ARTICULOS

Un Supresor de Ambigüedades Léxicas mediante Métodos Estadísticos

Javier Andrade Garda
María Concepción Álvarez Lebredo
Alberto Valderruten Vidal
Susana Sotelo Docío

Resumen

Presentamos el trabajo realizado para conseguir un sistema que, integrado en el proyecto GALENA (Generador de Analizadores de LEnguajes NAturales) [9, 10], logre eliminar, de forma estadística, las ambigüedades que provocan las palabras al ser tratadas por el módulo de análisis léxico.

El análisis léxico proporciona a las palabras unas etiquetas, es decir, descripciones que contienen toda la información (tipológica, morfológica...) necesaria para caracterizarlas en el conjunto del léxico de la lengua. A cada palabra aislada, fuera de contexto, pueden corresponderle varias etiquetas. Esta situación imposibilita el análisis sintáctico de los textos. Surge, por tanto, la necesidad de eliminar las ambigüedades en el proceso de etiquetación.

El objetivo del presente trabajo es proveer de una única etiqueta a cada palabra, asignando la que en cada caso sea más probable según la historia del texto en estudio. Para ello, es necesario un análisis estadístico de textos del mismo estilo literario que el que va a ser tratado, y la aplicación de ese estudio al texto en cuestión.

El supresor de ambigüedades que proponemos permite la selección, por parte del usuario, de la información léxica a utilizar en la desambiguación, y ofrece funcionalidades complementarias para el tratamiento de las matrices de aprendizaje.

Palabras clave: etiquetación léxica, ambigüedad, estadística, matriz de aprendizaje, desambiguación.

1 Introducción

La etiquetación se realiza de forma automática. Es el método más natural si tratamos textos de gran tamaño. Para evitar la necesidad de un retoque manual de las etiquetas que proporciona el analizador, cuando nos encontramos ante algoritmos de etiquetación automática que no cuentan con las suficientes evidencias para eliminar ambigüedades, resulta práctico conservar más de una etiqueta por palabra encontrada en el texto. Nuestro analizador léxico sigue esta estrategia. Siempre etiqueta cada token con todas las posibilidades que ofrece. Esta es una de sus características más importantes, porque es la que permite poner de manifiesto las ambigüedades.

Lo que persigue un supresor de ambigüedades es seleccionar, de entre todas las etiquetas posibles de una palabra, la más probable en función del contexto en que aparece dentro del texto en consideración.

Para lograrlo, en una primera fase se entrena el sistema con textos, denominados textos de aprendizaje, del mismo estilo literario que el texto que se vaya a tratar. Estos textos son etiquetados por lingüistas, de tal forma que no presenten ambigüedades. El procedimiento

J. Andrade Garda y A. Valderruten Vidal pertenecen al Laboratorio de Fundamentos de la Computación e Inteligencia Artificial, Departamento de Computación, Universidad de A Coruña, Campus de Elviña s/n, 15071 La Coruña, España. Email: {andrade, valderruten}@dc.fi.udc.es.

S. Sotelo Docío y M. C. Álvarez Lebredo pertenecen al Departamento de Filología Española, Universidad de Santiago de Compostela, Burgo de las Naciones s/n, 15705 Santiago de Compostela, España. Email: {fesdocio, femcal}@usc.es.

Este trabajo está parcialmente financiado por los proyectos XUGA20403B95 y XUGA10505B96 de la *Xunta de Galicia*.

que sigue nuestro sistema consiste básicamente en obtener de los textos de aprendizaje, las frecuencias de aparición de las etiquetas en relación al contexto. Posteriormente, esas frecuencias de aprendizaje se aplican al texto en estudio y seleccionan la etiqueta más probable para cada palabra ambigua. Los resultados de la selección estarán avalados por la fase de aprendizaje.

En consonancia con lo expuesto, el sistema que aquí se presenta consta de dos módulos para lograr una operatividad total. El primero constituye la fase de aprendizaje y el segundo la operacional.

2 El problema de la etiquetación

Supondremos, para exponer el problema que se plantea en este punto, que el usuario ha definido un conjunto de etiquetas vinculadas a las palabras. El juego de etiquetas que se ha considerado en este proyecto se puede ver en la tabla 1.

