativa* (32 JAIIO).

R. Navigli, P. Velardi. 2003. An analysis of ontology-based query expansion strategies. *Workshop on Adaptive Text Extraction and Mining* (ATEM 2003), *14th European Conference on Machine Learning*.

L. Sangoi Pizzato, V. Strube de Lima. 2003. Evaluation of a Thesaurus-Based Query Expansion Technique. *PROPOR'2003*. Faro, Portugal, June 26-27.

S. Studer, R. Benjamins y D. Fensel. 1998. Knowledge Engineering: Principles and Methods. *Data and*

Knowledge Engineering, 25: 161-197.
E. Voorhees. 1998. Using Wordnet for Text Retrieval. In Fellbaum C. *WordNet, an electronic Lexical Database*, MIT Press.

Capítulo 5

EJERCICIOS DE TRADUCCIÓN AUTOMÁTICA CATALÁN - CASTELLANO

Gustavo A. González Capdevila

En Víctor M. Castel, Comp. (2005) *Desarrollo, implementación y utilización de modelos para el procesamiento automático de textos*. Mendoza: Editorial de la Facultad de Filosofía y Letras, UNCuyo: 47-58.
ISBN 987-575-019-0 del soporte Internet

Ejercicios de traducción automática catalán - castellano

Gustavo Alberto González Capdevila

Facultad Católica de Química e Ingeniería "Fray Rogelio Bacon"
Rosario – Argentina
info@gonzalezcapdevila.com.ar // ggonzalez@bacon.org.ar

Resumen

Esta presentación está basada en el libro titulado *Catalán-Castellano: Ejercicios de traducción automática basados en el modelo de Syntactic Structures*, publicado en castellano por la Editorial de la Universidad Nacional de Rosario (© Gustavo Alberto González Capdevila, Junio 2005). El principal objetivo de esta ponencia es demostrar la validez de la teoría del análisis sintáctico de Noam Chomsky para el catalán y el castellano, basada en el concepto de árbol y mapa sintáctico que permiten demostrar de una forma visual e intuitiva los principales aspectos de ella. Por otro lado, la gramática transformacional o de traducción automática catalán-castellano también se basa en la estructura tabular de mapa sintáctico empleada aquí, que permite visualizar la conversión de las diferentes estructuras sintácticas del idioma catalán al idioma castellano. La aplicación de estas teorías tienen como objetivo la implementación de un *sistema prototipo de traducción automática catalán-castellano*, actualmente en desarrollo y que será presentado durante esta ponencia, como consecuencia de ofrecer una modesta alternativa de solución con relación a los diversos problemas que se presentan hoy con algunos sistemas de traducción automática de documentos catalán-castellano desde la perspectiva morfológica-sintáctica, considerando, además, las importantísimas contribuciones realizadas por las diferentes corrientes lingüísticas actuales. En síntesis, se hará una exposición de las teorías aplicadas en el *sistema prototipo de traducción automática catalán-castellano* y, finalmente, se correrá esta aplicación en desarrollo mostrando ejemplos concretos de traducción automática catalán-castellano con diferentes grados de complejidad a fin de demostrar su validez.

1 Introducción

Esta presentación está basada en el libro titulado *Catalán-Castellano: "Ejercicios de traducción automática basados en el modelo de Syntactic Structures"*, publicado en castellano por la Editorial de la Universidad Nacional de Rosario (Gustavo Alberto González Capdevila, Junio 2005).

El principal objetivo de esta ponencia es demostrar la validez de la teoría del análisis sintáctico de Noam Chomsky y de otras contemporáneas para el catalán y el castellano, partiendo del concepto de árbol y mapa sintáctico que permiten demostrar de una forma visual e intuitiva sus principales aspectos, y de las metodologías de formalización de las estructuras de una lengua desde el marco morfológico-sintáctico y considerando algunas estructuras semánticas.

Por otro lado, la gramática transformacional o de traducción automática catalán-castellano también se basa en la estructura tabular de mapa sintáctico empleada aquí, que permite visualizar la conversión de las diferentes estructuras del idioma catalán al idioma castellano.

La aplicación de estas teorías tienen como objetivo la implementación de un *sistema prototipo de traducción automática de documentos catalán-castellano*, actualmente en desarrollo y que será presentado durante esta ponencia, como consecuencia de ofrecer una modesta alternativa de solución con relación a problemas específicos que se presentan hoy con algunos sistemas de traducción automática de documentos catalán-castellano desde la perspectiva morfológica-sintáctica, considerando, además, las importantísimas contribuciones realizadas por las diferen-

tes corrientes lingüísticas actuales vinculadas con la traducción automática de documentos, especialmente en el aspecto semántico.

Es importante destacar que a pesar de las afirmaciones de los escépticos sobre la no viabilidad de la traducción automática en base a los desalentadores resultados obtenidos por algunos traductores automáticos, es conveniente indicar que dentro de las limitaciones existentes en la formalización de las estructuras sintácticas y semánticas de una lengua, es posible obtener resultados positivos dentro de una frontera específica o controlada.

En síntesis, se hará una exposición de las teorías aplicadas en el *sistema prototipo de traducción automática de documentos catalán-castellano* y, finalmente, se correrá esta aplicación en desarrollo mostrando ejemplos concretos de traducción automática catalán-castellano con diferentes grados de complejidad a fin de demostrar su validez dentro de la frontera establecida.

ID	SIGNIFICADO	ID	SIGNIFICADO
ADJ	adjetivo	FVS	frase verbal simple
ADJDEM	adjetivo demostrativo	GER	gerundio
ADJPOS	adjetivo posesivo	INTERJ	interjección
ADV	adverbio	LOC	locución
AP	aposición	MD	modificador directo
ART	artículo	MI	modificador indirecto
CCAN	circunstancial de cantidad	O	oración o proposición

Figura 1. Abreviaturas de objetos de una proposición.

ID	SIGNIFICADO	ID	SIGNIFICADO
CCAU	circunstancial de causa	OB	oración bimembre
CCO	circunstancial de compañía	OBE	oración bimembre expresa
CF	circunstancial de fin	OBT	oración bimembre tácita
CL	circunstancial de lugar	OD	objeto directo
CM	circunstancial de modo	OI	objeto indirecto
CT	circunstancial de tiempo	OU	oración unimembre
CONJ	conjunción	PREP	preposición
COPR	contracción o preposición	PRONDEB	pronombre débil
EPR	expresión preposicional	PRONIND	pronombre indefinido
FN	frase nominal	PRONPOS	pronombre posesivo
FNC	frase nominal compuesta	PRONREL	pronombre relativo
FNS	frase nominal simple	SC	sustantivo común
FU	frase unimembre	SP	sustantivo propio
FV	frase verbal	V	verbo
FVC	frase verbal compuesta	VEV	verbo o expresión verbal

Figura 2. Abreviaturas de objetos de una proposición.

2 Representación general

A continuación, se detalla la representación general correspondiente a las lenguas catalana y castellana del *sistema prototipo de traducción automática de documentos catalán-castellano*:

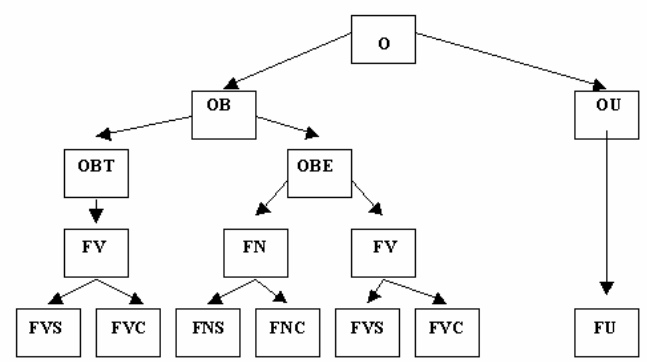


Figura 3. Representación general de la lengua catalana.

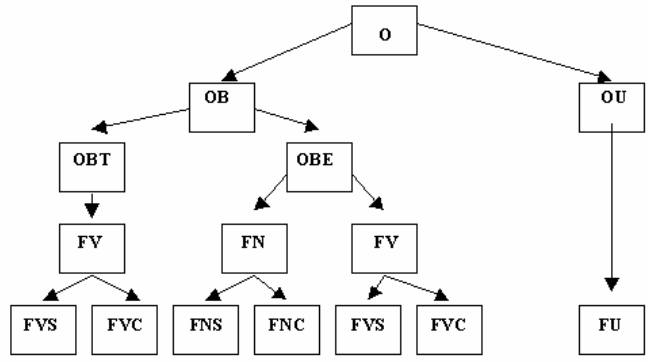


Figura 4. Representación general de la lengua castellana.

3 Reglas de concordancia

Las *reglas de concordancia* indican la relación existente entre distintos componentes de una oración. A continuación se detallan las *reglas de concordancia* soportadas por el *sistema prototipo de traducción de documentos catalán-castellano*:

3.1 Reglas de concordancia para género y número

Este tipo de regla analiza la relación existente entre los distintos componentes de la oración desde el punto de vista del *género* y el *número*. Estos componentes son los siguientes:

- adjetivo
- adjetivo posesivo
- adjetivo demostrativo
- artículo
- pronombre posesivo
- pronombre demostrativo
- sustantivo o nombre, etc.

A continuación se citan algunos ejemplos clásicos y sus correspondientes atributos asociados:

LENGUA	EJEMPLO
Castellano	*a. Jorge analiza* ***un libro*** *con* ***sus hermanos****.*
	b. El hombre escribe ***una bella poesía****.*
	c. Trabajé todo ***el domingo****.*
Catalán	*a. Jordi analitza* ***un llibre*** *amb* ***els seus germans****.*
	b. L'home escriu ***una bella poesia****.*
	c. Vaig treballar tot ***el diumenge****.*

Figura 5. Ejemplos clásicos de oraciones para el castellano y el catalán.

LENGUA	ID	COMPONENTE	CLASE	GÉNERO	NÚMERO
Castellano	*a.*	*un*	*ART*	*Masculino*	*Singular*
		libro	*SC*	*Masculino*	*Singular*
		sus	*ADJPOS*	*Masc./Fem.*	*Plural*
		hermanos	*SC*	*Masculino*	*Plural*
	b.	*una*	*ART*	*Femenino*	*Singular*
		bella	*ADJ*	*Femenino*	*Singular*
		poesía	*SC*	*Femenino*	*Singular*
	c.	*el*	*ART*	*Masculino*	*Singular*
		domingo	*SC*	*Masculino*	*Singular*
Catalán	*a.*	*un*	*ART*	*Masculino*	*Singular*
		llibre	*SC*	*Masculino*	*Singular*
		els	*ART*	*Masculino*	*Plural*
		seus	*ADJPOS*	*Masculino*	*Plural*
		germans	*SC*	*Masculino*	*Plural*
	b.	*una*	*ART*	*Femenino*	*Singular*
		bella	*ADJ*	*Femenino*	*Singular*
		poesia	*SC*	*Femenino*	*Singular*
		el	*ART*	*Masculino*	*Singular*
		diumenge	*SC*	*Masculino*	*Singular*

Figura 6. Reglas de concordancia de los ejemplos citados en la Figura 5.

De la tabla anterior de propiedades se puede indicar que todos los objetos involucrados en la estructura sintáctica, como por ejemplo: *una bella poesia*, deben tener idéntico género y número, de lo contrario, se refleja claramente un *error sintáctico* ya sea por diferir el género y/o el número de alguno/s componente/s.

3.2 Reglas de concordancia para verbos

Este tipo de reglas analiza la relación existente entre los distintos componentes de la *frase nominal* respecto al *verbo* o *expresión verbal* asociada. Estos componentes son los siguientes:

- pronombre personal
- pronombre demostrativo
- sustantivo o nombre, etc.

A continuación se citan algunos ejemplos clásicos y sus correspondientes atributos asociados:

LENGUA	EJEMPLO
Castellano	*a.* ***María e Inés leen*** *un libro con sus amigas.*
	b. ***Mis monedas son*** *del siglo XII.*
	c. ***La sábana es*** *blanca.*
	d. ***Yo vivo*** *en Gerona.*
	e. ***Ellos trabajan*** *mucho.*
Catalán	*a.* ***Maria i Agnès llegeixen*** *un llibre amb les seves amigues.*
	b. ***Les meves monedes són*** *del segle XII.*
	c. ***El llençol és*** *blanc.*
	d. ***Jo visc*** *a Girona.*
	e. ***Ells treballen*** *molt.*

Figura 7. Ejemplos clásicos de oraciones para el castellano y el catalán.

LENGUA	ID	COMPONENTE	CLASE	NÚMERO
Castellano	*a.*	*María*	*SP*	*Singular*
		e	*CONJ*	*-*
		Inés	*SP*	*Singular*
		leen	*V*	*3ª persona plural*
	b.	*mis*	*ADJPOS*	*Plural*
		monedas	*SC*	*Plural*
		son	*V*	*3ª persona plural*
	c.	*la*	*ART*	*Singular*
		sábana	*SC*	*Singular*
		es	*V*	*3ª persona singular*
	d.	*yo*	*PRONPER*	*1ª persona singular*
		vivo	*V*	*1ª persona singular*
	e.	*ellos*	*PRONPER*	*3ª persona plural*
		trabajan	*V*	*3ª persona plural*
Catalán	*a.*	*Maria*	*SP*	*Singular*
		i	*CONJ*	*-*
		Agnès	*SP*	*Singular*
		llegeixen	*V*	*3ª persona plural*
	b.	*les*	*ART*	*Plural*
		meves	*ADJPOS*	*Plural*
		monedes	*SC*	*Plural*
		són	*V*	*3ª persona plural*

Figura 8. Reglas de concordancia de los ejemplos citados en la Figura 7.

LENGUA	ID	COMPONENTE	CLASE	NÚMERO
Catalán	*c.*	*el*	*ART*	*Singular*
		llençol	*SC*	*Singular*
		és	*V*	*3ª persona singular*
	d.	*jo*	*PRONPER*	*1ª persona singular*
		visc	*V*	*1ª persona singular*
	e.	*ells*	*PRONPER*	*3ª persona plural*
		treballen	*V*	*3ª persona plural*

Figura 9. Reglas de concordancia de los ejemplos citados en la Figura 7.

De la tabla de propiedades de componentes se puede indicar que todos los *núcleos* o *nombres* involucrados en la *frase nominal*, como por ejemplo, *llençol* y *monedes* tienen asociado una conjugación verbal que corresponde al *número* de *núcleos* existentes en la *frase nominal* o *suje-*

to si este núcleo es un *sustantivo* o *nombre*. Si es un *pronombre personal* como *nosaltres*, la conjugación será la correspondiente a esa persona. La siguiente tabla muestra la regla sintáctica entre la *frase nominal* y el *verbo*:

FRASE NOMINAL		TIEMPO VERBAL	EJEMPLO
Cant. de Núcleos	Clase		
1	*SC*	*3° persona del singular*	*El **automóvil es** francés.*
			*El **cotxe és** francès.*
	SP	*3° persona del singular*	***Jaime treballa** en Badalona.*
			***Jaume treballa** a Badalona.*
	PRONPER	*1° persona del singular*	***Yo tengo** diez canciones.*
			***Jo tinc** deu cançons.*
		2° persona del singular	***Tú vivirás** en la región pirenaica..*
			***Tu viuràs** a la regió pirenaica.*
		3° persona del singular	***Ella me quiere** mucho.*
			***Ella m'estima** molt.*
		1° persona del plural	***Nosotros estudiamos** en la biblioteca.*
			***Nosaltres vam estudiar** a la biblioteca.*
		2° persona del plural	***Vosotros habéis conocido** las poesías.*
			***Vosaltres heu conegut** les poesies.*
		3° persona del plural	***Ellos han sido** perseverantes.*
			***Ells han estat** perseverants.*
≥ 2	*SC*	*3° persona del plural*	***El arte y la cultura son** un patrimonio de la humanidad.*
			***L'art i la cultura són** un patrimoni de la humanitat.*
	SP	*3° persona del plural*	***Ana y Carolina harán** un viaje a Italia.*
			***Anna i Carolina faran** un viatge a l'Itàlia.*

Figura 10. Ejemplos de relación de concordancia entre la frase nominal y la frase verbal.

4 Análisis morfológico

Es el análisis más elemental del lenguaje y consiste en determinar la existencia o no de las palabras que componen una oración, por lo tanto, el *error morfológico* tiene lugar cuando una o más palabras de la proposición no se encuentran en la *base de conocimiento* asociada.

Es importante destacar que si una palabra no se encuentra en el mencionado vocabulario no significa que ésta no exista en el mundo real de los hablantes. Muchas palabras pertenecen al hablar cotidiano y, en muchos casos, no se las encuentran en las bases de conocimiento sistematizadas como los diccionarios, por ejemplo. Otras, a pesar de su uso frecuente y reconocido por las autoridades en cuestiones lingüísticas, tampoco se las encuentran en las mencionadas bases.

5 Análisis sintáctico

Luego de haber finalizado exitosamente el análisis morfológico se podrá aplicar el análisis sintáctico a las oraciones correspondientes. El análisis sintáctico es el segundo nivel de estudio del lenguaje y su función es analizar cada oración determinando si ésta satisface las reglas sintácticas definidas. Empleando el mapa sintáctico catalán (Figura 3) se analizará el siguiente caso:

Joan ha cantat una vella òpera.

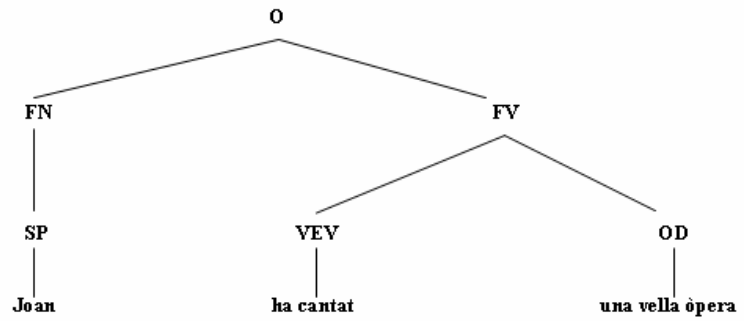


Figura 11. Mapa sintáctico de una proposición en catalán.

A continuación, se detallan los procedimientos utilizados por el *sistema prototipo de traducción automática de documentos catalán-castellano* para realizar el *análisis sintáctico*:

- *Comparar la primera palabra de la oración con las reglas sintácticas del mapa sintáctico catalán:* El objetivo de esta primera etapa es la de detectar qué tipo de oración se está analizando. Para ello, se deberá comparar la palabra **Joan** con las *reglas sintácticas catalanas* **FN**. Si satisface alguna de las reglas anteriores, se podrá afirmar que esta oración tiene **FN**. La regla que satisface esta primera etapa es **FNS31** porque esta primera palabra es un *sustantivo propio*.

COMPONENTE				
Clase	**Id**	**Estructura**	**Id**	**Subestructura**
FNS	**FNS1**	**[MD] + SC + ([MI])**	**FNS11**	**[MD1] + SC + ([MI1])**
			FNS12	**[MD3] + SC + ([MI2])**
			FNS13	**[MD3] + SC + ([MI3])**
	FNS2	**[MD] + SC ([MD])**	**FNS21**	**[MD1] + SC + ([MD2])**
			FNS22	**[MD3] + SC ([MD2] + [AP1])**
			FNS23	**[MD3] + SC ([MD2] + [AP2])**
	FNS3	**([MD]) + SP**	**FNS31**	**([MD1]) + SP**

Figura 12. Validación del primer significante con las reglas definidas para el catalán.

- *Comparar la segunda palabra de la oración:* Hasta este momento se sabe que la oración tiene **FN**. La segunda palabra no pertenece a la **FN** debido a que la regla **FNS31** no admite un componente adyacente a **SP** y, en consecuencia, la palabra **ha** pertenece a la **FV** con seguridad, más específicamente al grupo de **FVS** por tener esta oración un solo *núcleo verbal*. Como **ha** es un *verbo*, resulta que se la encuentra en todos los casos de este grupo de reglas.

COMPONENTE				
Clase	**Id**	**Estructura**	**Id**	**Subestructura**
FVS	**FVS1**	**[VEV] + [OD]**	**FVS11**	**[VEV] + [OD]**
			FVS12	**[VEV] + [OD] + ([OI])**
			FVS13	**[VEV] + [OD] + ([CL])**
	FVS2	**[VEV] + [OI] + [OD]**	**FVS21**	**[VEV] + [OI] + [OD]**
	FVS3	**(PRONREL o CONJ) + [VEV] + [CL]**	**FVS31**	**(PRONREL o CONJ) + [VEV] + [CL]**
			FVS32	**(PRONREL o CONJ) + [VEV] + [CL] + ([CT])**
			FVS33	**(PRONREL o CONJ) + [VEV] + [CL] + ([CM])**
	FVS4	**[VEV] + [CT]**	**FVS41**	**[VEV] + [CT] + ([CL])**
			FVS42	**[VEV] + [CT] + ([CCAN])**
			FVS43	**[VEV] + [CT] + ([CCO])**
	FVS5	**[VEV] + [CM]**	**FVS51**	**[VEV] + [CM] + ([CL])**
			FVS52	**[VEV] + [CM] + ([CT])**
	FVS6	**[VEV] + [CF]**	**FVS61**	**[VEV] + [CF]**
	FVS7	**[VEV] + [CCAU]**	**FVS71**	**[VEV] + [CCAU]**

Figura 13. Validación del segundo significante con las reglas definidas para el catalán.

El primer componente de esta regla es **VEV** (*verbo o expresión verbal*) y, a su vez, la palabra **ha** es una conjugación del verbo *haver* y, por consiguiente, las reglas **VEV1** son las que se tendrán en cuenta en la siguiente etapa.

- *Comparar la tercera palabra de la oración:* En esta tercera etapa se deberá comparar la tercera palabra con alguna de las *reglas sintácticas* del grupo **VEV** debido a que la palabra anterior es parte componente de alguna de las mencionadas reglas. Luego del análisis se determina que la palabra **cantat** es el participio del verbo *cantar* y, en consecuencia, satisface la regla **VEV11**.

COMPONENTE				
Clase	**Id**	**Estructura**	**Id**	**Subestructura**
VEV	**VEV1**	**HAVER**	**VEV11**	**HAVER + PARTICIPIO**
			VEV12	**HAVER + (ÉSSER) + PARTICIPIO**
			VEV13	**HAVER + ESTAR + GER**
			VEV14	**HAVER + DE + INFINITIVO**

Figura 14. Validación del tercer significante con las reglas definidas para el catalán.

- *Comparar la cuarta palabra de la oración:* En esta cuarta etapa se deberá comparar la cuarta palabra con alguna de las *reglas sintácticas* del grupo **FVS**. Esta cuarta palabra ya no pertenece a la regla **VEV11**, en consecuencia, se deberá continuar evaluando las reglas **FVS** para establecer si esta cuarta palabra pertenece a un **OD** o no. Analizando primeramente los **OD**, la palabra **una** satisface el conjunto de reglas **OD1** por ser un *artículo* y, por lo tanto, la oración corresponde al primer componente del grupo de reglas **FVS1**.

COMPONENTE				
Clase	**Id**	**Estructura**	**Id**	**Subestructura**
OD	**OD1**	**ART**	**OD11**	**ART + SC**
			OD12	**ART + ADJ + SC**
			OD13	**ART + SC + ADJ**
			OD14	**ART + SC + PRONREL o CONJ o COPR + [VEV] + (PREP + ART + ADJPOS + SC)**
			OD15	**ART + SC + PREP + SC + (PREP + SC)**
	OD2	**SC**	**OD21**	**SC**
			OD22	**[OD21] + CONJ + [OD21]**
	OD3	**ADJDEM**	**OD31**	**ADJDEM + ADJ + SC**
	OD3	**ADJDEM**	**OD32**	**ADJDEM + SC + PRONREL o CONJ + ADV o CONJ + [VEV] + PARTICIPIO + ADV**
	OD4	**PRONREL o CONJ**	**OD41**	**PRONREL o CONJ + ART + ADJPOS + SC + [VEV]**
	OD5	**INTERJ**	**OD51**	**INTERJ**
	OD6	**[OD12] + CONJ**	**OD61**	**[OD12] + CONJ + [OD12]**
			OD62	**[OD12] + CONJ + [OD14]**
		[OD13] + CONJ	**OD63**	**[OD13] + CONJ + [OD15]**
			OD64	**[OD13] + CONJ + [OD13]**

Figura 15. Validación del cuarto significante con las reglas definidas para el catalán.

- *Comparar la quinta palabra de la oración:* En esta etapa se deberá comparar la quinta palabra con alguna de las *reglas sintácticas* del grupo **OD1**. Validando la palabra **vella** con cada una de las reglas del mencionado grupo, resulta que pertenece a la subclase **OD12**.

COMPONENTE				
Clase	**Id**	**Estructura**	**Id**	**Subestructura**
OD	**OD1**	**ART**	**OD11**	**ART + SC**
			OD12	**ART + ADJ + SC**
			OD13	**ART + SC + ADJ**
			OD14	**ART + SC + PRONREL o CONJ o COPR + [VEV] + (PREP + ART + ADJPOS + SC)**
			OD15	**ART + SC + PREP + SC + (PREP + SC)**

Figura 16. Validación del quinto significante con las reglas definidas para el catalán.

- *Comparar la sexta palabra de la oración:* La palabra **òpera**, que es un *sustantivo común*, confirma la validez del *objeto directo* de la oración concluyendo, en consecuencia, que el objeto directo asociado es el de la regla **OD12**.

6 Traducción automática catalán - castellano

A continuación, se visualizan diferentes instancias de traducción (Figuras 17 y 18) pertenecientes al *sistema prototipo de traducción automática de documentos catalán-castellano* "*Castelunya 2005"*, empleando un ejemplo real en formato AVI correspondiente al concierto de Joan Manuel Serrat titulado *D'un temps, d'un país*, transmitido en directo por Radio Televisión Española en 1996.

Figura 17. Ejecución del sistema prototipo de traducción automática de documentos catalán-castellano "Castelunya 2005".

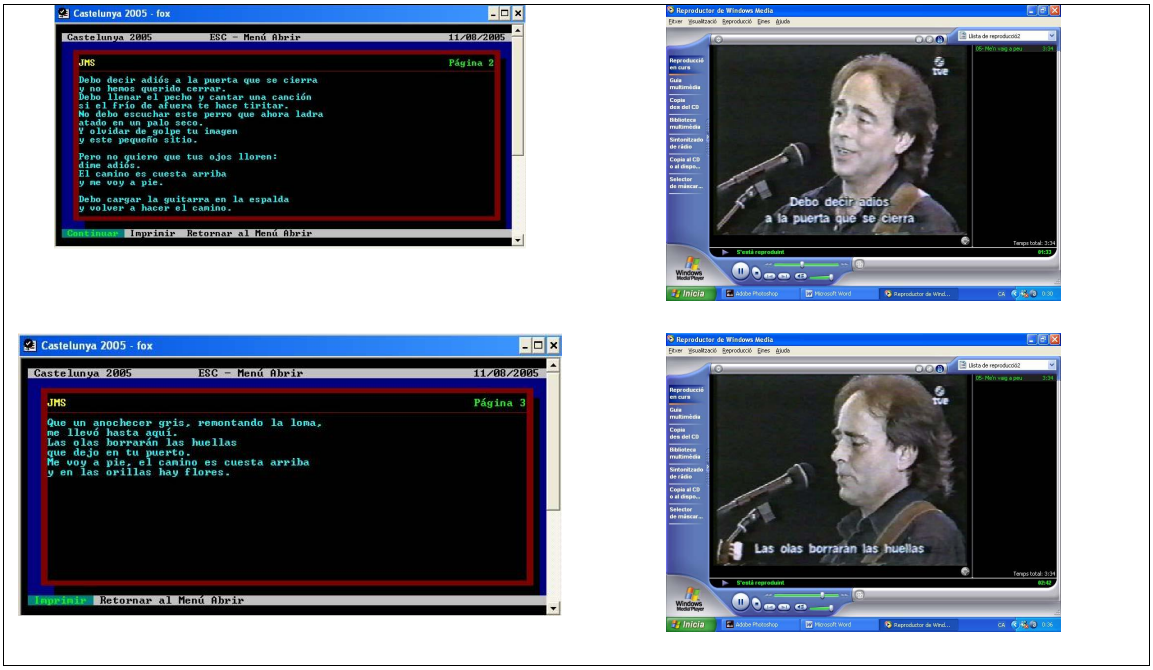


Figura 18. Ejecución del sistema prototipo de traducción automática de documentos catalán-castellano "Castelunya 2005".

7 Conclusión

Este artículo ha pretendido presentar una implementación informática en desarrollo acompañada de una síntesis de su fundamentación teórica, con el objetivo de proponer soluciones aproximadas frente a diversos problemas de traducción automática de documentos catalán-castellano, indicando sus alcances, limitaciones y proyección futura.

Si bien este artículo contempla sólo los aspectos más trascendentes de las teorías aplicadas, se podrá obtener mayor información consultando la bibliografía detallada abajo.

Referencias

Gustavo A. González Capdevila. 2005. *Catalán-Castellano: Ejercicios de traducción automática basados en el modelo de Syntactic Structures*. Editorial de la Universidad Nacional de Rosario. Rosario. Argentina. Publicación en catalán: *Català-Castellà: Exercicis de traducció automàtica basats en el model de Syntactic Structures*. 2005. Edición del autor. Rosario. Argentina.

Capítulo 6

LINGÜÍSTICA COMPUTACIONAL: DEL PROTOTIPO A LA APLICACIÓN

Daniel E. Guillot

En Víctor M. Castel, Comp. (2005) *Desarrollo, implementación y utilización de modelos para el procesamiento automático de textos*. Mendoza: Editorial de la Facultad de Filosofía y Letras, UNCuyo: 59-67.
ISBN 987-575-019-0 del soporte Internet

Lingüística computacional: del prototipo a la aplicación

Daniel E. Guillot

Cedia Consultora
Mendoza, Argentina
dangui@lanet.com.ar

Resumen

Ciertos tipos de programas de computadoras se propondrán para el tratamiento informático de las lenguas**:** El programa *SMORPH* y el software *xfst*, que presentaremos como aplicación industrial. El modelo de análisis propuesto por Bès (2002) es utilizado para categorizar el status epistemológico de las ciencias del lenguaje. Nos aporta tres elementos presentes en toda ciencia empírica: la *observación*, en la que las expresiones constituyen la base necesaria de la construcción lingüística, *el sistema de hipótesis*, que constituye el mayor campo para el desarrollo de programas tipo prototipo ya que los mismos deberán hacer la mayor parte del trabajo deductivo sobre la expresiones, y, finalmente, la *corroboración*. Dentro de este análisis ubicaremos la función de los programas de computadoras y su posible interpretación. Esta relación entre lenguaje y tecnología nos lleva a una interpretación más amplia expuesta por Auroux (1998) en la que los programas informáticos son la inteligencia. La única inteligencia real es la artificial, producto del uso social y cultural en la que se practica.

1 Introducción

Nuestro tema apuntará a cierto tipo de programas de computadora que se utilizan en el tratamiento informático de las lenguas. Necesitaremos precisar ese contexto de lingüística algorítmica en el que se produce la confluencia de disciplinas específicas como las ciencias del lenguaje y la máquina de cálculo que implican estos programas. Para categorizar el status epistemológico de estas ciencias del lenguaje utilizaremos el modelo de análisis propuesto por Bès (2002) con la forma de grilla analítica. En los segmentos establecidos por este análisis situaremos la función de los programas informáticos y su posible interpretación. Por otra parte, esta relación entre lengua y tecnología nos lleva a un horizonte más amplio en el que cabe preguntarse por una interpretación más general como la expuesta por Auroux (1998) en la que la versión algorítmica llega a ser inteligencia a secas.

La postura que considera predominante, es decir, el racionalismo contemporáneo, tiene las siguientes tesis esenciales:

"(i) Hay una distinción de naturaleza entre los *datos* (datos sensibles – lo que Russell denominaba los *sense data,* y el empirismo clásico la *sensation* – informaciones, elementos memorizados, etc.) y *los procedimientos de tratamiento* de estos mismos datos.
(ii) El fenómeno cognitivo (su funcionamiento, como su explicación) apunta al individuo, o aún, el conocimiento está localizado en el individuo" (Auroux 1998: 5)[1]

Se postula la diferencia esencial entre el dato e interpretación o tratamiento de la información. Por otra parte, el peso del fenómeno cognitivo permanece en el interior del individuo. Como contrapartida Auroux propone:

[1]Las traducciones son nuestras en todos los casos.

"La hipótesis contraria reposa sobre la existencia de estructuras cognitivas externas al individuo. Estas estructuras han conocido su desarrollo gracias a la tecnología intelectual de la escritura; dependen igualmente de instrumentos externos (libros, bibliotecas, instrumentos de cálculo y de observación, etc.) y también de las estructuras sociales de producción y de acumulación de conocimientos (enciclopedias, sociedades eruditas, redes culturales de producción y reproducción del saber). El proceso cognitivo depende de la estructuración social como depende de ella la producción de la riqueza … Un individuo aislado no podría ser inteligente, no simplemente que su inteligencia no podría desarrollarse (lo que es una trivialidad), sino fundamentalmente porque no tendría acceso a la maquinaria de la inteligencia" (Auroux 1998: 6-7).

La inteligencia es esencialmente artificial y el espíritu es antes que nada histórico y empírico. No hay absoluto, y la computadora es un caso más de la actividad humana de fabricar utensilios.

En un contexto de realismo epistemológico formulado por Auroux y también apoyado e*n general* por Bès, el *status* que buscamos no será un conjunto de normas que indiquen qué deben ser los programas de computadoras en el contexto lingüístico, sino que adoptará de arranque, una modalidad descriptiva a partir del uso histórico y social en el que se practica y no en la proximidad de tipos ideales.

2 Una grilla de análisis

La grilla propuesta se presenta como instrumento de análisis para tipificar con mayor precisión trabajos de formalización o tratamiento informático de las lenguas.

"Nuestra grilla introduce precisiones sobre los dominios de la observación, las hipótesis y la puesta en relación de las dos. En paralelo, hay tres objetos básicos: expresiones en soporte magnético asociadas a observaciones, un sistema de hipótesis que suponemos informatizado o informatizable, un *test* de corroboración, que va a poner en relación los resultados deducibles del sistema de hipótesis con las expresiones asociadas a las observaciones" (Bès 2002: 60).

Estos tres niveles conforman la estructura básica de las ciencias de la experiencia. Aquí se aplican a la lingüística en particular.

2.1 La observación

En el dominio de la observación, las expresiones constituyen la base necesaria de la construcción lingüística, se presentan como secuencias de códigos de caracteres pertenecientes a estándares informáticos (ASCII o UNICODE, etc.). Estos códigos denominados habitualmente *page code* implican una elección que condiciona el trabajo lingüístico posterior. El código ASCII simple, sólo contendrá 128 caracteres y por lo tanto no soportará acentos, la *Ñ* y otros inconvenientes que irá descubriendo el investigador a medida que avance en la informatización. En el otro extremo, el UNICODE histórico soporta 65536 caracteres, ya que construye con dos *bytes* y contiene cómodamente diferentes alfabetos y lenguas orientales como el chino o el japonés. Como contrapartida, el procesamiento de estas cadenas doble *byte* no puede ser hecho con las rutinas más universales y usadas de lecto-escritura, como el algoritmo de lectura izquierda-derecha, implicando, por lo tanto, un costo adicional de programación. Estas consideraciones no tienen un valor permanente: se modifican dinámicamente dependiendo de muchos factores. Cuando la comunidad informática se ve obligada a aceptar abruptamente que el ASCII de 128 caracteres no puede contener la variedad de simbología que se comienza a usar, las entidades como los comités de estándares (ISO, ANSI) y las grandes compañías de computación (*Microsoft*, *Apple*, IBM) generan las diferentes propuestas de *ASCII ampliado* a 256 caracteres. Por ejemplo para las lenguas europeas occidentales aparece el ISO 8859-1, el *code page* 850 de IBM, ANSI y *Microsoft* lanzan el *Windows* 1252 y *Apple* el *Macintosh Roman*. Estos conjuntos de caracteres difieren entre sí particularmente en las nuevas asignaciones que se agregan al ASCII original. Tomemos como ejemplo el caso del carácter *Ñ*

que se representa como ‘209’ en ISO 8859-1 y Windows 1252. En IBM 850 es ‘165’ y en *Macintosh Roman*, ‘132’.

Hacia 1986 se comienza a trabajar en un *UNICODE Standard* cuya primera versión aparecerá en 1992. La página técnica de *Internet* explica: "… UNICODE provides a consistent way of encoding multilingual plain text and brings order to a chaotic state of affairs that has made it difficult to exchange text files internationally".

Esta propuesta consigue con el tiempo el apoyo de las grandes compañías como *Apple*, HP, IBM *JustSystem*, *Microsoft*, *Oracle*, Sap, Sun, Sybase, etc.

El *UNICODE Standard* define tres formas de codificación que permiten que el mismo dato pueda ser transmitido en 8, 16 y 32 bits por unidad de código. El UTF-8 (*UNICODE Transformation Format*) transforma todos los caracteres *UNICODE* en un código de *bytes* de longitud variable. Tiene la ventaja de mantener el ASCII familiar y que puede ser usado sin excesiva reescritura de software. El UTF-16 es más razonablemente compacto en un uso económico del almacenamiento. Finalmente, UTF-32 es útil cuando el espacio ocupado de memoria no es importante, pero la longitud fija de los caracteres es una ventaja deseable. La versión 4.0 soporta 96.447 caracteres incluyendo los diferentes alfabetos europeos, la escritura del Oriente Medio de derecha a izquierda y la asiática, también incluye signos de puntuación, símbolos matemáticos, técnicos, flechas, etc.

La situación actual del *UNICODE Standard* es la de un proyecto óptimo pero aún lejos de la realidad, sobre todo, para el software “viejo”.

En el caso del programa *xfst* de análisis morfológico que veremos más adelante, el texto sobre *UNICODE* es significativo:

> “The Xerox finite-state networks have been defined to accommodate 16-bit UNICODE characters, wich would have distinct advantages in many applications. However, the dificulties of UNICODE text-editing, display and printing make UNICODE processing awkward at this time” (Beesley y Kartunen 2003: xiii).