Campo	Valores posibles	
Word	*La palabra en la forma introducida.*	
Lemma	*La forma canónica de la palabra.*	
Category	Adjective	*Sin tipo.*
	Adverb	exclamative, modifier, nuclear, nuclear & modifier, interrogative *y* relative.
	Article	*Sin tipo.*
	Conjunction	coordinate *y* subordinate.
	Demonstrative	*Sin tipo.*
	Indefinite	*Sin tipo.*
	Interjection	*Sin tipo.*
	Interrogative	*Sin tipo.*
	Numeral	cardinal, ordinal, partitive *y* multiple.
	Peripheral	foreign word, formula, abbreviation, symbol, acronym *y* other.
	Preposition	*Sin tipo.*
	Personal Pronoun	tonic, proclitic atonic *y* enclitic atonic.
	Possessive	*Sin tipo.*
	Punctuation Mark	dot, comma, semicolon, colon, open ¡ close question mark, open ¡ close exclamation mark, open ¡ close parenthesis, dash, quotes *y* dots.
	Relative	*Sin tipo.*
	Substantive	common *y* proper.
	Unknown	*Sin tipo.*
	Verb	*Sin tipo.*
Subtype	determiner, non-determiner *y* both.	
Gender	masculine, feminine, both, neutral *y* non-applicable.	
Number	singular, plural, both *y* non-applicable.	
Degree	comparative *y* non-applicable.	
Person	first, second, third, first & third *y* non-applicable.	
Person number	singular, plural, both *y* non-applicable.	
Case	nominative, accusative, dative, accusative & dative, case with preposition *y* nominative & case with preposition.	
Verbal tense	present, preterite, copreterite, copreterite_se, future, postpreterite, antepresent, antepreterite, antecopreterite, antecopreterite_se, antefuture, antepostpreterite *y* non-applicable.	
Mode	indicative, subjunctive, imperative, infinitive, compound infinitive, gerund, compound gerund *y* participle.	

Tabla 1: Juego de etiquetas.

Considérese una secuencia $W = w_1 w_2 ... w_n$, donde cada w_i es una palabra, y una secuencia de etiquetas $T = t_1 t_2 ... t_n$. Llamemos al par (W, T) una alineación (*alignment* para Bernard Merialdo [6]). Se dice que a la palabra w_i se le ha asignado la etiqueta t_i en esta alineación.

Se asume que las etiquetas tienen algún significado para el usuario, de forma que entre todas las posibles alineaciones para una secuencia sólo hay una correcta desde el punto de vista gramatical. El objetivo final en la eliminación de ambigüedades es encontrar la

alineación correcta haciendo uso, en nuestro caso, de la información adquirida en la fase de aprendizaje.

Un procedimiento de etiquetación (*tagging procedure* para Bernard Merialdo [6]) es un procedimiento o que selecciona una secuencia de etiquetas, esto es, define una alineación para la secuencia W dada. La forma de expresarlo formalmente sería:

$$o : W \rightarrow T = o(W)$$

Hay al menos dos medidas para evaluar la calidad de un procedimiento de etiquetación[1]:

- a nivel de frase:

$$Q_S(o) = \text{porcentaje de frases correctamente etiquetadas.}$$

- a nivel de palabra:

$$Q_W(o) = \text{porcentaje de palabras correctamente etiquetadas.}$$

En la práctica $Q_S(o)$ es generalmente menor que $Q_W(o)$, puesto que, para que una frase sea considerada como correctamente etiquetada, todas las palabras que la componen deben estar etiquetadas de forma adecuada. Por esta razón $Q_S(o)$ no es una medida totalmente fiable del funcionamiento del procedimiento de etiquetación.

El sistema que presentamos se puede definir como un procedimiento de etiquetación, y la medida de calidad que consideraremos es $Q_W(o)$, que es la medida estándar usada en la literatura.

3 Fase de Aprendizaje

En esta fase se parte de un texto sin ambigüedades léxicas, resultado del trabajo de un equipo de lingüistas. A partir de este texto se elaboran las estadísticas que sirven para suprimir las ambigüedades que puedan aparecer en la etiquetación de una palabra.

Para realizar el estudio estadístico se eligió una ventana temporal de dos tokens que mantiene la historia del texto. Esa historia permite elegir entre las varias alternativas de etiquetación que puede tener una palabra a la salida del analizador léxico.

3.1 Selección del tamaño de la ventana temporal

Church y Mercer [1] afirman que la etiqueta de un token se puede estimar teniendo en cuenta sólo las etiquetas de los dos tokens anteriores, debido a que el cálculo en base a los n tokens anteriores es demasiado complejo. En el caso del inglés, esta aproximación parece suficiente; se puede asegurar que la etiqueta de un token depende únicamente de las dos anteriores. Pese a que no está demostrado para otros idiomas como, por ejemplo, las lenguas latinas, esa aproximación ha servido de base para muchos trabajos, entre los cuales se encuentra el nuestro.