Volviendo al carácter de dato que se le asigna a la *expresión* en esta grilla, hemos visto que los procedimientos de tratamiento en el nivel más bajo ya están incorporados al dato. El código elegido para soportar la expresión observacional ya está llevando implícitos procedimientos interpretativos, plataformas selectivas y hasta marcas de *hardware* computacional. La intervención de *software* en este nivel se limita a la adopción de un *page code* y los *drives* respectivos que influyen a nivel de teclado, pantalla e impresión. Los *drives* de pantalla e impresión resuelven la selección de la representación gráfica del código elegido.

La primera tesis del racionalismo mencionada por Auroux es la que se presenta como particularmente interesante en el análisis de la primera fase de la grilla de Bès. El dato de la observación para el racionalismo vigente está deslindado de la interpretación posible. Para Bès, este dato está ligado a un Observador que consagra la función interpretativa y ciertas aptitudes que lo hacen un observador válido. Deberá reconocer en forma explícita “que no hay una sola forma de observar” (Bès 2002: 61).

Para Auroux es el advenimiento de la informática el que plantea por primera vez la necesidad de ligar el dato y el procesamiento del dato en un solo movimiento. Estas discusiones abren un viejo debate de la filosofía tradicional: la oposición de materia y forma del conocimiento, el léxico y la sintaxis en el lenguaje, percepción *versus* las ideas innatas, lo accidental frente a lo eterno. Condillac es presentado como “el primero en cuestionar radicalmente la dicotomía sobre la que reposa el racionalismo” (Auroux 1998: 6). La capacidad de cálculo no es una facultad misteriosa, sino aprendida trabajosamente en la historia. “No sabríamos recordar la ignorancia, en la que hemos nacido”, nos dirá Condillac al comienzo del *Tratado de las sensaciones*.

2.2 Sistema de hipótesis

El sistema de hipótesis propuesto por Bès para evaluar una lingüística empírica ofrece el mayor campo para el desarrollo de programas tipo prototipo ya que ellos deberán hacer la mayor parte del trabajo deductivo sobre las expresiones de entrada vistas en el punto anterior.

"Suponemos que nuestro sistema de hipótesis está informatizado o es informatizable, que las expresiones deducidas son obtenidas por una máquina algorítmica y que tienen la forma <*expr., inf.*> en la que *expr*. es siempre una secuencia de códigos, y que *inf.* denota las informaciones asociadas por la máquina a *expr.* a partir de SH (Sistema de hipótesis)" (Bès 2002: 65).

El primer elemento de un sistema de hipótesis es para Bès el tipo de formalismo utilizado. Si el sistema de hipótesis esta formalizado será gracias al cálculo del formalismo que alcanzaremos las consecuencias, es decir, las expresiones deducidas. El programa informático en este caso debe ejecutar el cálculo aplicando las formulas para obtener las expresiones deducidas. De aquí se concluye: "La máquina no inventa nada. En general, una teoría formalizada es completamente independiente de la máquina que la calcula" (Bès 2002: 65). Es decir, que en el caso ideal de un sistema completamente formalizado, la computadora es una simple máquina de cálculo y el programa es fácilmente generalizable. Esto no le quita necesariamente su carácter de prototipo ya que puede mantenerse en un nivel de uso *ad hoc* sin participar de ciertas pautas de universalización informática.

Pero debemos preguntarnos qué pasa cuando el sistema de hipótesis no está completamente formalizado o no lo está a secas.

"Un programa informático está, ciertamente, siempre constituido por fórmulas. ¿Pero, se puede especificar lo que hace este programa y probar que hace lo que la especificación exige que haga? Si la respuesta es sí, se dirá que el programa calcula efectivamente un SH formal. Si el programa obtiene resultados asociando entradas y salidas, pero no se puede conocer los resultados obtenidos más que ejecutándolo, o, en el mejor de los casos, examinando su código y suponiendo intuitivamente lo que debería producir a partir del código y de los comentarios que el informático les habrá querido asociar, se dirá que tenemos un SH no formalizado puesto en máquina. En este caso, es el algoritmo de cálculo efectivo de expresiones deducidas el que va, en gran medida, a determinas éstas" (Bès 2002: 65).

La relación entre el grado de terminación que tenga el sistema formal y el uso que se haga de la computadora permitirán clasificar las diferentes situaciones. Un sistema en el que el formalismo de cálculo se distingue netamente del programa nos dará un *sistema de cálculo formal.* Por otra parte, si el formalismo de cálculo y el programa están fusionados hablaremos de *sistema de cálculo algorítmico.* Relacionando el sistema de cálculo (formal o algorítmico) con el sistema de hipótesis, diremos que es subyacente. Al aplicar esta distinción a gramáticas en tanto que sistemas de hipótesis, los modelos de gramáticas funcionarán como cálculo formal subyacente. De este modo, se prefiere a nivel de sistema de hipótesis la modalidad declarativa de las descripciones lingüísticas, sin mezclar exigencias de procedimientos (algorítmicas). Finalmente, haciendo un balance de la relación formalismo y procedimiento algorítmico, Bès concluye:

"Para caracterizar el abanico de nuestras SH con relación al tipo de formalismo utilizado, podemos fijar dos polos con posibles situaciones intermedias. Tenemos, por un lado, las teorías formalizadas, que en la especificación de las expresiones deducidas no dependen del útil informático que las obtiene efectivamente. Por otro lado, cuando la especificación de las expresiones deducidas resulta de dos factores: las fórmulas, en tanto que hipótesis, apuntan a caracterizar el dominio observacional vía expresiones deducidas por una parte, y las operaciones efectivas que permiten obtener las expresiones deducidas por otra parte, los dos factores están de tal modo imbricados que es imposible distinguir lo que viene de cada uno de ellos. Los objetos del primer polo son *SH formales*, con cálculos formales

subyacentes; los objetos del segundo polo son *SH algorítmicos*, con cálculos algorítmicos subyacentes. En los dos casos, suponemos, que se trata de SH informatizados o informatizables" (Bès 2002: 66).

2.3 Conclusión

Vemos que en el sistema de hipótesis el programa de computadora interviene en el nivel del cálculo deductivo, siendo en este caso subyacente al sistema de hipótesis y en el caso de que una gramática constituya el sistema de hipótesis, es subyacente a la gramática.

Es en el nivel de cálculo deductivo en el que hay que distinguir **cálculo formal** y **cálculo algorítmico** (programa). El cálculo algorítmico incluye especificaciones de procedimientos que no tienen la transparencia deductiva del cálculo formal. Este hecho produce una zona de ambigüedad en la lógica algorítmica que no tienen equivalencia en el cálculo puramente formal.

Estos dos modos de cálculo pueden funcionar en una concordancia perfecta o puede haber zonas en las que predomine la discordancia. Así, llegamos a una cierta descalificación epistemológica de los útiles informáticos que no presentan un nivel de formalización explícito.

2.4 *Test* de corroboración

El *test* de corroboración propuesto por Bès compara las expresiones observadas en el *dominio de la observación* con la expresión informada por la cadena deductiva del sistema de hipótesis. El resultado de esta comparación puede ser adecuado, inadecuado, o también, pueden presentarse situaciones escalares y hasta delç tipo indeterminado (Bès 2002: 68).

Los conceptos expuestos para la programación informática se mantienen aquí con los cálculos formales o algorítmicos.

3 El prototipo

El programa *SMORPH* (Segmentación y Morfología), en su versión original, presenta las características de un prototipo de realización universitaria de *software*, pero que puede calcular sobre volumen de información y de textos en condiciones de utilización no restringida. Y que ejecuta un tratamiento algorítmico de textos en lengua natural. *SMORPH* es presentado como útil de validación (Aït-Mokhtar 1998) del análisis presintáctico automático de textos escritos. Su autor, Salah Aït-Mokhtar, lo describe así:

"*SMORPH* es un utilitario que agrupa un conjunto de funcionalidades en torno a la morfología: compilación de diccionario (transformación de un diccionario fuente, es decir, una descripción morfológica legible de un conjunto de palabras, en una representación interna utilizable para el análisis y/o generación morfológica), análisis y generación morfológicas, segmentación y lematización de textos. Puede ser considerado como un elemento de un sistema de tratamiento lingüístico más amplio en el que su papel principal sería asegurar la primera parte del tratamiento: convertir los archivos de textos ASCII a secuencias de ítems significativos adecuados como entrada del análisis sintáctico. En este cuadro, *SMORPH* permite construir diccionarios electrónicos voluminosos necesarios para un análisis lingüístico de textos. Estos diccionarios podrán utilizarse en generación o en segmentación y análisis" (Aït-Mokhtar 1998: 125).

El diccionario es realizado a través de un autómata de estados finitos que es la misma técnica utilizada por Xerox (cf. más adelante) para la construcción de su analizador morfológico.

Los archivos fuentes para el diccionario usan editores de textos planos que describen definiciones tipográficas sobre los caracteres, terminaciones, modelos de flexión y entradas léxicas.

Las definiciones ASCII sobre la información tipográfica indican las clases de separadores, espacios y potenciales tipográficos. Estos últimos indican el conjunto de representaciones que puede asumir un carácter en ese tipo de texto, por ejemplo de *A* podemos declarar *a*, *à*, *á, Á, À* como representaciones posibles. Las terminaciones serán utilizadas en los modelos de flexión. Los rasgos serán utilizados en las definiciones morfosintácticas asociadas a las formas. También

es posible declarar para cada rasgo el conjunto de sus valores posibles. Cada modelo de flexión describe las flexiones posibles de las distintas categorías, como así también los rasgos morfológicos correspondientes.

> "El compilador de diccionarios lee las entradas léxicas del diccionario fuente, calcula las formas flexivas según los modelos de flexión de entradas y los organiza con su información tipográfica y morfológica en un autómata de estados finitos determinista. A continuación, el autómata así obtenido es minimizado (reagrupando los estados equivalentes) y compactado con un algoritmo de compresión que alinea estados y transiciones en una tabla. El diccionario así minimizado y compactado es utilizado en generación o análisis y puede ser guardado en el mismo formato en disco, siendo en este caso su carga en memoria más rápida. […] El analizador presintáctico de SMORPH utiliza el diccionario compilado para segmentar y lematizar textos escritos en un solo proceso …" (Aït- Mokhtar, 1998: 94).

SMORPH está programado en C norma ANSI. El código fuente tiene cerca de 15000 líneas, y está compilado con gcc (Gnu C *Compiler*) en una estación Sparc 2 bajo el sistema operativo Solaris 2.5. El ejecutable producido está en el orden de los 150k.

4 La aplicación

El *software* que presentamos como aplicación industrial es *xfst* desarrollado por *Xerox* y usado por *Xerox Research Centre Europe* (XRCE, Grenoble, Francia) y *Palo Alto Research Center* (PARC, California, USA) y otros centros de investigación lingüística en el mundo.

La aplicación se presenta como una implementación de autómatas de estados finitos para producir análisis morfológicos y generación. Incluye *lexc*, un lenguaje declarativo de alto nivel usado para especificar lexicones de lengua natural. La sintaxis de este lenguaje ha sido diseñada para facilitar la definición de la estructura morfotáctica, tratamiento de grandes irregularidades y miles de formas base de una lengua natural. Los archivos fuentes declarativos pueden introducirse con un editor de textos planos como el *notepad* de *Windows* o el emacs de *Linux*.

También incluye *tokenize,* una aplicación que ejecuta un autómata de estados finitos con un texto de entrada para segmentarlo en *tokens* pudiendo ejecutarse acoplado a otros programas como *lookup* que recibe *tokens* y produce análisis morfológico.

Este software tiene un valor estratégico ya que el análisis morfológico es una función básica para pasar a procesos más complejos de las lenguas naturales como el etiquetado (*tagging*) de partes del discurso, análisis sintáctico, traducción y otras aplicaciones de alto nivel.

Las versiones ejecutables incluyen compilaciones para los sistemas operativos *Solaris*, *Linux* (base *intel*), *Windows* (2000, Me, XP) y *Macintosh* OS X. Esta característica de ser multiplataforma es tal vez el rasgo más distintivo de una aplicación tipo industrial.

Una plataforma o sistema operativo es un conjunto de programas que manejan el *hardware* y los recursos de *software* de un sistema o computadora. Estos recursos incluyen el procesador, la memoria, el espacio en disco, etc. y el sistema operativo los distribuye a medida que los programas particulares lo solicitan. Por otra parte, les permite a las diversas aplicaciones conectarse a los elementos de *hardware* (impresora, lectoras. etc.) sin necesidad de conocer los detalles de funcionamiento de los que se hace cargo. Es, por decirlo de una manera gráfica, la primera capa de *software* que recubre los "fierros" de la máquina.

Los sistemas operativos en los que puede correr *xfst* son los más importantes del mercado.

Solaris es el sistema operativo de Sun para máquina Sparc y también en sistemas x64/x86, es decir, incluyendo las PCs de 32 y 64 bits. *Xfst*, de hecho, ha sido desarrollado en el sistema *Solaris*, tipo Unix.

Linux es un sistema operativo creado por un estudiante de la Universidad de Helsinki hacia 1991. La versión 1.0 en 1994 se libera bajo *GNU General Public License*, estando su código fuente disponible para cualquiera. En la actualidad se ha convertido en la principal alternativa frente a los sistemas *Windows* de *Microsoft*. Fabricantes de primera línea como IBM y Hewlett-Packard han adoptado *Linux* y dan soporte a su desarrollo.

Windows en sus últimas versiones (2000, Me, XP) es el sistema operativo de PCs más usado en el mundo y *xfst* ha hecho recientemente esta versión.

Macintosh OS X sistema operativo de *Apple* usado en las computadoras Mac cuya popularidad sigue a Intel y AMD.

También acompaña al software una publicación de más de 480 páginas, *Finite State Morfology*, (Beesly y Karttunen 2003) con un CD incluido que permite utilizar distintas versiones de los programas en un contexto no comercial; acompañando al libro la página *web*: http://www.fsmbook.com/, la que contiene:

Información sobre actualizaciones de Software
Errata
Aclaraciones
Nuevos Ejercicios y material auxiliar
Preguntas Frecuentes (FAQ)
Ejemplos que pueden bajarse de *Internet*
Anuncios de cursos y seminarios sobre programación de estados finitos
Información para la adquisición de Licencias Comerciales

5 Conclusiones

La función de la programación informática en el contexto lingüístico debe situarse esencialmente en el nivel del cálculo deductivo. El análisis epistemológico que hemos utilizado para segmentar los dominios especificables en la descripción de las ciencias del lenguaje nos permite acotar las funciones informáticas por dominio. En el dominio de la observación los datos se encuentran informatizados en el nivel de sistema de códigos accesibles a la lecto-escritura de la computadora. Esta operación, relativamente simple, permite el acceso de la expresión lingüística a la simbología informática y, en consecuencia, abre la posibilidad de tratamiento algorítmico en forma directa.

Este cálculo de expresiones codificadas exige trasladarse al dominio del sistema de hipótesis en el que el cálculo algorítmico estará al servicio de la deducción requerida desde las hipótesis lingüísticas. Aquí la programación hará efectivo el cálculo deductivo planteado a nivel formal.

Finalmente en el dominio del *test* de corroboración se utilizarán los elementos anteriores, pero orientados a la evaluación de adecuación o no entre la expresión observada y la expresión deducida.

Desde el punto de vista de la integración de los útiles informáticos a las estructuras de la producción de conocimientos y de bienes en general, hemos distinguido dos modalidades que se encuentran en los extremos de programación informática: el prototipo y la aplicación industrial.

El prototipo de programa computacional generado en el ámbito académico tiene por objetivo efectuar las deducciones correspondientes a un sistema de tesis. Predomina la función de evaluación científica y la corrección formal de los resultados. De este modo, es normal que se descuiden aspectos como la facilidad de uso del útil informático, el transporte de los programas a otras plataformas, el soporte por medio de página *web*, la literatura de ayuda y los foros en *Internet*. Estas características se encuentran en la aplicación industrial que está dirigida ya sea a la comunidad científica internacional o al mercado. El círculo de desarrollo virtuoso entre los dos es fácil de imaginar: un buen prototipo como SMORPH puede ser la base necesaria para completar una aplicación industrial

Referencias

Gabriel G. Bès. 2002. La linguistique entre science et ingénierie. En *TAL*, 3: 57-81.

Kenneth R. Beesley y Lauri Karttunen. 2003. *Finite State Morphology*. CSLI Publications, Stanford University.

Salah Aït-Mokhtar. 1998. *L'analyse présyntaxique en une seule étape.* Tesis de doctorado, Universidad Blaise-Pascal, Clermont-Ferrand.

Sylvain Auroux. 1998. *La raison, le langage et les normes*. Presses Universitaires de France.

Capítulo 7

EL SINTAGMA NOMINAL NÚCLEO

Zulema Solana y Andrea Rodrigo

En Víctor M. Castel, Comp. (2005) *Desarrollo, implementación y utilización de modelos para el procesamiento automático de textos*. Mendoza: Editorial de la Facultad de Filosofía y Letras, UNCuyo: 69-79.
ISBN 987-575-019-0 del soporte Internet

El sintagma nominal núcleo

Zulema Solana y Andrea Rodrigo

Universidad Nacional de Rosario
Facultad de Humanidades y Artes
Rosario, Argentina
zsolana@arnet.com.ar // andreafrodrigo@yahoo.com.ar

Resumen

El trabajo presenta una primera etapa de la descripción y modelización, en el marco del Paradigma 5P, del sintagma nominal núcleo en español (snn). La descripción lingüística y la formalización del snn, en nuestro planteo, obran como fuente declarativa de los algoritmos. Hay por lo tanto, independencia entre los dos planos. A partir de las categorías morfosintácticas que integran el snn vamos a establecer dónde comienza, dónde termina, cómo está constituido y cuál es su núcleo. Nos concentramos en las Propiedades (P2) de un subconjunto significativo de expresiones del snn del español, relacionándolas con elementos de observación de los Protocolos respectivos (P1). Trataremos las especificaciones lingüísticas de las categorías mediante rasgos y sus relaciones con la observación. Distinguimos dos tipos de rasgos: los morfosintácticos y los de concordancia.

1 Introducción*

Como el sintagma nominal núcleo (desde ahora snn) se comporta de modo diferente, respecto de las categorías que pueden iniciarlo, según vaya en el sujeto antepuesto al verbo, el sujeto pospuesto, el objeto o en otras posiciones, hemos planificado su descripción en tres etapas: (i) los snn en el sujeto antepuesto, (ii) los snn que están en sujetos pospuestos y en objetos directos y (iii) el resto de los snn. Este procedimiento permitirá dar cuenta de las diferencias respecto de la posibilidad del nombre de iniciar o no el snn, ya que, si está en singular, cuando es objeto de verbos transitivos puede estar al comienzo del snn (*busco tela de buena calidad*) y no así cuando es sujeto antepuesto de cualquier tipo de verbo (**hombre trabaja*). También cuando el nombre va después de preposición puede aparecer solo (*con desagrado*). Consideramos que la complejidad de las posibilidades de combinación hacen necesario un tratamiento por etapas sucesivas. En este trabajo se expone la primera etapa de la descripción y no se aborda la coordinación dentro del snn. Dejaremos algunas cuestiones sin desarrollar, porque todavía las estamos investigando.[1]

La descripción lingüística y la formalización del snn, en nuestro planteo, obran como fuente declarativa de los algoritmos. Hay por lo tanto, independencia entre los dos planos, el mismo algoritmo puede servir para trabajar distintas lenguas, y cada lengua requiere una descripción particular. En la descripción y formalización del snn nos ubicamos en el marco del paradigma 5P; cf. (Bès 99; Bès et al. 99; Coheur et al. 2000; Hagège 2000).

2 Los sintagmas núcleos: observaciones generales

*Agradecemos a Gabriel G. Bès sus comentarios y críticas a las versiones anteriores de este trabajo, del cual asumimos toda la responsabilidad. Esta presentación forma parte del proyecto de investigación INFOSUR: investigación y desarrollo de la Universidad Nacional de Rosario.

[1]No trataremos aquí: determinantes interrogativos (ejemplos: *qué cosa, cuál puerta*), correlaciones (ejemplo: *los unos .... los otros*), indefinidos relativos que pueden introducir subordinadas (ejemplos: *quienquiera, cuantos*), relativos e interrogativos que integran sintagmas nominales núcleos, como núcleos o antecediendo al núcleo (ejemplos: *¿Quién llegó*? *El hombre cuyas hijas se casaron vive lejos*). Se deja para un tratamiento posterior a las variaciones en la expresión de cardinales por ejemplo: *una docena* en lugar de *doce* y la particularidad de poder estar seguidos por partitivo, como una *docena de lápices*. Tampoco tratamos en esta etapa a los snn iniciados por adverbio (ejemplo: *Sólo Juan lloró*) ni nombres propios precedidos por artículo y seguidos por relativa (ejemplos: *El Pérez de quien te hablé*), ni los snn que incorporan dígitos (ejemplos: *los 3, el 4/5/2001*).

A partir de las categorías morfosintácticas que integran los sintagmas núcleos, es posible determinar dónde éstos comienzan, dónde terminan, cómo están compuestos y cuál es su núcleo. Las propiedades de linealidad restringen la posibilidad de combinación de sus elementos. Un sintagma núcleo es un bloque casi inseparable; ejemplos: (1) *no los he visto*; (2) *las lejanas playas*; (3) *muy hermoso.* En (1), (2) y (3) se tiene respectivamente un sintagma verbal núcleo (svn), un sintagma nominal núcleo (snn) y un sintagma adjetivo núcleo (san).

Los sintagmas núcleos son los *chunks* (Abney 1991). Abney los justifica por razones prosódicas y psicolingüísticas, por un lado, y porque permiten un análisis automático del texto con menores dificultades. A partir de Aït-Mokhtar y Chanod (1997) se han hecho investigaciones de análisis sintáctico automático utilizando la plataforma XIP que recurre al análisis en *chunks* o sintagmas núcleos (Nuria 2005).

En el marco del Paradigma 5P se considera que el análisis en sintagmas núcleos permite reducir significativamente la ambigüedad de la categorización morfo-sintáctica al concatenar las expresiones internas. En *los ha comprado* o en *comprarlos, los* no puede ser artículo. Además es un paso en la tokenización que significa un avance para el logro de la representación semántica. Los análisis realizados dentro del Paradigma 5P focalizan especialmente la estructura interna de los sintagmas núcleos, lo que no ocurre en los trabajos mencionados antes.

3 El paradigma 5P

Las 5P se declinan del siguiente modo:

• P1: P de Protocolos; un Protocolo es la representación de un dato obtenido por un Observador (explícitamente modelado).

• P2: P de Propiedades; una Propiedad es formalmente análoga a un axioma. Un conjunto finito de Propiedades especifica en intensión un conjunto de secuencias de expresiones de una lengua determinada. Un modelo es una secuencia que satisface un conjunto de Propiedades.

• P3: P de Proyecciones. Las Proyecciones son generalizaciones sobre las Propiedades o sobre un subconjunto de Propiedades de una lengua natural.

• P4: P de Principios. Un Principio es una restricción sobre las Proyecciones que son válidas para todas las lenguas o para un conjunto de ellas.

• P5: P de Procesos. Un Proceso es un procedimiento que está implantado en máquina o que puede ser implantado y con el que se pueden tratar las secuencias de las lenguas naturales.

4 El tratamiento lingüístico del snn basado en el Paradigma 5P

Nos concentramos en las Propiedades (P2) de un subconjunto significativo de expresiones del snn del español, relacionándolas con elementos de observación de los Protocolos respectivos (P1). Las Propiedades se expresan mediante categorías, que son conjuntos de etiquetas/valores (cf. sección 4.1.1.) y que se suponen asociadas a un significante gráfico (por ejemplo*, el, la, lo, los, las* son los significantes gráficos asociados a las categorías que son artículos). Las Propiedades especifican los modelos. Un modelo es un objeto formal que satisface las Propiedades. Se reconocen tres tipos de Propiedades:

-de Existencia, que expresan las categorías que pueden encontrarse en un modelo y sus relaciones de concordancia;

-de Linealidad, que expresan las relaciones de orden entre las categorías de un modelo;

-de Flechado, que expresan las relaciones entre las categorías de un modelo, a partir de las cuales será posible calcular la representación semántica del modelo.

En la sección 4.1 se tratan las categorías, en la 4.2 se presentan las Propiedades de existencia, sus aspectos formales y lingüísticos. De manera análoga en las secciones 4.3. y 4.4. se tratan las Propiedades de linealidad y de flechado.

4.1 Las categorías

A continuación se tratan las categorías, sus aspectos formales (4.1.1), sus especificaciones lingüísticas mediante rasgos (4.1.2), y sus relaciones con la observación (4.1.3).

4.1.1 Aspectos formales de las categorías

Se utiliza una noción de categoría muy próxima a la de GPSG (Gazdar et al.1985, cap. 2). Se supone así un Vocabulario de rasgos, es decir, un conjunto no vacío y finito de rasgos. Un rasgo es una etiqueta asociada a uno o más valores. Por ejemplo, la etiqueta NUMERO (o en abreviatura, NUM), posee los valores 'sg' y 'pl', la etiqueta GÉNERO (o en abreviatura, GEN), los valores 'm' y 'f'. Una categoría es un conjunto finito no vacío de valores de rasgos (o de manera abreviada, de valores). Un solo valor de cada etiqueta es admitido en cada categoría. Por convención, no se utiliza un mismo valor para dos etiquetas diferentes, de manera que se pueden describir las categorías por la sola enumeración de sus valores. En general, un rasgo con sus valores se escribe:

<ETIQUETA; {$valor_1$, ... , $valor_n$}

Sea el Vocabulario de rasgos *V* que sigue, en donde *EMS* es la abreviatura de *Etiqueta Morfo-Sintáctica*, *TDET* es la abreviatura de *Tipo de DETerminante, TNOM* es la abreviatura de *Tipo de NOMbre* y *det, n, adj, art, indf, nc, npr* anotan los valores corresponden, respectivamente, a determinante, nombre, adjetivo, artículo, indefinido, nombre común y nombre propio.

V = { <EMS; {det, n, adj}>, <TDET; {art, indf}>, <TNOM; {nc, npr}> }

Para organizar los valores en una jerarquía y expresar las relaciones de herencia se utiliza la sangría. Así, por ejemplo, mediante

```
det
    art
    indf
n
    nc
    npr
adj
```

expresamos que si en una categoría definida mediante este esquema tenemos el valor 'art', tenemos también el valor 'det'. Hemos entonces definido las categorías [det, art], [det, indf], [n, nc], [n, npr], [adj].

Dado el Vocabulario de rasgos *V* las categorías así definidas son *categorías máximas*. En general, una categoría máxima es un conjunto de valores tal que, a partir de un Vocabulario de rasgos, ningún otro valor puede agregarse. Es decir, si a *V* agregamos los rasgos de GEN y NUM y obtenemos *V'*, las categorías [det, art], [det, indf], [n, nc], [n, npr], [adj], no son máximas, ya que habría que definir, por ejemplo:

[det, art, m, pl]

categoría máxima que corresponde al significante gráfico *los.*

Las categorías son, formalmente, conjuntos de valores, y, por lo tanto, el orden de notación de los valores no es pertinente. Un conjunto A de valores subsume un conjunto B, si A es un subconjunto de B. Así, por ejemplo, [det] subsume [det, indf] o [det, f]; [f] subsume [det, art, f, sg], [art, f, pl].

Para expresar las Propiedades, la organización del Vocabulario de rasgos, de las relaciones de herencia y las categorías máximas que éstas determinan, las relaciones de subsunción, son elementos claves. Las Propiedades se expresan sobre conjuntos de valores. Van a ser válidas sobre todas las categorías subsumidas por el conjunto de valores mencionados en una Propiedad. Así, cuando se exprese que la categoría [det] precede la cat [n], se expresa que toda categoría con el valor [det] precede toda categoría con el valor [n], de tal manera que, según el ejemplo de *V'* precedente, los artículos, los indefinidos, en singular o plural, en masculino o femenino, deben preceder a los nombres, sean propios o comunes.

4.1.2 Especificación lingüística de las categorías

Distinguimos dos tipos de rasgos: los morfosintácticos y los de concordancia. Los rasgos morfosintácticos utilizados son los que siguen.[2]

<EMS; {det, n, adj, num, pron, adv}>
<TDET; {art, indf, dem, pos, tod, amb, send, cad, ciert, tant, mism}>
<TN; {nc, npr}>
<TADJ; {adj1, adj2}>
<TNUM; {num1, num2}>
<TPRON; {prpers, prindf, prdem, prpos}>
<TINDF; {indf1, indf2}
<TINDF1; {indf1a, indf1b}
<TINDF2; {indf2a, indf2b, indf2c}

Los rasgos de concordancia que usaremos son los que siguen.

<GEN; {m, f, neu, _}>

El símbolo '_' anota una variable que, intuitivamente, expresa "cualquier valor". Por ejemplo, mediante `[GEN= _] utilizado en la categoría del numeral *tres*, se expresa que éste puede usarse tanto con un masculino como con un femenino (*tres libros, tres flores*).

La combinatoria de valores de concordancia necesaria para completar la descripción de las categorías del snn es la que sigue. Por convención *ad hoc,* el primer valor del par es de GEN y el segundo de NUM; a la derecha de cada par se lo ejemplifica.

[m, sg] el, libro
[f, sg] la, flor
[_, sg] bastante, su
[neu, sg] lo

Una categoría máxima es la unión de una categoría formada por todos los rasgos morfosintácticos que resultan de aplicar la jerarquía de rasgos, con uno de los pares posibles de concordancia. Así, la categoría [det, tant] formará con los pares de concordancia categorías máximas; ejemplos:

[det, tant, f, sg] tanta
[det, tant, m, sg] tanto

[2]Abreviaturas: det (determinante), art (artículo), indf (indefinido), dem (demostrativo), pos (posesivo), tod (todo), amb (ambos), send (sendos), cad (cada), ciert (cierto), tant (tanto), mism (mismo), n (nombre), nc (nombre común), npr (nombre propio), adj (adjetivo), num (numeral), pron (pronombre), adv (adverbio), prpers (pronombre personal), prindf (pronombre indefinido), prdem (pronombre demostrativo), prpos (pronombre posesivo).

Es claro que no todas las categorías de rasgos morfosintácticos se combinan con todos los pares posibles de rasgos de concordancia.

4.1.3 Observaciones lingüísticas sobre las categorías

Los rasgos morfosintácticos se organizan en la jerarquía siguiente, que determina las relaciones de herencia y las categorías constituidas por rasgos morfosintácticos. Entre corchetes se indican los valores terminales que están, todos, en un mismo nivel.

det
    art
    indf
        indf1
            [indf1a, indf1b]
        indf2
            [indf2a, indf2b, indf2c]
    [dem, pos, tod, amb, send, cad, ciert, tant, mism]
n
    [nc, npr]
adj
    [adj1, adj2]
num
    [num1, num2]
pron
    [prpers, prindf ,prdem, prpos]
adv

Las categorías morfosintácticas así constituidas se completan mediante los valores de concordancia. Los mismos son indicados para determinantes y pronombres.

4.1.3.1 Determinantes

Los determinantes tienen en común las siguientes características:
(i) Pueden referirse a otro elemento, que es el núcleo del *snn* y se anteponen a él.
(ii) Constituyen clases cerradas.
(iii) Cuando son núcleo cumplen lo que podría llamarse *función pronominal.*

	Cat	m, sg	f, sg	_, sg	m, pl	f, pl	_, pl	_, _	neu, sg	núcleo
1	art	el	la		los	las			lo	-
2	indf1a	un, cualquier algún, ningún								-
3	indf1b		mucha, poca demasiada							-
4	indf2a	mucho, demasiado	una, alguna, ninguna	bastante	unos, algunos, muchos, demasiados, varios	unas, algunas, muchas, demasiadas, varias	bas-tantes			+
5	indf2b	otro	otra		otros	otras				+
6	indf2c	poco			pocos	pocas				+
7	dem	este, ese aquel	esta, esa aquella		estos, esos aquellos	estas, esas aquellas				-
8	pos	nuestro, vuestro	nuestra, vuestra	mi, tu, su	nuestros, vuestros	nuestras, vuestras	mis, tus,			-

							sus			
9	tod	todo	toda		todos	todas				+
10	amb				ambos	ambas				+
11	send				sendos	sendas				-
12	cad							cada		-
13	ciert	cierto	cierta		ciertos	ciertas				-
14	tant	tanto	tanta		tantos	tantas				+

Tabla 1. Inventario de los determinantes.

art	Modifican a 'n', 'adj', 'num', 'mism', 'prpos', salvo *lo* que no va con 'n' ni con 'num1'.
indf1a	Modifican a 'n', 'ad'j y 'num' (en el caso de' num1' cuando se refiere al nombre del número). Se combinan con 'indf2b'.
indf1b	Modificn a 'n' y pueden combinarse con'indf2b'.
indf2a	Pueden modificar a 'n' y, salvo *mucho,* a 'adj', *algunos, muchos, demasiados, bastantes* y *varios* pueden modificar a 'num1' cuando se refieren al nombre del número y *unos* y *unas* también pueden modificar a 'num1'. Pueden modificar a 'num2', salvo *mucho, demasiado* y *bastante*. Admiten combinaciones con 'indf2b': *alguna otra, algunos otros, algunas otras, ninguna otra, muchos otros, muchas otras, demasiados otros, demasiadas otras, varios otros, varias otras.*
indf2b	Se anteponen a 'n' y a 'adj', también a 'num2' y, salvo *otra* a 'num1'. Se posponen a 'dem', a 'indf2a' salvo *una, unos, unas, demasiado, mucho y bastante. Otros* y *otras* se posponen a *tantos* y *tantas.*
indf2c	Se anteponen a 'n' y a 'adj'. Se posponen a 'indf2a' en plural y se anteponen y posponen a 'indf2b', excepto **poco otro*.
dem	Modifica a 'n', 'adj', 'num2' y a 'num1' (salvo los femeninos singulares).
pos	Las formas *nuestro, nuestra, nuestros, nuestras* son ambiguas en cuanto a su asignación categorial. Son 'pos' cuando inician el snn y 'prpos' cuando son núcleo.
tod	Siempre inicia el snn. En singular modifica directamente a 'n'; en plural, seguido de 'art', puede modificar a 'n' , 'adj' o 'indf2c'.
amb	Puede modificar a 'n' o 'adj'. Siempre está al comienzo.
send	Puede modificar a 'n'. Siempre está al comienzo.
cad	Va en singular con 'n', 'adj' y 'num2' y en singular o plural con 'num1'.
ciert	Va con 'n' o 'adj'.
tant	Va con 'n', 'adj' o 'indf2b'

Tabla 2. Observaciones sobre los determinantes.

4.1.3.2 Nombres

Trataremos separadamente a los nombres comunes ('nc') y a los nombres propios ('npr').
Los ‘nc’:

(i) Constituyen un inventario abierto.
(ii) Son núcleos de snn.
(iii) Pueden ir antecedidos de ‘det’, ‘adj’, ‘num’ y ‘adv’.
(iv) En plural pueden ir solos, sin que los anteceda ningún elemento
(v) En singular no pueden iniciar el snn.

Los ‘npr’:

(i) Los de persona simple, en dialecto rioplatense, van solos o precedidos de artículo acompañado por ‘adj’, o ‘num’ (ejemplo: *Carlos, el famoso Carlos*)[3] o precedido de ‘dem’.
(ii) Los de persona complejos están formados por uno o más nombres de pila y un apellido y las restricciones respecto de la combinatoria son las mismas que las planteadas en (i).
(iii) Los de institución lleva siempre: ‘art’, ‘indf’, ‘dem’, 'pos', ‘cad’, ‘ciert’, ‘amb’, ‘tant’, 'num' esté el nombre solo, o con ‘adj’ (ejemplos: *la Corte Suprema de Justicia, la famosa Corte Suprema de Justicia*).
(iv) Los geográficos se comportan distinto según los lemas: *Asia/ el Asia, Argentina/ la Argentina, Chile/*el Chile,* pero todos pueden comportarse como (i), ejemplo: *el Chile de Pinochet.*

4.1.3.3 Adjetivos

Distinguimos dos tipos de adjetivos: 'adj1' y 'adj2'. Ambos pueden ser núcleos precedidos de 'det' o 'num'. Los primeros ('adj1') corresponden a los tradicionalmente llamados calificativos; incluyen a los participios. En plural pueden iniciar el snn y pueden estar modificados por un adverbio. Los 'adj2' corresponden a los tradicionalmente llamados relacionales. No pueden ir entre 'det' y 'n' y no pueden ser modificados por un adverbio.

4.1.3.4 Numerales

En esta etapa vamos a considerar sólo a los numerales tradicionalmente llamados cardinales ('num1') y ordinales ('num2'). Ambos:
(i) Pertenecen a una clase abierta.
(ii) Pueden ser núcleo del snn y cumplir lo que podría llamarse *función pronominal.*
(iii) Pueden estar al comienzo del snn u ocupar el lugar siguiente al determinante.
(iv) Cuando no son núcleo se refieren o modifican al núcleo del snn al que pertenecen.

4.1.3.5 Pronombres

Los pronombres son siempre núcleo y con la excepción de 'prpos' deben encabezar el snn y no exigen, con la excepción de la relación con 'tod', satisfacer relaciones de concordancia dentro del snn; con excepción de los 'pindf' pueden estar precedidos de 'tod'; cf. la tabla 3, que sigue a continuación.

Cat	m, sg	f, sg	_, sg	neu, sg	m, pl	f, pl	_, pl
prpers	él	ella	yo, tú, vos, usted		nosotros, vosotros	nosotras vosotras	ustedes
prindf	algo, uno alguno, alguien, ninguno, cualquiera		alguien, nadie, nada				
prdem	éste, ése, aquél	ésta, ésa, aquélla		esto, eso, aquello	éstos, ésos, aquéllos	éstas, ésas, áquéllas	
prpos	mío, tuyo, suyo, nuestro, vuestro	mía, tuya, suya, nuestra, vuestra		mío, tuyo suyo, nuestro vuestro	míos, tuyos, suyos, nuestros, vuestros	mías, tuyas, suyas, nuestras, vuestras	

Tabla 3. Inventario de pronombres.