En la mayoría de los sistemas se utilizan ventanas de un único token puesto que la cantidad de datos que hay que almacenar es considerablemente menor. En ocasiones, se

[1] calidad se representa mediante Q, del inglés *quality*.

utilizan ventanas de dos tokens, que se denominan trigramas o trietiquetas. Podríamos pensar en ampliar la ventana temporal; sin embargo, se han realizado experimentos utilizando ventanas mayores y la mejoría de los resultados no compensa la cantidad de datos que hay que almacenar y utilizar. Además, para hacer uso de 4-gramas, o n-gramas en general, en el aprendizaje, el tamaño del corpus debería ser muchísimo mayor que el necesario para obtener resultados aceptables con bigramas o trigramas. Con una ventana de dos tokens, la cantidad de datos que es necesaria entra dentro de los límites de lo razonable. No ocurre lo mismo con ventanas de tres o más tokens.

En nuestro sistema, como se ha indicado anteriormente, hemos elegido las trietiquetas, que proporcionan una información útil para desambiguar, pero sin pagar un precio excesivo en el almacenamiento y manejo de datos.

3.2 Resultado de la fase de aprendizaje

Como resultado de la fase de aprendizaje se crea una matriz que refleja las situaciones que se han producido en el texto. La estructura creada debe ser una matriz debido a que la historia del texto se mantiene en una ventana temporal de dos tokens. Las dimensiones de la matriz se corresponden con cada uno de los tokens.

En cada acceso a la matriz pueden presentarse distintas posibilidades: para la misma ventana temporal, que fija el acceso a la matriz, la situación actual puede ser distinta de las anteriores, o bien puede repetirse. Esto obliga a mantener una dimensión más en la matriz de aprendizaje, que refleja todas las situaciones aprendidas, así como el número de veces que han aparecido, posibilitando la fase operativa posterior basada en esas estadísticas.

Al finalizar el proceso, se generará un fichero que contendrá la matriz de aprendizaje. Esto posibilita:

- que se puedan generar matrices de aprendizaje para los diferentes estilos literarios que se deseen tener en cuenta, y
- que realizando el proceso de aprendizaje una única vez, la matriz se pueda utilizar en diferentes momentos de la fase operacional.

El código desarrollado para esta fase permite combinar varios textos en los que ya no se presentan ambigüedades, para generar una sola matriz de aprendizaje. De esta forma se consigue:

- no tener que almacenar todo un corpus en un único fichero[2], y
- poder reflejar, en una única matriz, el aprendizaje aportado por varios textos que posean estilos literarios similares.

3.3 Desarrollo del sistema

Para desarrollar el código necesario en el aprendizaje, inicialmente se utilizó el lenguaje de reconocimiento de patrones GAWK [4] y el lenguaje de programación C [3]. Los tiempos obtenidos en la generación de matrices de aprendizaje resultaron bastante elevados, como se puede apreciar en las estadísticas presentadas en la tabla 2.

[2]el aprendizaje debe estar avalado por el estudio de una amplia casuística, lo cual significa manejar textos de aprendizaje muy grandes.

En la búsqueda de una solución alternativa también nos pareció conveniente que el usuario pudiese decidir, sin ningún tipo de restricción, qué campos de las etiquetas léxicas quisiera considerar en el proceso de desambiguación. Permitimos así al usuario que determine qué información le interesa tener en cuenta para llevar a cabo el proceso, en lugar de imponérsela. Esta funcionalidad resulta además muy útil para controlar la cantidad de datos que el sistema debe almacenar. En el caso de considerar toda la información posible, la matriz de aprendizaje tendría un tamaño aproximado de 5700^2 posiciones, sin tener en cuenta la tercera dimensión, que es variable y depende del volumen del corpus utilizado para el aprendizaje. Una matriz de tales proporciones resulta poco conveniente.

En relación a las etiquetas léxicas a considerar en la desambiguación, hemos comprobado que seleccionando sólo la información de *Category* y *Type*, la desambiguación no hila tan fino como cabría esperar. Por ejemplo, en los casos en que hay que discernir entre varias etiquetas de una misma palabra, de las cuales varias (o todas) corresponden a la categoría *Verb*, lo interesante sería tener en cuenta los campos *Person*, *Verbal tense* y *Mode*, por proponer algunos. También se podrían considerar *Gender* y *Number* en las categorías *Personal Pronoun* y *Substantive* para que en el proceso de desambiguación se tenga en cuenta la concordancia que debe existir entre género y número.