[3]Salvo registros propios de la oralidad, marcados sociolingüísticamente.

4.1.3.6 Adverbios

Los ‘adv’:

(i) Pueden ir antes de un ‘adj1’.
(ii) Pueden ir antes de ‘n’: Son muy pocos los nombres que admiten adverbio (*la casi totalidad de sus costos, la no observancia de los criterios básicos*).
(iii) *muy, bastante y demasiado* pueden ir antes de *pocos.*

4.2 Propiedades de existencia

Las propiedades de existencia se expresan mediante las categorías que deben subsumir a las categorías que pueden ser reconocidas en los modelos, y a las relaciones de implicación que se pueden dar entre ellas. Determinan: (i) cuáles son las categorías que se pueden utilizar en los modelos, (ii) cuáles pueden ser núcleo, (iii) cuáles aparecen una sola vez, (iv) cuáles exigen o excluyen la presencia de otras, (v) cuáles concuerdan entre sí.

P1: Es la Propiedad referente al vocabulario. Determina todas las categorías (y sólo ellas) que se pueden utilizar en los modelos.
P1: amod (snn-esp, [det, n, adj, num, pron, adv]).

P2: Es la Propiedad que determina qué categorías pueden ser núcleo del snn. Una categoría núcleo se anota *°cat*.
P2: núcleo (snn-esp, [indf2, tod, amb, tant, mism, n, adj, num, pron]).

P3: Es la propiedad que determina cuáles categorías aparecen una sola vez.
P3 unic (snn-esp, [art, indf1, indf2a, indf2b, indf2c, dem, pos, tod, amb, send, cad, ciert, tant, mism, nc, adj2, num2, pron])[4].

Propiedades de exigencia

Las propiedades de exigencia expresan que un subconjunto de categorías está exigido en un modelo si se da otro subconjunto de categorías. La presencia en un modelo de una categoría subsumida por la primera categoría de la fórmula determina la presencia en el modelo de por lo menos otra categoría subsumida por otra categoría de la fórmula. Por ejemplo, dado P+1, si en un modelo hay 'art' debe haber también por lo menos una categoría subsumida por 'n', 'adj', 'indf2b', 'mism', 'num' o 'prpos'. *NEANT* es una variable sobre la ausencia de toda categoría.

P+1: exig (snn-esp, [[art],[n],[adj],[indf2b],[mism],[num],[prpos]]).
P+2: exig (snn-esp, [[indf1],[n]]).
P+3: exig (snn-esp, [[indf1a],[indf2b],[adj],[num]]).
P+4: exig (snn-esp, [[pos],[n],[adj],[num2]]).
P+5: exig (snn-esp, [[prpos],[art]]).
P+6: exig (snn-esp, [[ciert],[n],[adj]]).
P+7: exig (snn-esp, [[cad],[n, sg],[adj,sg],[num1,_],[num2,sg]]).
P+8: exig (snn-esp, [[send],[n,pl]]).
P+9: exig (snn-esp, [[nc, sg],[det, sg]]).
P+10: exig (snn-esp, [[°tod], NEANT]).
P+11: exig (snn-esp, [[°amb], NEANT]).
P+12: exig (snn-esp, [[°tant], NEANT]).
P+13: exig (snn-esp, [[prindf], NEANT]).
P+14: exig (snn-esp, [[prpers], NEANT]).

[4]'num1' puede repetirse cuando uno de ellos es el nombre del número, ejemplo: *Salieron dos tres* (en un juego)

P+15: exig (snn-esp, [[prdem], NEANT]).

Concordancia

Con la regla que sigue se expresan las relaciones de concordancia entre dos categorías con valores de concordancia.

Pconc: exigc (snn-esp, [[detGN], [nGN], [adjGN], [numGN], [pronGN]]).

Propiedades de exclusión

Las propiedades de exclusión expresan que un subconjunto de categorías está excluido en un modelo si se da otro subconjunto de categorías. La presencia en un modelo de una categoría subsumida por cualquier categoría en la fórmula determina la ausencia en el modelo de una categoría subsumida por toda otra categoría en la fórmula. Por ejemplo, dado P~1, si en un modelo hay 'art' no puede haber 'dem' ni ninguna categoría subsumida por otra categoría de la fórmula.

P~1: exclu(snn-esp, [[ºdet],[n],[ºadj],[ºnum],[pron]]).
P~2: exclu(snn-esp, [[amb],[art],[dem],[pos],[cad],[prpers],[prindf],[prdem]]).
P~3: exclu(snn-esp, [[amb],[indf]])
P~4: exclu(snn-esp, [[amb],[tod],[send],[tant]]).
P~5: exclu(snn-esp, [[amb],[prpos]]).
P~6: exclu(snn-esp, [[dem],[indf1],[indf2a],[tant]]).
P~7: exclu(snn-esp, [[dem],[cad],[prpers],[prindf],[prdem]]).
P~8: exclu(snn-esp, [[pos],[art],[dem],[cad]]).
P~9: exclu(snn-esp, [[pos],[ciert],[pron]]).
P~10: exclu(snn-esp, [[num1],[tod],[send],[ciert],[tant],[prpers],[prindf],[prdem]]).
P~11: exclu(snn-esp, [[art],[ciert],[tant],[prpers],[prindf],[prdem]]).
P~12: exclu(snn-esp, [[cad],[tod],[send],[mism],[adv]]).
P~13: exclu(snn-esp, [[cad],[pron]]).

4.3 Propiedades de linealidad

Introducen las relaciones de orden. Formalmente transforman los conjuntos en listas. La presencia en un modelo de una categoría subsumida por la primera categoría de la fórmula antecede la presencia de cualquier otra categoría en el modelo subsumida por otra categoría de la fórmula. Por ejemplo, dado P<1, si en un modelo hay 'det' y cualquier categoría subsumida por otra categoría de la fórmula 'det' la precede.

P<1 <snn-esp, {_}, precede(det, {n, adj, num, pron, adv})>.
P<2 <snn-esp, {_}, precede(art, {indf})>.
P<3 <snn-esp, {_}, precede(indf1, {indf2})>.
P<4 <snn-esp, {_}, precede(adj, {n})>.
P<5 <snn-esp, {_}, precede(tod, {n, det, adj})>.
P<6 <snn-esp, {_}, precede(num, {n})>.
P<7 <snn-esp, {_}, precede(num1, {adj})>.
P<8 <snn-esp, {_}, precede(adv, {n, adj, num1})>.

4.4 Propiedades de flechado

Las propiedades de flechado especifican el grafo a partir del cual se calculan las relaciones semánticas. Si varias categorías flechan sobre una (generalmente el núcleo) las que flechan y la que recibe el flechado obran como los argumentos de una función a partir de la cual puede calcularse la representación semántica que debe estar asociada a estas categorías; *flecha x, y [y]*

expresa *x flecha sobre y;\{x}* expresa la exigencia de ausencia de *x*. Por ejemplo, dado F1, 'det' flecha sobre 'n'.

F1 <snn-esp,{_}, flecha(det, [n])>.
F2 <snn-esp,{_}, flecha(det, [prpos])>.
F3 <snn-esp,{_}, flecha(det, [mism])>.
F4 <snn-esp,[\(n)], flecha(det, [adj])>.
F5 <snn-esp,[\{[n],[adj]}], flecha(det, [num])>.
F6 <snn-esp,[_], flecha(num, [n])>.
F7 <snn-esp,[\(n)], flecha(num, [adj])>.
F8 <snn-esp,[\{[n],[adj]}], flecha(num1, [num2])>.
F9 <snn-esp,[_], flecha(cad, [n])>.
F10 <snn-esp,[\{[n],[adj]}], flecha(cad, [num])>.
F11 <snn-esp,[_], flecha(ciert, [n])>.
F12 <snn-esp,[_], flecha(amb, [n])>.
F13 <snn-esp,[_], flecha(adj, [n])>.
F14 <snn-esp,[_], flecha(art, [prpos])>.
F15 <snn-esp,{_}, flecha(indf, [n])>.
F16 <snn-esp,[\(n)], flecha(indf, [adj])>.
F17 <snn-esp,[\{[n],[adj]}], flecha(indf, [num2])>.
F18 <snn-esp,[\{[n],[adj]}], flecha(indf1, [indf2])>.

5 Balance y perspectivas

Nuestro trabajo futuro se organiza en torno a dos ejes: (a) Completar la descripción y modelización del snn en lo que hace a los puntos no presentados aquí, mencionados en la nota 1 y a las etapas mencionadas en la Introducción; (b) Completar el tratamiento del snn con las herramientas Smorph y MPS, que ya tenemos muy avanzado, pero que las limitaciones de espacio nos han impedido mostrar.

El orden en que lo presentamos no es cronológico ya que el trabajo con las herramientas informáticas es necesario para poner a prueba el llevado a cabo en la descripción y formalización.

6 Referencias

Steven Abney. 1991. Parsing by Chunks. En Berwick et al. (1991).

Salah Aït-Mokhtar y Jean-Pierre Chanod. 1997. Incremental Finite-State Parsing. En *Proceedings of the 8th Conference on Applied Natural Language Processing, ANLP-97*. Washington, 72-79

Robert Berwick, Steven Abney y Carol Tenny (eds.). 1991. *Principle-Based Parsing.* Kluwer Academic Publishers, Dordrecht.

Gabriel G. Bès. 2002. La linguistique, entre science et ingénierie. *TAL*, 41(3): 57-81.

Gabriel G. Bès. 1999. La phrase verbale noyau en français. *Recherches sur le français parlé*, 15: 273-358.

Gabriel G. Bès, Caroline Hagège y Luisa Coheur. 1999. De la description des propriétés linguistiques à l'analyse d'une langue. En *VEXTAL*, Venecia, noviembre 1999.

Gabriel G. Bès y Zulema Solana. 2004. Los clíticos en español. Congreso Internacional sobre Políticas de la Cultura, UBA, Buenos Aires.

Luisa Coheur, Nuno Mamede y Gabriel G. Bès. 2003. AsdeCopas: a syntactic-semantic interface. En *EPIA03 Workshop on Natural Language and Text Retrieval*, Evora.

Núria Gala. 2005. *Analyse syntaxique automatique avec XIP.* Séminaire TAL du DELIC, Universidad de Provence.

G. Gazdar, E. Klein, G. Pullum y I. Sag. 1985. *Generalized Phrase Structure Gram*mar. Oxford, Blackwell.

Caroline Hagège. 2000. *Analyse syntaxique automatique du portugais*. Tesis de Doctorado. Universidad Blaise-Pascal/GRIL, 2000.

Capítulo 8

MODELIZACIÓN DE LAS FUENTES DECLARATIVAS EN UNA HERRAMIENTA DE ANÁLISIS Y CONJUGACIÓN AUTOMÁTICOS DE VERBOS DEL ESPAÑOL

Zulema Solana, Rodolfo Bonino y Viviana Valenti

En Víctor M. Castel, Comp. (2005) *Desarrollo, implementación y utilización de modelos para el procesamiento automático de textos*. Mendoza: Editorial de la Facultad de Filosofía y Letras, UNCuyo: 81-92.
ISBN 987-575-019-0 del soporte Internet

Modelización de las fuentes declarativas en una herramienta de análisis y conjugación automáticos de verbos del español

Zulema Solana, Rodolfo Bonino y Viviana Valenti

Universidad Nacional de Rosario
Facultad de Humanidades y Artes
Rosario, Argentina
zsolana@arnet.com.ar // rodolfobonino@yahoo.com.ar // letsgo@tau.org.ar

Resumen

El español presenta una compleja morfología verbal en los paradigmas llamados regulares y en mayor medida en los irregulares, verbos que, en su flexión, presentan cambios vocálicos, consonánticos y acentuales. Tenemos como objetivo lograr la modelización de las fuentes declarativas en una herramienta de análisis y conjugación automáticos de verbos del español. Consideramos que el trabajo planteado puede ser de utilidad para la enseñanza de la lengua tanto a hablantes nativos como extranjeros, ya que no sólo describimos la metodología empleada en la implantación en máquina de la morfología verbal sino también la acompañamos con lo que habría que informarle a un hablante para que pueda conjugar como lo hace la herramienta aludida. En nuestra modelización diferenciamos la conjugación regular y los subconjuntos de terminaciones verbales regulares que se concatenan con raíces no regulares, damos a conocer las raíces irregulares y los rasgos que completan la información. De este modo es posible requerir la generación de formas verbales de determinados paradigmas de verbos que experimentan determinadas modificaciones en su raíz y, además, obtener la conjugación completa de las formas simples de cualquiera de los verbos ingresados. Presentamos además un "algoritmo conceptual" fundado en los factores que determinan las irregularidades, el que va a permitir asociar terminaciones con raíces, y que también puede ser explotado en la enseñanza del español. En esta presentación vamos a focalizar nuestra atención en los verbos en -ar.

1 Introducción[*]

El español presenta una compleja morfología verbal en los paradigmas llamados regulares y en mayor medida en los irregulares. Se menciona con este nombre a los verbos que, en su flexión, manifiestan cambios vocálicos, consonánticos y acentuales en su raíz o tienen terminaciones distintas a las del paradigma regular.

Hacemos nuestros los argumentos que despliega Alegría (1995) para justificar la necesidad de un analizador morfológico frente a los diccionarios completos o léxicos desplegados que se usan para lenguas como el inglés y nos parece de particular importancia la argumentación basada en la posibilidad de desarrollar herramientas a partir de los analizadores morfológicos: correctores, lematizadores para recuperación de información, aplicaciones para enseñanza de lenguas, etc. En esta presentación focalizaremos la posible aplicación a la enseñanza del español; creemos que el aporte es nuevo en lo que respecta a las informaciones que puede brindar tanto al enseñante como al aprendiente.

[*]Este trabajo se inscribe en el proyecto *MVE* (Modelización del Verbo Español), coordinado por Gabriel G. Bès, a quien agradecemos sus sugerencias para el planteamiento del problema y para la especificación del algoritmo propuesto en la Sección 5 para el cálculo de las entradas; la responsabilidad del trabajo corre por nuestra exclusiva cuenta. Han participado en la especificación de las fuentes de SMORPH: Liliana Bolla, Susana Freidenberg, Rosana Gasparini, Claudia Kocak, Walter Kozza, Stella Maris Moro y Andrea Rodrigo, integrantes del Proyecto INFOSUR: investigación y desarrollo.

Nuestro propósito es lograr la modelización de las fuentes declarativas en una herramienta de análisis y lematización automáticos de verbos del español que puede ser utilizada sobre textos reales. En esta presentación vamos a ocuparnos de los verbos en -ar, pero la metodología es aplicable a las otras conjugaciones. Antes de exponer la modelización, haremos algunas referencias a tratamientos no computacionales de la morfología verbal, en especial la irregular, y nos referiremos a las herramientas accesibles en la web.

2 Tratamientos no computacionales: algunos factores determinantes de la variabilidad morfológica verbal del español

En todo tratamiento que se haga de los verbos españoles van a estar presentes todos o algunos de los factores mencionados a continuación:

DIPTONGACIÓN

Una gran parte de los verbos (Pensado 1999) que en latín tenían -e- u -o- breves, por lo tanto abiertas, en la sílaba tónica diptongan. Así en lugar de -e- acentuada podemos encontrar –ie- y en lugar de –o-, -ue-; ejemplos: *acierto* (y no **acerto*), *vuelo* (y no **volo*), lo que produce alternancia en la raíz de las formas con sílabas tónicas y con sílabas átonas.

ACENTUACIÓN

La acentuación de los verbos sigue un esquema fijo (Harris 1983) que se limita a una de las tres últimas sílabas de las palabras y no es sensible a la estructura de éstas como ocurre con otro tipo de categorías (como los nombres por ejemplo). En los verbos irregulares hay modificaciones acentuales, por ejemplo en lugar de *canté/cantó* del pretérito, se encuentra *anduve/anduvo*.

MODIFICACIONES CONSONÁNTICAS

- FONOLÓGICAS: Pueden introducir elementos consonánticos, eliminarlos o modificarlos; daremos solamente algunos ejemplos. Algunos verbos introducen un infijo de origen incoativo -z- ante -c, como *traducir/traduzco;* otros eliminan la –i- de la vocal temática cuando la precede una consonante palatalizada, como *partió/ciñó*.[1]
- GRÁFICAS: Los verbos cuya raíz termina en -c, -g o-z la cambian respectivamente en -qu, -gu o -c ante -e o -i; ejemplos: *sacar/saqué, pagar/pagué, aterrizar/aterricé*.

Haremos primero mención de una descripción tradicional reciente de los verbos irregulares para determinar qué es lo que le falta para ser apta en el marco de la lingüística computacional, es decir, para ser apta para servir como fuente declarativa de una herramienta de tratamiento automático. Nos referimos al tratamiento que Alcoba (1999) hace de los verbos mencionados; hacemos esta elección porque es una muy buena descripción. Presentaremos los problemas que plantea y las respuestas que da.

Problema 1: En qué consiste la irregularidad.

Respuestas:

a) Se altera el constituyente radical (*apretar/aprieto*). Aquí distingue entre irregularidad fonológica e irregularidad gráfica.
b) Se suprime la vocal temática (*tendré* y no **teneré*).
c) Toma diferentes raíces (ten-/tien-/teng-/tuv).

Problema 2: Qué extensión tiene la irregularidad. Distingue entre:

[1]Damos ejemplos de otras conjugaciones con la única finalidad de presentar el esquema completo, ya que los verbos en -ar no presentan esta irregularidad.

a) Extensión externa de la irregularidad: conjunto de verbos afectados.
b) Extensión interna: tiempos verbales afectados (tema de presente, de pretérito o de futuro). A pesar de ser una descripción bastante exhaustiva, ¿qué inconveniente le vemos, para tomarla como base de un tratamiento computacional? No se ve cómo, a partir de ella, pueden establecerse modelos de verbos, dado que los problemas no aparecen integrados en un eje.[2]

3. Herramientas de análisis automático accesibles en la web

Una de las ventajas de Internet es la de poner conjugadores al alcance del usuario. Ahora bien, la eficacia de dichos conjugadores no debe medirse por la cantidad de verbos que éstos son capaces de flexionar, ni por la sofisticación de los enlaces o propuestas de navegación que sugieren, sino por la precisión con que lo hacen y por su capacidad productiva. Para tener una idea de la precisión hay que someterlos a diferentes pruebas y comparar el modo en que responden. De aquí que sugiramos la agrupación de los conjugadores de verbos del español en dos grandes grupos: aquellos que son confiables y aquellos que, en ocasiones, dan respuestas incorrectas o incompletas, por lo cual su uso no es recomendable. En este último grupo se encuentran conjugadores como los siguientes:

http://www.prologo.net/spanconj.aspx
http://www.idiomax.com/Es/conjugate.asp
http://www.tranzsend.com/Spanish/Demo.asp#
http://allserv.ugent.be/~gdschrij/cgi-bin/es_conjugador3.pl
http://www.sintx.usc.es/conjuga.html

Si bien se podría describir a cada uno de ellos en particular, se puede hablar en general de sus falencias. Una de ellas es visual y hace a la manera en que la información es presentada al usuario. Conjugadores como los dos primeros no visualizan la conjugación en toda la pantalla sino que lo hacen en una ventana muy pequeña que requiere el uso de la barra deslizadora, dificultando la obtención de los datos requeridos. Algunos no toman palabras escritas en mayúsculas o que contengan una de sus letras en mayúsculas, llegando a tildarse. Otro inconveniente se da con verbos que pueden conjugarse de dos maneras (ejemplos. *aterrar*, *atentar*). En general nos encontramos con que dan una sola conjugación. Los verbos defectivos son otra de sus debilidades. Puede darse que presenten toda su conjugación o bien que los rechacen. El más grave de los problemas lo tienen los dos últimos conjugadores ya que aceptan palabras inexistentes en el español, como por ejemplo *tradusir* y, luego de asociarlas a un paradigma, las conjugan.

Veamos ahora los conjugadores más confiables:

http://www.gedlc.ulpgc.es/investigacion/scogeme02/flexver.htm (Universidad de las Palmas de Gran Canarias)
http://buscon.rae.es/diccionario/cabecera.htm (Real Academia Española)
http://www.verbix.com/webverbix/index.asp (UNESCO)
http://tradu.scig.uniovi.es/conjuga.html (Universidad de Oviedo)
http://www.verba.org/owa-v/verba_dba.verba_main.create_page?lang=es
http://www.lenguaje.com/herramientas/conjugador (SIGNUM)
http://turingmachine.org/compjugador/ (Daniel M. Germán)

Todos ellos aceptan mayúsculas o minúsculas indistintamente, lo que garantiza la obtención de la información. Salvo el primero, que exige la selección del tiempo y modo, todos dan el total de la conjugación del verbo con una muy buena presentación visual. Los más completos son los

[2]El Larousse (1993) y el Bescherelle (1997) multiplican los paradigmas (el primero presenta 90 y el segundo 89 para las tres conjugaciones) sin que la organización subyacente sea explícita. Le Goffic (1997), por su parte, hace un aporte importante proponiendo las seis formas verbales claves para la conjugación del verbo francés al estilo del modo de enunciar los verbos de los diccionarios del latín.

tres primeros porque son capaces de trabajar los verbos con distinto significado, aclarar la situación de los defectivos y permiten la obtención de datos categoriales. Asimismo, facilitan la búsqueda de datos complementarios como sinónimos, antónimos, colocaciones idiomáticas, modelos verbales, datos descriptivos de diversa índole, entre otros. También se caracterizan por tener vínculos con diccionarios y traductores. Los conjugadores restantes son confiables respecto a la manera en que flexionan los verbos pero presentan algunas limitaciones en función de los verbos con doble conjugación, verbos defectivos, o bien, en la posibilidad de la obtención de datos correlacionados con la entrada léxica. Independientemente del mayor o menor grado de precisión, estos conjugadores tienen dos limitaciones: sólo se pueden usar en tareas de conjugación y, por lo menos para el usuario final, no son declarativos. Es decir, actúan como cajas negras imposibles de modificar para ser adaptadas a diferentes usos. Los resultados finales no pueden ser alterados por la intervención de un usuario que desee modificar el tipo de información que se desea obtener.

4 SMORPH, analizador y conjugador

Nos proponemos implantar en la herramienta informática SMORPH (Aït-Mokhtar 1998) la morfología de las formas verbales simples, flexionadas o no, del español,[3] para poder luego:

- relacionar propiedades lingüísticas con el análisis y la generación automáticos;
- utilizar las propiedades de la modelización utilizada en tareas de enseñanza del español como conjugador declarativo e inteligente;
- utilizar las propiedades de la modelización en tareas de análisis automático y de gestión de la documentación.

El software SMORPH (Segmentación y morfología), es un analizador y conjugador de palabras simples o compuestas, con o sin flexión.[4] En cuanto al analizador, trata el análisis presintáctico (tokenización y análisis morfológico) en una sola etapa. En cuanto conjugador, produce las formas correspondientes a un lema o a un subconjunto de lemas con los valores requeridos. Es una herramienta perfectamente declarativa: la información utilizada por SMORPH está totalmente separada de la maquinaria algorítmica, lo que permite adaptarla al uso que quiera darse. En SMORPH deben declararse cinco tipos de información:

i Códigos Ascii
ii Rasgos
iii Terminaciones
iv Modelos
v Entradas

En (i) se especifican los códigos Ascii que van a merecer un tratamiento especial, p. ej. aquellos que quieren utilizarse como separadores de palabras; se supone que (i) es común para todas las estructuras de la lengua y no se lo abordará en este trabajo.

Los rasgos en (ii) obedecen a la estructura general:

<ETIQUETA; {v_1, ...; v_n}>

Es decir, un rasgo es una ETIQUETA asociada a un conjunto de valores. Ejemplo: la etiqueta *NUM* va a tener como valores *sg* y *pl.*

Los rasgos utilizados se pueden clasificar en dos grupos:

[3]SMORPH no concatena en el análisis ocurrencias sucesivas ni genera secuencia de ocurrencias.
[4]Ha sido utilizado para el español en (Aït-Mokhtar y Rodrigo 1995; Rodrigo y Bès 2004).

ii_a los relacionados con los valores morfológicos de las terminaciones verbales: modo, tiempo, persona, género (para los participios), número;

ii_b los relacionados con la caracterización del tipo de conjugación y con sus aspectos regulares o irregulares;

Los rasgos de tipo (ii_a) son los habituales para un conjugador o analizador. Los rasgos de tipo (ii_b), presentados en la Sección 6, van a permitir analizar y conjugar subconjuntos de lemas caracterizados por su regularidad o irregularidad morfológica.

Las terminaciones en (iii) son simplemente un conjunto de secuencias de caracteres; ejemplos: *abas, aron, é* para las formas flexionadas, o, *ando*, para las formas no flexionadas.

Los modelos en (iv) especifican ocurrencias complejas por medio de la concatenación de cadenas adyacentes. Un modelo tiene la estructura general:

@IDM -n
+ $term_1$ LV_1.
...
+ $term_n$ LV_n.

en donde *IDM* es la Identificación del modelo, $n \geq 0$ es el número de caracteres que debe suprimirse de la terminación del lema para especificar la raíz, $term_i$ es una terminación en (iii) y LV_i es una lista de valores de rasgos declarados en (ii). Por ejemplo, en la flexión nominal, a partir de un modelo, se obtienen *niño, niña, niños, niñas* mediante la concatenación del radical *niñ* con, respectivamente, las terminaciones *o, a, os, as*, concatenaciones a las que se les asocia los valores correspondientes de género y número; a partir del mismo modelo se obtienen las formas correspondientes a *abogado* y *perro*, entre otras.

Hay varios tipos posibles de declaraciones de entradas en (v). En este trabajo utilizamos los cuatro siguientes:

v_a	lema		@IDM.
v_b	lema	raíz	@IDM.
v_c	lema		LV.
v_d	lema	raíz	LV.

En (v_a) y (v_b), *IDM* es el identificador de modelo por el que será tratado el lema en (v_a) o la raíz en (v_b). En el caso de (v_a), es el modelo quien calcula la raíz. En ambos tipos se puede completar el *IDM* con una *LV*. Por ejemplo, con las entradas

acertar	@v5/c1.
sonar	@v5/c1.

se dice que los lemas *acertar* y *sonar* serán tratados por el modelo *v5* (*v5*: identificador del modelo usado en este trabajo para las formas regulares como *acertaban*, *soñaban* de los verbos que diptongan en otros casos, como *aciertan* y *sueñan*). Este modelo va a asociar las terminaciones a las raíces calculadas sobre los lemas (es decir, raíces *acert* y *soñ*) y cada terminación estará asociada a la *LV* especificada en el modelo. Pero en las entradas también se dice que SMORPH, al analizar o conjugar, va a agregar el valor *c1* a cada una de las *LV* asociadas por *v5* a las terminaciones, es decir, *c1* se hereda en todas las formas tratadas por *@v5.*

En los tipos (v_c) y (v_d), la entrada no asocia el lema con un modelo, sino que especifica directamente la *LV* que debe asociarse a la forma del lema (v_c) o la raíz (v_d).

5 Cálculo de modelos

La conjugación regular está expresada mediante un conjunto finito de terminaciones, cada una de las cuales está asociada a un conjunto de valores, es decir a una *LV*. La raíz es, en todos los casos, el lema menos los dos caracteres de la terminación del infinitivo. Esa raíz va a concatenarse con cada terminación. La conjugación regular está expresada mediante el modelo *v1*.

La irregularidad se manifiesta en la raíz o en las terminaciones, lo que exige determinar:

- qué subconjunto de terminaciones regulares se concatenan con raíces no regulares;
- qué subconjunto de terminaciones irregulares se concatenan con raíces regulares o no.

De manera que, al extraer los conjuntos de terminaciones que no siguen la conjugación regular quedan como complemento las terminaciones que sí siguen las conjugaciones regulares. Esto posibilita determinar todo lo que es regular en los verbos irregulares y especificar todo lo que es irregular en ellos. Basándonos en estos criterios establecemos los modelos irregulares y sus complementos según aparecen en la Tabla 1. Aclaraciones para su lectura:

-La primer columna corresponde a las formas del modo indicativo, del que se enumeran las terminaciones asociadas a la raíz irregular, la segunda columna corresponde al subjuntivo.
-Las letras en mayúscula, ubicadas en el ángulo inferior izquierdo de cada rectángulo, identifican a los subconjuntos de terminaciones.
-Las filas, en las que aparecen R1, R2, R3 y R4, definen el complemento del o de los modelos irregulares correspondientes; *Tar* es el conjunto de terminaciones regulares menos la terminación de infinitivo; ejemplo: R1 es el conjunto que resulta de sustraer la unión de A y D a *Tar*; en R1 las terminaciones y la raíz a la que están asociadas son regulares.
-En la tercer columna introducimos ejemplos, en la cuarta sus raíces irregulares y en la última (quinta columna) los modelos con su *IDM* utilizados en SMORPH.

Indicativo	Subjuntivo	Ejemplos	Raíces	Modelos
pres/1a/sg -o pres/2a/sg -as pres/3a/sg -a pres/3a/pl -an A	pres/1a/sg -e pres/2a/sg -es pres/3a/sg -e pres/3a/pl -en D	acertar sonar	aciert suen	v4
R1 Tar \A U D				v5
pres/1a/sg -o pres/2a/sg -as pres/3a/sg -a pres/3a/pl -an A		jugar trocar colgar negar comenzar avergonzar	jueg truec cuelg nieg comienz avergüenz	v6
	pres/1a/sg -e pres/2a/sg -es pres/3a/sg -e pres/3a/pl -en D		juegu truequ cuelgu niegu comienc avergüenc	v7
indf/1a/sg -é B	pres/1a/pl -emos pres/2a/pl -éis C		jugu troqu colgu negu comenc avergonc	v8

R2 Tar \A U D U B U C				v9
indf/1a/sg -é B	pres/1a/sg -e pres/2a/sg -es pres/3a/sg -e pres/1a/pl -emos pres/2a/pl -éis pres/3a/pl -en E	sacar pagar cazar averiguar	saqu pagu cac averigü	v10
R3 Tar \B ∪ E				v11
indf/1a/sg -uve indf/2a/sg -uviste indf/3a/sg -uvo indf/1a/pl -uvimos indf/2a/pl -uvisteis indf/2a/pl -uvieron indf/3a/pl -uvieron F	pret/subj/1a/sg -ara pret/subj/2a/sg -aras pret/subj/3a/sg -ara pret/subj/1a/p l-áramos pret/subj/2a/pl -arais pret/subj/2a/pl -aran pret/subj/3a/pl -aran G	andar estar	and est	v12
indf/1a/sg -í indf/2a/sg -iste indf/3a/sg -ió indf/1a/pl -imos indf/2a/pl -isteis indf/2a/pl -ieron indf/3a/pl -ieron F	pret/subj/1a/sg -iera pret/subj/2a/sg -ieras pret/subj/3a/sg -iera pret/subj/1a/pl -iéramos pret/subj/2a/pl -ierais pret/subj/2a/pl -ieran pret/subj/3a/pl -ieran G	dar	d	v13
R4 Tar \F ∪ G				v14

Tabla 1. Modelos.

¿Cómo decidimos a cuál o cuáles modelos y con qué raíz o raíces está asociado un verbo dado? Es el cálculo de entradas, en la sección que sigue, el que debe dar la solución.

5 Cálculo de entradas

Sabemos que, en SMORPH, es en las entradas en donde se especifica la asociación de lemas con modelos. Nuestra posición es que las entradas pueden calcularse mediante un algoritmo en cuya especificación se aprovechan los factores generales que introducen modificaciones vocálicas, consonánticas y de acentuación. Proponemos, además, que el mismo algoritmo pueda utilizarse para explicar al aprendiente los procesos que condicionan la morfología verbal del español. En este trabajo se presenta lo que puede caracterizarse como un "algoritmo conceptual" para el cálculo de las entradas, algoritmo no implantado en máquina (pero que podría serlo de manera directa) y que es ejecutable por un ser humano.

Las entradas se calculan entonces mediante el algoritmo propuesto (Sección 5.1) y mediante estipulaciones complementarias (Sección 5.2), las que van a completar la especificación de los verbos hiper-irregulares *estar, andar* y *dar*.

5.1 Algoritmo para el cálculo de entradas

Definiciones:

lema = infinitivo
raíz = raíz del infinitivo = infinitivo menos terminación del infinitivo (2 caracteres)
vocal de la última sílaba de la raíz del infinitivo = vocal más próxima a la terminación

En lo que sigue *raíz-m1* a *raíz-m6* será una raíz modificada por solamente un cambio de tipo vocálico, *raíz-m1-f* a *raíz-m5-f* será una raíz modificada por cambios de tipo vocálico y conso-

nántico y *raíz-f* será una raíz modificada por solamente cambios consonánticos. A continuación de cada entrada, en caracteres más pequeños, se indican ejemplos.

Pr0 Mirar el lema. ¿Es estar o andar o dar?
SÍ
SALIDA:
lema raíz @v12.
lema @v14.
en donde el lema es *andar* o *estar*

dar d @v13.
dar @v14.
NO: ir a Pr 1.

Pr1 Mirar la vocal de la última sílaba de la raíz. ¿Es *a*?
-SÍ: ir a Pr2
-NO: ir a Pr3

Pr2 Mirar la consonante final de la raíz. ¿Es g/c/z?
-SÍ: ir a modificaciones de la consonante y obtener raíz-m.
SALIDA:
lema raíz-m @v10. pagar pagu, sacar saqu, cazar cac
lema @v11. pagar, sacar, cazar

-NO:
SALIDA:
lema @v1. cantar

Pr3 Mirar pres/ind/1a/sg. Si la vocal
e se modifica en *ie,* obtener raíz-m1, ir a Pr4,
o,u, se modifica en *ue,* obtener raíz-m2, ir a Pr4,
o se modifica en *üe*, obtener raíz-m3, ir a Pr4,
i se acentúa, obtener raíz-m4, ir a Pr4,
u se acentúa, obtener raíz-m5, ir a Pr4,
o se modifica en *hue*, obtener raíz-m6, ir a Pr4,
en los otros casos, ir a Pr5.

Pr4 Mirar pres/subj/1a/sg. ¿La raíz: raíz-m1, raíz-m2, raíz-m3 y raíz-m4 termina en consonante g/c/z?
-SÍ: ir a modificaciones de la consonante y obtener respectivamente: raíz-m1-f, raíz-m2-f, raíz-m3-f, raíz-m4-f y raíz-m5-f.

Mirar indf/ind/1a/sg. ¿La raíz: raíz-m1, raíz-m2, raíz-m3 y raíz-m4 termina en consonante g/c/z?
-SÍ: ir a modificaciones de la consonante y obtener:
raíz-f.
SALIDA:
lema raíz-m1 @v6. negar nieg
o lema raíz-m2 @v6. jugar jueg
o lema raíz-m3 @v6. averiguar averigüe
o lema raíz-m4 @v6. ahincar ahínc
o lema raíz-m5 @v6.
o lema raíz-m6 @v6. desosar deshues

lema	raíz-m1-f	@v7.	negar niegu
o lema	raíz-m2-f	@v7.	jugar juegu
o lema	raíz-m3-f	@v7.	avergonzar avergüenc
o lema	raíz-m4-f	@v7.	ahincar ahinqu
o lema	raíz-m5-f	@v7.	
lema	raíz-f	@v8.	negar negu, jugar jugu, avergonzar avergüenc, ahincar ahinqu
lema		@v9.	negar, jugar, avergonzar, ahincar

-en los otros casos:
SALIDA:

lema	raíz-m1	@v4.	apretar apriet
o lema	raíz-m2	@v4.	mostrar muestr
o lema	raíz-m3	@v4.	enfriar enfrí
o lema	raíz-m4	@v4.	actuar actú
lema		@v5.	apretar, mostrar, enfriar, actuar

Pr5 Mirar pres/subj/1a/sg. ¿La raíz termina en la consonante g/c/z?
-SÍ: ir a modificaciones de la consonante y obtener raíz-m.

¿La raíz termina en gu?
-SÍ: ir a modificación de gu y obtener raíz-m5.

Mirar /indf/ind/1a/sg. ¿La raíz termina en gu?
-SÍ: ir a modificación de gu y obtener raíz-m5.
SALIDA:

lema	raíz-m	@v10.	tocar toqu
lema		@v11.	tocar
lema	raíz-m5	@v10.	averigü averiguar
lema		@v11.	averiguar

en otro casos:
SALIDA:

lema		@v1.	votar

Modificaciones de la consonante y de gu:
g (no seguida de *u*) → gu
c → qu
z → c
gu →gü

5.2 Estipulaciones complementarias para el cálculo de entradas

A las entradas definidas por el algoritmo precedente se les agregan las que siguen. Para todo lema no asociado al modelo *v1*, es decir los verbos que presentan alguna irregularidad, la entrada correspondiente a su infinitivo:

lema		/inf/c1/LV.

en donde *LV* es una lista de valores que variarán en función de cada lema (sobre los valores utilizados en las entradas y en los modelos, entre los cuales está *c1*, cf. la Sección 6). Para los lemas *estar, andar* y *dar* se agrega una entrada con la *LV* del ejemplo siguiente:

estar	estoy	/pres/ind/1a/sg/c1/LV.

6 Rasgos y tipos de conjugaciones

La utilización de modelos asociados a lemas y raíces en las entradas permite particionar los diferentes tipos de conjugación y caracterizar cada tipo mediante valores de rasgos. Los rasgos utilizados son (en donde *T* abrevia *Tipo*):

<TC; {c1, c2, c3}>; TC: T de conjugación; c: conjugación.

<TR; {r, ir}>; TR: T de regularidad; r: regulares, ir: irregulares.

<TIRR; {vo, co, voco, ac}>; TIRR: T de irregularidad; vo: vocálica, co: consonántica, voco: vocálica y consonántica, ac: acentual.

<TVO; {ue, ie}>; TVO: T [de irregularidad] vocálica; ue: diptongación en *ue*, ie: diptongación en *ie.*

<TCO; {g, gu, c, z}>; TCO: T [de irregularidad] consonántica; g: modificación en *gu*, gu: modificación en *gü,* c: modificación en *qu*, z: modificación en *ce.*

<TNVO; {o, e}>; TNVO: T [de ausencia de regularidad] vocálica; o: no diptongación de *o*, e: no diptongación de *e*.

Mediante estos rasgos, cuyos valores se incorporan en los modelos y en las entradas y que van a conformar las *LV* juntamente con los valores propios de la morfología, es posible conjugar, si así se lo desea, diferentes subconjuntos de la conjugación de un mismo lema o clase de lemas. Ejemplos en la Tabla 2.

LV	Tipo de conjugación
vo, ue	Formas irregulares de verbos cuya sola irregularidad consiste en la diptongación en *ue* de la raíz (ejemplo: *soñar*).
voco, ue, c	Formas irregulares de verbos cuya irregularidad consiste en la diptongación en *ue* de la raíz y en el cambio c → qu (ejemplo: *tocar*).
o	Formas de verbos que no han diptongado en *ue* (ejemplo: *votar*).

Tabla 2. Rasgos y tipos de conjugación.

7 Conclusiones

Estudiar la relación entre modelización lingüística, su implementación en herramientas declarativas de tratamiento automático y su explotación en tareas aplicativas como, por ejemplo, la enseñanza de lenguas, parece ser una vía promisoria a seguir. La partición de los tipos de conjugación permite ofrecer a los intervinientes en el aprendizaje de lenguas un acceso pormenorizado a los distintos problemas de las irregularidades morfológicas, lo que no excluye un acceso global.

Por otro lado, la explicitación de un algoritmo conceptual organizado en torno a los factores subyacentes que condicionan los tipos de conjugación, permite apuntar un resultado interesante: conociendo las formas del infinitivo de un lema y las primeras personas de los presentes de indicativo y de subjuntivo, y del indefinido, es posible calcular todas las concatenaciones de terminaciones con las raíces, regulares o no, correspondientes. Ejemplos:

amar	amo	ame	amé
regar	*riego*	riegue	regué
volar	*vuelo*	vuele	volé
andar	ando	ande	*anduve*
votar	voto	vote	voté
ahincar	ahínco	ahínque	ahinqué

Más aún: sólo conociendo el infinitivo y la 1ra persona del indicativo de los verbos que diptongan en *ie* o en *ue*, y además, para los verbos *andar, estar* y *dar,* la 1ra persona del indefinido

(en el ejemplo anterior las formas subrayadas), se pueden calcular todas las otras. Creemos entonces que el algoritmo conceptual propuesto puede ayudar a quienes aprenden la morfología del español a construir activamente los procesos que permitirán su utilización, activa y correcta.

Referencias

Salah Aït-Mokhtar. 1998. *L'analyse présyntaxique en une seule étape*. Tesis doctoral. Universidad Blaise-Pascal/GRIL, Clermont-Ferrand.

Salah Aït-Mokhtar. 1995. *SMORPH: Guide d'utilisation. Rapport technique*, Universidad Blaise Pascal/GRIL, Clermont-Fd.

Salah Aït-Mokhtar y José Lázaro Rodrigo Mateos. 1995. Segmentación y análisis morfológico de textos en español utilizando el sistema SMORPH. *SEPLN*, 17: 29-41.

Santiago Alcoba. 1999. La flexión verbal. En Bosque y Demonte (1999: capítulo75).

Iñaqui Alegría. 1995. *Morfología de estados finitos*, Informatika Fakultatea (UPV/EHU).

Bescherelle. 1997. *Les verbes espagnols*. Hatier, Paris.

Ignacio Bosque y Violeta Demonte. 1999. *Gramática Descriptiva de la Lengua Española*. Espasa, Madrid.

J. Harris. 1983. *Syllabe Structure and Stress in Spanish: A non Linear Análisis*. Cambridge, Mass. MIT Press.

Larousse. 1993. *Conjugación,* Larousse, París.

Pierre Le Goffic. 1997. *Les formes conjuguées du verbe français oral et écrit*. OPHRYS, Paris.

Carmen Pensado. 1999. Morfología y fonología. Fenómenos morfofonológicos. En Bosque y Demonte (1999: capítulo 68).

José Lázaro Rodrigo Mateos y Gabriel G. Bès. 2004. Análisis e implementación de clíticos en una herramienta declarativa de tratamiento automático de corpus. En *VI Congreso de Lingüística General,* Santiago de Compostela.

Capítulo 9

COMPACTANDO EL CAST3LB

Demetrio Vilela y Gabriel Infante-López

En Víctor M. Castel, Comp. (2005) *Desarrollo, implementación y utilización de modelos para el procesamiento automático de textos*. Mendoza: Editorial de la Facultad de Filosofía y Letras, UNCuyo: 93-100.
ISBN 987-575-019-0 del soporte Internet

Compactando el Cast3LB

Demetrio Vilela y Gabriel Infante-Lopez

Universidad Nacional de Córdoba
Facultad de Matemática, Astronomía y Física
Córdoba, Argentina
demetrio@hal.famaf.unc.edu.ar // gabriel@famaf.unc.edu.ar

Resumen

Los corpus anotados con estructura sintáctica, como el *Penn Treebank* (PTB) o el 3LB, ofrecen una manera simple de obtener una gramática para realizar el análisis sintáctico automático de la lengua. Las reglas de esta gramática se infieren a partir de las estructuras sintácticas que aparecen en el corpus. En la bibliografía se muestra como el conjunto de reglas extraídas del PTB aumenta a medida que aumenta la cantidad de material (corpus) que se usa para inferir la gramática, sin que este crecimiento llegue a estabilizarse. Este fenómeno sugiere, por un lado, que la cantidad de material necesario para que la cantidad de reglas se estabilice podría ser infinito. Por otro lado, este resultado también sugiere que el PTB posee una alta redundancia en el uso de estructuras. Estos resultados son claramente dependientes del PTB: en otros corpus anotados se observan fenómenos distintos. En este trabajo reproducimos los experimentos de (Krotov et al. 1998) para el 3LB. Discutimos los resultados, y proponemos nuevas direcciones de trabajo futuro.

1 Introducción

Los corpus anotados sintacticamente (*treebanks*) contienen oraciones cuyas estructuras sintácticas han sido explicitadas por anotadores expertos. En particular, el *Penn Treebank* (PTB) (Marcus et. al. 1993) ha sido ampliamente usado para la generación automática de gramáticas: las anotaciones de un *treebank* definen de manera implícita una gramática libre de contexto (CFG); cada subárbol en el banco de árboles define una regla de la siguiente manera. La raíz del árbol define la parte izquierda de la regla y los nodos en el primer nivel del árbol la parte derecha o cuerpo de la regla (Charniak 1996). Aún mas, las gramáticas obtenidas de esta manera ofrecen una precisión de alrededor de 80% (Charniak 1996) cuando son usadas para analizar sintacticamente oraciones no vistas en el *treebank*. Muchas de las reglas obtenidas implícitamente del PTB son redundantes, en el sentido de que su eliminación de la gramática nunca hace que una oración no se pueda analizar si previamente era analizable.

Krotov et al. (1998) mostraron que, para el caso particular del PTB, eliminar un número importante de las reglas obtenidas de esta manera (más del 90%) no altera la *performance* de las gramáticas cuando éstas son utilizadas para análisis sintáctico. Aún más, los autores muestran que compactar una gramática no sólo no empeora la *performance* de la gramática, sino que la incrementa. Los autores atribuyeron la enorme cantidad de reglas redundantes a dos principales razones. Primero, a que las anotaciones no explicitan suficiente subestructura, o, en otras palabras, a la poca profundidad de los árboles en el PTB. Segundo, a la falta de normas claras para los anotadores. Esto daría una mayor consistencia a la anotación y a las reglas usadas. Finalmente, Krotov et al. (1998) observaron que la cantidad de reglas de la gramática subyacente aumenta según la raíz cuadrada de la cantidad de texto analizado; esto implica un crecimiento *no acotado* de la cantidad de reglas necesarias para analizar sintacticamente una oración. Por otra parte, observan que cuando la gramática se compacta, la cantidad de reglas *no redundantes* tiende a un límite a partir de un determinado número de reglas.

Los resultados mencionados hasta ahora dependen del texto analizado y de los criterios empleados para la anotación. Eso implica que la cantidad de reglas redundantes puede variar para

cada esquema de anotación. La cantidad de reglas redundantes refleja el nivel de subestructura que muestran las anotaciones, y por ello, es necesario realizar experimentos comparables a los de Krotov et al. (1998) utilizando como material bancos de árboles distintos al PTB. En este trabajo presentamos un análisis cuantitativo de la cantidad de reglas redundantes para la versión en castellano del 3LB, el Cast3LB (Palomar et. al. 2004)

En este trabajo presentamos los siguientes experimentos. En primer lugar, recorremos el material presente en el Cast3LB y medimos el crecimiento del conjunto de reglas inducidas en función de la cantidad de texto anotado que leemos. Después observamos cómo aumenta la cantidad de reglas no redundantes a medida que incorporamos oraciones del corpus. Finalmente transformamos las anotaciones del Cast3LB en el sentido de simplificar el grado de subestructura sintáctica que exponen. Los resultados confirman la hipótesis de que el nivel de detalle de las anotaciones tiene que ver con el punto hasta donde se puede compactar la gramática. En efecto, las gramáticas transformadas según este artificio se mostraron más redundantes que las inducidas a partir de las anotaciones originales.

El resto del trabajo está organizado como sigue. La Sección 2 discute el algoritmo de eliminación de reglas, la Sección 3 muestra y discute los resultados de nuestros experimentos y finalmente la Sección 4 concluye el trabajo y presenta líneas de trabajo futuro.

2 Eliminación de reglas redundantes

Algunas reglas que aparecen en el Cast3LB pueden deducirse a partir de otras. Formalmente, una regla p: $A \rightarrow \alpha$ es redundante en la gramática G si es posible obtener en G una derivación de α a partir del símbolo A que no utilice la regla p.

Por ejemplo, muchas veces encontramos que una frase como

La cara de la luna

se analiza como muestra la Figura 1.

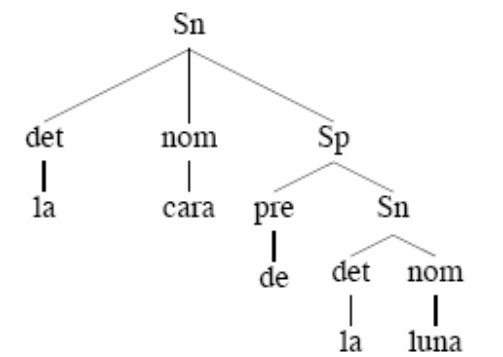


Figura 1. Análisis sintáctico de la fase *la cara de la luna*.

Sin embargo, esta frase también puede analizarse como muestra la Figura 2:

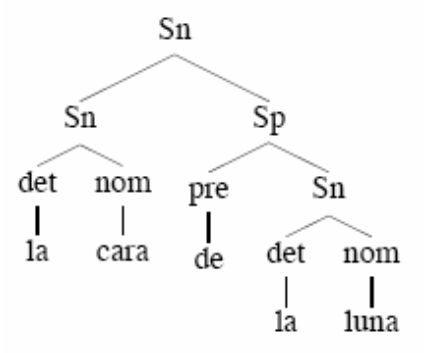


Figura 2. Segundo análisis posible para la frase *la cara de la luna*.

En este caso particular, el segundo análisis es preferible desde el punto de vista lingüístico, porque la última regla explicita la ocurrencia del sintagma nominal. Entonces si la regla *Sn → Sn Sp* perteneciera a un gramática donde además estuvieran las reglas *Sn → Sn Sp* y *Sn → det nom* , la primera podría ser eliminada y su derivación reemplazada por la siguiente: *Sn → Sn Sp → det nom Sp*.

A. Krotov et. al. (1996) argumentan que este fenómeno es común en el caso del PTB y proponen reducir el tamaño de la gramática inducida eliminando las reglas que no permiten analizar oraciones que no puedan ser analizadas usando otras reglas ya presentes en la gramática.

Es importante notar que muchas reglas así eliminadas tienen sentido gramatical, Por ejemplo, en una conjunción:

S **g** *S.co cnj S.co cnj S.co*

resulta más clara la estratificación:

S **g** *S S.co*
S.co **g** *cnj S.co*

Esto hace evidente que la reducción de una gramática hace diferente el análisis de las oraciones siempre en el sentido de hace más jerárquica la estructura a las frases analizadas. Si bien esta reducción no altera el lenguaje lineal de una gramática, esto es, el conjunto de oraciones que se reconocen como pertenecientes a un lenguaje, sí modifica el lenguaje de árboles de la gramática, esto es, la estructura sintáctica que se atribuye a las oraciones. Eso puede llegar a afectar la *performance* del analizador, no una mayor eficiencia y precisión

2.1 Algoritmo de eliminación de reglas

En nuestro trabajo consideramos como los terminales de la gramática las categorías morfológicas mayores que se distinguen en las etiquetas EAGLES del Cast3LB, sin distinciones menores (género, número), salvo para el caso de los verbos, donde distinguimos entre verbos principales, auxiliares y el verbo ser. Incluimos como terminales los símbolos de puntuación. Para los no terminales, usamos las etiquetas de constituyentes del Cast3LB, del tipo *sintagma nominal* o *sintagma preposicional*, sin tener en cuenta las partes de las etiquetas que describen género y número. Tampoco tenemos en cuenta los agrupamientos *especificador* y *grupo*, que pertenecen a un nivel descriptivo distinto al de los constituyentes, ya que, a diferencia de éstos, su distribución está fuertemente limitada por los constituyentes.

Obtenemos la gramática subyacente al Cast3LB de manera directa: para cada nodo en un (sub)árbol de análisis, consideramos que la raíz es un terminal si el nodo es una hoja y si no, agregamos al conjunto de reglas la que resulta de considerar como lado izquierdo a la raíz del

árbol el cuestión y como lado derecho a la secuencia ordenada de las raíces de los subárboles del árbol.

El orden en que se busca eliminar las reglas no altera el resultado de la reducción de la gramática si se parte de una gramática que no tenga reglas unarias o reglas vacías, es decir, si la aplicación de cada regla aumenta estrictamente el largo de la oración analizada. Para garantizar la unicidad del resultado de la gramática reducida, colapsamos la aparición de reglas de ese tipo con las reglas del nivel superior de manera recursiva. Por ejemplo para:

El fumar es perjudicial para la salud

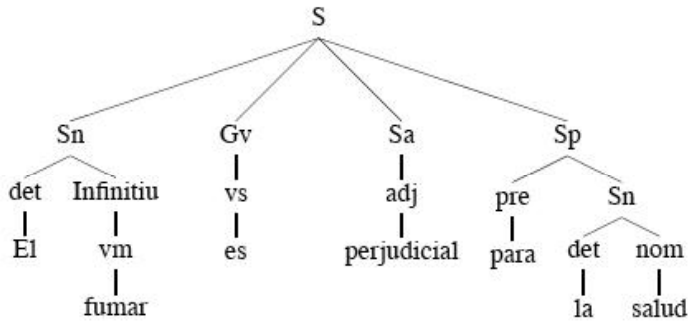


Figura 3. Análisis de la oración *El fumar es perjudicial para la salud*.

Tendríamos las reglas:

S g Sn Gv Sa Sp
Sn g det Infinitivo
Gv g vs
Sa g adj
Infinitivo g vm
Sp g prep Sn
Sn g det nom

Las reglas unarias se eliminan haciendo los correspondientes reemplazos en el nivel superior, quedando:

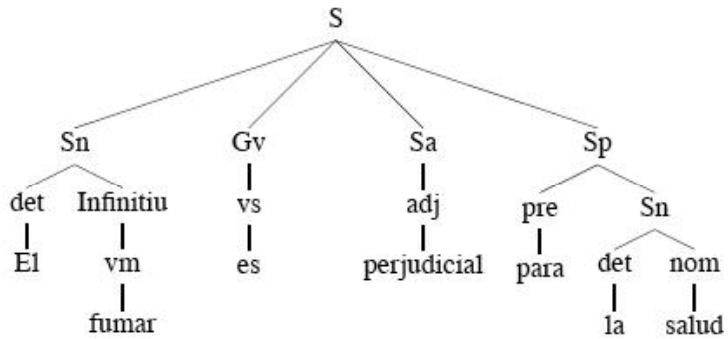


Figura 4. Análisis de la oración *El fumar es perjudicial para la salud* sin reglas unarias.

S g Sn vs adj Sp
Sn g det vm
Sp g prep Sn
Sn g det nom

La idea de redundancia en las reglas nos da una manera simple para reducir el tamaño de una gramática sin que se vea afectada la capacidad de decidir si una oración es analizable o no en castellano.

El algoritmo de eliminación de reglas que usamos recorre un bucle donde trata de encontrar una forma de analizar cada regla a partir de las demás, en cualquier cantidad finita de pasos. Si esto es posible, la regla se elimina; si no, se conserva.

3 Experimentos

En primer lugar describimos cómo crece el tamaño de la gramática inducida en relación con la cantidad de texto anotado leído. Como era esperable, encontramos que el tamaño aumenta de manera no acotada a medida que se toman en cuenta más y más oraciones.

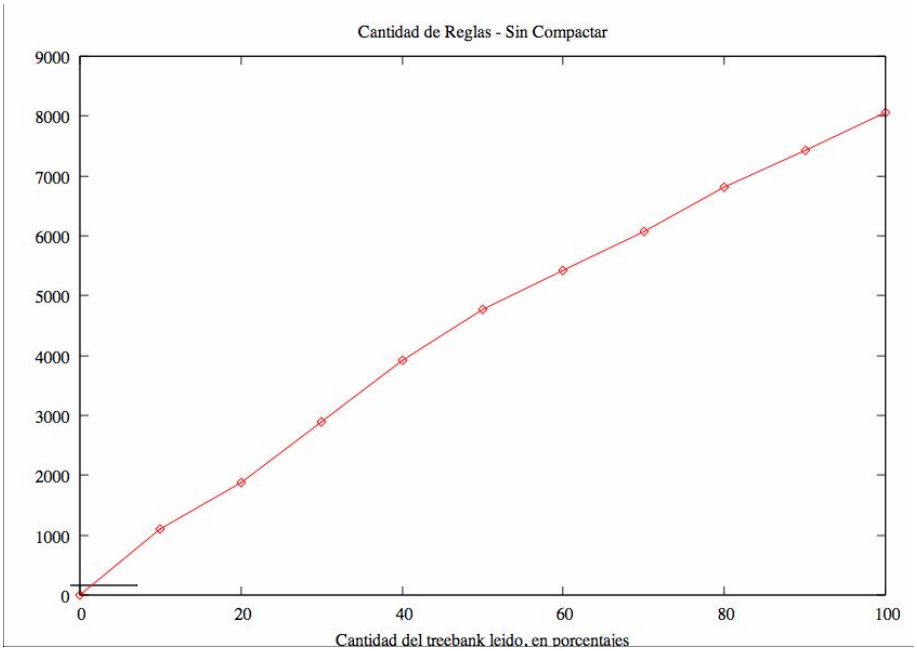


Figura 5. Evolución de la cantidad de reglas, sin compactar la gramática resultante.

Ahora, para nuestro próximo experimento, eliminamos de la gramática las reglas redundantes según se explicó en la Sección 2.1. Para examinar el crecimiento de las gramáticas reducidas a partir de la eliminación de reglas redundantes, dividimos el corpus en diez partes, de acuerdo con el tamaño en *bytes* del texto anotado. La Figura 5 muestra como crece la cantidad de reglas a medida que incorporamos nuevo material al ya usado para inducir la gramática.

Medimos entonces los tamaños de las gramáticas subyacentes y los de las gramáticas resultantes de eliminar las reglas redundantes para el 10%, 20% … 100% del corpus. La Figura 6 muestra el resultado de este experimento.

En contraste con los resultado obtenidos por (Krotov et al. 1998), nuestros experimentos no revelan que este crecimiento tienda a un límite. Suponemos que hay dos razones principales que explicarían estos resultados: por un lado, las anotaciones del Cast3LB son más detalladas que las del PTB, ya que hay un mayor número de categorías no-terminales; por otro, la cantidad de texto analizado en el Cast3LB puede ser insuficiente para mostrar la tendencia a un límite del tamaño de la gramática.

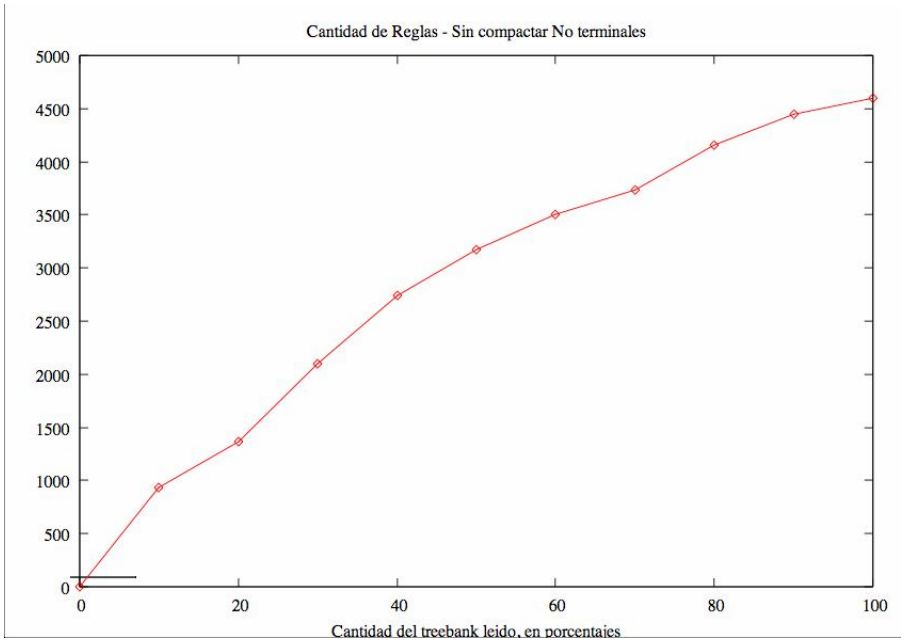


Figura 6. Evolución de la cantidad de reglas, eliminando reglas redundantes, sin compactar los no-terminales.

Para investigar la influencia del grado de subestructura de las anotaciones, repetimos los experimentos simplificando las anotaciones del Cast3LB. Eliminamos los sufijos en los nodos no terminales que indicaban conjunción u otras características menores de la parte de la oración. De esta manera buscamos indicios de que las diferencias respecto de los resultados observados por (Krotov et al. 1998) provienen de las características de las anotaciones en el corpus. La Figura 7 muestra los resultados de estos experimentos.

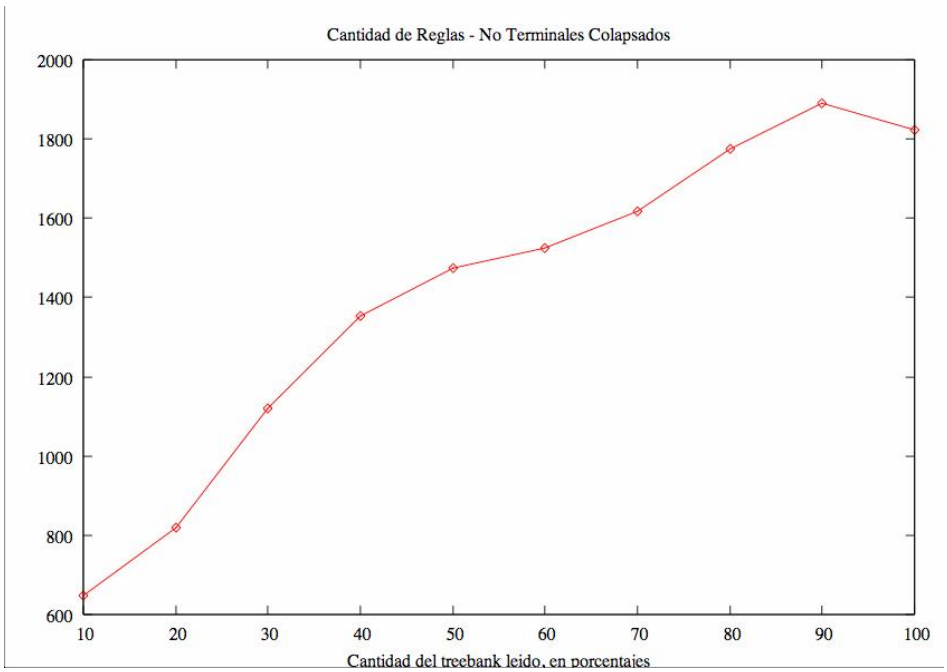


Figura 7. Evolución de la cantidad de reglas, eliminando reglas redundantes, sin compactar los no-terminales.

En este caso podemos observar una reducción de la tasa de crecimiento de la gramática reducida. Incluso en el último paso observamos un reducción del tamaño efectivo de la gramática resultante, manifestando que la adición de reglas nuevas hace redundantes reglas encontradas con anterioridad. Esto sugiere que el tamaño de la gramática reducida podría aproximarse a un límite.

Encontramos esta vez que el crecimiento de la gramática podría estabilizarse alrededor de las 1800 reglas, una cantidad que concuerda con la encontrada por (Krotov et al. 1998) para el caso del PTB y el inglés.

4 Conclusiones y trabajo futuro

Una de las implicaciones directas de nuestros experimentos está en que la estructura de los no-terminales usados en la anotación es de gran importancia. Cuanto más refinando es el conjunto de no-terminales, menos se puede compactar la gramática. Este efecto es una consecuencia directa de la pequeña cantidad de árboles que hay en el Cast3LB. La escasa cantidad de reglas dificulta que el número de instancias de cada regla sea significativo. Por otro lado, y relacionado con lo anterior, la calidad de análisis sintáctico depende de la calidad de los no-terminales (G. Infante-Lopez and M. de Rijke 2004). Como trabajo futuro, queremos estudiar profundamente la relación entre la granularidad de los no-terminales y la *performance* de las gramáticas compactadas cuando éstas son utilizadas para análisis sintáctico.

Los experimentos mostrados en este trabajo tratan el compactamiento de las distintas gramáticas con respecto a sus lenguajes lineales. Nuestros experimentos no tienen en cuenta la manera en la que el algoritmo de compactación presentados en la Sección 3 modifica el lenguaje de árboles. Claramente, para aplicaciones como análisis sintáctico, el lenguaje de árboles es más que fundamental. Como trabajo futuro inmediato planeamos realizar experimentos que indiquen cómo se modifica el lenguaje de árboles que describen las gramáticas propuestas en este trabajo. La mejor manera de realizar este tipo de análisis es analizando sintacticamente una parte de las oraciones del Cast3LB que no hayan sido utilizadas para inducir las gramáticas con las que serán analizadas.

Referencias

E. Charniak. 1996. Tree-bank Grammars. En *Proceedings* AAAI ’96, Portland, Oregon.

G. Infante-Lopez and M. de Rijke. 2004. Alternative approaches for generating bodies of grammar rules. In *Proceedings of the 42nd Annual Meeting of the ACL*, Barcelona.

A. Krotov, M. Hepple, R. J. Gaizauskas, and Y. Wilks. 1998. Compacting the Penn treebank grammar. In *Proceedings of the 36th Annual Meeting of the ACL and 17th International Conference on Computational Linguistics COLING*, 699–703.

M. Marcus, B. Santorini, and M. Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn treebank. *Computational Linguistics*, 19: 313–330.

M. Palomar, M. Civit, A. Díaz, L. Moreno, E. Bisbal, M. Aranzabe, A. Ageno, M. A. Martí, and B. Navarro. 2004. 3LB: Construcción de una base de datos de árboles sintáctico-semánticos para el catalán, euskera y español. In *XX Congreso de la Sociedad Española para el Procesamiento del Lenguaje Natural (SEPLN)*, 81–88.

APÉNDICE
RESÚMENES DE TALLERES

Taller 1

LA OBTENCIÓN DE LÍMITES DE ORACIONES

Celina Beltrán y Gabriel G. Bès

En Víctor M. Castel, Comp. (2005) *Desarrollo, implementación y utilización de modelos para el procesamiento automático de textos*. Mendoza: Editorial de la Facultad de Filosofía y Letras, UNCuyo: 101-104.
ISBN 987-575-019-0 del soporte Internet

La obtención de límites de oraciones

Celina Beltrán

Universidad Nacional de Rosario / INDEC
Facultad de Ciencias Agrarias
Rosario, Argentina
beltranc@dat1.net.ar

Gabriel G. Bès

Universidad Blaise-Pascal
Groupe de Recherche dans les Industries de la Langue (GRIL)
Clermont-Fd., Francia
Gabriel.Bes@univ-bpclermont.fr

Resumen

El Taller trata la problemática de la obtención de límites de oraciones, etapa básica y clave del análisis automático de textos. Se presenta la técnica estadística de máxima entropía (ME) y se la utiliza efectivamente sobre textos del español. Para ello, después de presentada la problemática y los elementos del modelo, se hacen ejercicios prácticos sobre obtención de textos de entrenamiento, descripción de rasgos contextuales, utilización del algoritmo GIS (*Generalized Iterative Scaling*), utilización de fórmulas estadísticas, cálculos sobre los parámetros estimados. Especial atención se consagra a una doble evaluación del sistema, según la metodología estadística utilizada y según un análisis crítico de los resultados obtenidos, análisis que permite predecir los resultados obtenibles.

1 La problemática y sus herramientas

La obtención de los límites de oraciones es un problema clave para todo análisis automático que no se limite a trabajar sobre oraciones dadas como ya delimitadas, sin que sus límites hayan sido obtenidos mediante una metodología explícita (Manning y Schütze 1999). La problemática ha sido abordada mediante la técnica estadística de Máxima Entropía (ME), la que ha sido aplicada a textos en inglés (Reynar y Ratnaparkhi 1997), pero que utiliza herramientas informáticas disponibles en la *web*:

http://www.id.cbs.dk/~dh/corpus/tools/MXTERMINATOR.html

Estas herramientas son presentadas por sus autores como aplicables a textos de otras lenguas; cf. en (Silla Jr. y Kaestner 2004) su utilización para el portugués. Estas herramientas, en el estado en que se presentan, no son declarativas, y, por lo tanto, los rasgos contextuales elegidos no pueden ser modificados.

2 Organización del Taller

El Taller ha sido organizado en función del diagrama que sigue en la Figura 1, en donde se especifican las principales etapas del tratamiento del problema.

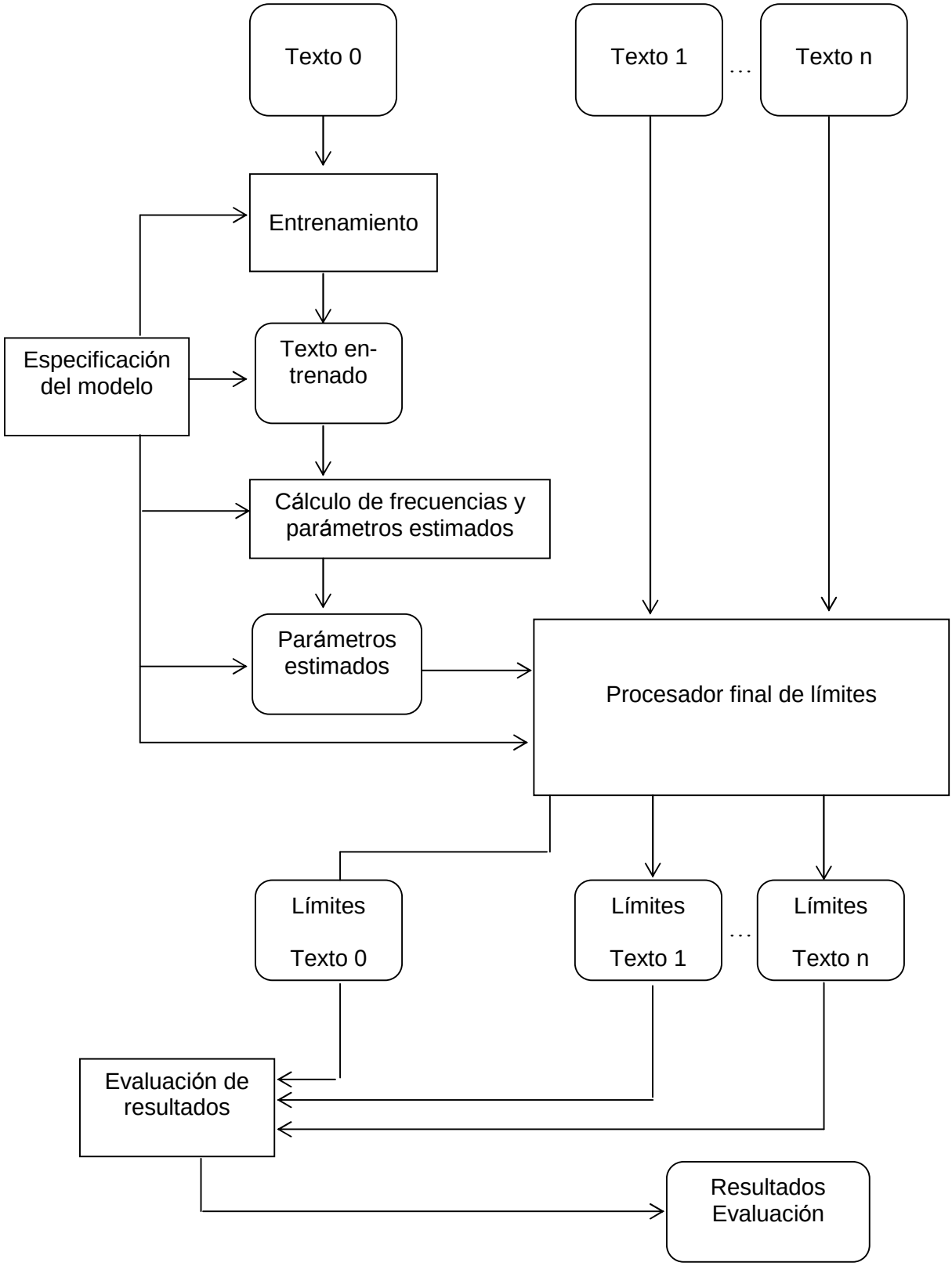


Figura 1. Diagrama de organización del taller.

3 Actividades del taller

La sucesión de actividades del Taller es la que sigue.

1) Problema de la detección del límite de oraciones. Importancia y dificultades. Presentación de la técnica estadística de máxima entropía.

2) Esquema general de una técnica estadística; aplicación a la detección de límites. Doble tipo de evaluación: metodología estadística y resultados obtenidos. Diferenciación de técnicas estadísticas de conteos simples.

3) Textos de base y de entrenamiento. Ejemplificación con el texto T. Candidatos. Ejemplificación de ocurrencias de candidatos en el texto T.

4) Elección de rasgos. Contexto de los candidatos. Descripción de ocurrencias de contextos de candidatos en textos.

5) Expresión en fórmula de los contextos de candidatos. Frecuencias.

6) Nociones técnicas. Nociones matemáticas, entropía y probabilidades.

7) Modelo de máxima entropía. Fórmulas utilizadas.

8) De las frecuencias a los parámetros estimados por el algoritmo GIS (*Generalized Iterative Scaling*). Presentación de las frecuencias y parámetros estimados en el texto T.

9) Ejercicios de cálculo sobre los parámetros estimados a partir del texto T.

10) Evaluación de la utilización de la técnica estadística de máxima entropía en función de la metodología utilizada y de los resultados obtenidos y obtenibles.

11) Evaluación en función de la metodología. Puntos considerados:
- elección de rasgos;
- decisiones sobre frecuencias marginales mínimas;
- aceptación final o no de un candidato como límite de oración.

12) Evaluación en función de los resultados. Bajas frecuencias de los rasgos definitorios de los contextos. Alto porcentaje de decisiones fundadas en los resultados por *default*.

13) Ejercicios de modificación del texto de base y obtención de diferentes textos de entrenamiento. Cálculo de resultados predecibles. Verificación posible en máquina.

14) Discusión y perspectivas.

Referencias

A. Gelbukh. 2004. *CICLing 2004*, LNCS 2945. Springer-Verlag, Berlin Heidelberg.

Christopher D. Manning y Hinrich Schütze. 1999. Foundations of Statistical Natural Language Processing. Cambridge Mass., The MIT Press.

Jeffrey C. Reynar y Adwait Ratnaparkhi. 1997. A Maximum Entropy Approach to Identifying Sentence Boundaries. En ANLP 1997; arXiv:cmp-lg/9704002 v1.

Carlos N. Silla Jr. y Celso A. A. Kaestner. 2004. An Analysis of Sentence Boundary Detection Systems for English and Portuguese Documents. En Gelbukh (2004: 135-141).

Taller 2

EL GRIAL. INTERFAZ COMPUTACIONAL DE ANOTACIÓN E INTERROGACIÓN DE CORPUS EN ESPAÑOL: ALGUNOS RESULTADOS DE SU APLICACIÓN

Giovanni Parodi S.

En Víctor M. Castel, Comp. (2005) *Desarrollo, implementación y utilización de modelos para el procesamiento automático de textos*. Mendoza: Editorial de la Facultad de Filosofía y Letras, UNCuyo: 105-106.
ISBN 987-575-019-0 del soporte Internet

El Grial. Interfaz computacional de anotación e interrogación de corpus en español: algunos resultados de su aplicación

Giovanni Parodi S.

Pontificia Universidad Católica de Valparaíso
Instituto de Literatura y Ciencias del Lenguaje
Valparaíso, Chile
gparodi@ucv.cl

Resumen

El Grial es una interfaz computacional que permite tanto la realización de anotaciones morfosintácticas en textos planos en lengua española como la interrogación o consulta en forma de base de datos de los corpora allí reunidos. En el marco de este Taller se da cuenta de los objetivos para la creación de esta herramienta, sus características y funcionalidades y se exponen los componentes esenciales. También se describen los corpora de base que dan sustento al sitio, los cuales son parte de las investigaciones PUCV y están disponibles para indagación. Otro aspecto destacable lo constituye la opción de acceso que ofrece esta herramienta informática para levantar, etiquetar y consultar otros corpora de manera gratuita y en línea durante un período de tiempo determinado. Como una forma de mostrar la utilidad de este recurso informático, se entrega el marco referencial y los resultados de una investigación en lingüística de corpus en la que se indagan diferencias entre registros orales y escritos, especializados y no especializados.