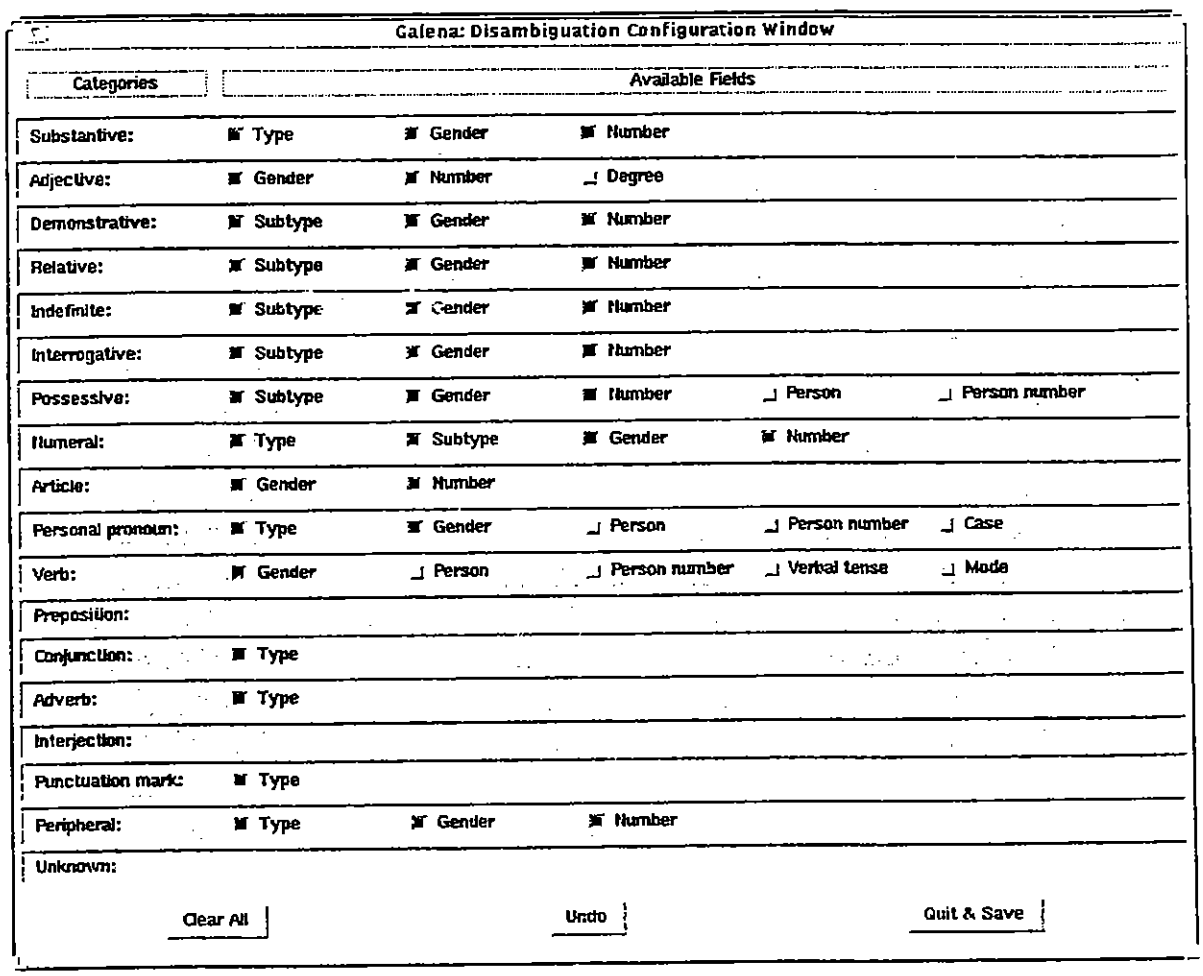

Figura 1: Ventana de configuración de la desambiguación.

Una vez que el usuario decida qué información quiere considerar, se genera automáticamente el código C adecuado para permitir la desambiguación. Todo esto se realiza a través de una interfaz gráfica, creada con TCL/TK [7], donde el usuario puede, de forma sencilla e intuitiva, tomar sus decisiones (figura 1). Acto seguido, se genera en el

sistema, de forma totalmente transparente, un fichero de configuración, que será tratad con FLEX y BISON[3] para generar el código C que mencionamos.

El fichero de configuración al que nos acabamos de referir no sería necesario si usásemo otra herramienta que permitiese generar interfaces gráficos con la posibilidad de integra el código C. Hemos decidido, sin embargo, usar las herramientas descritas para no limita nuestro sistema al uso de estaciones gráficas. Con este fichero mantenemos la tota operatividad del sistema aún en el caso de poseer sólo terminales de texto. Basta co editarlo y colocar un "-" a continuación de la información que no deseamos.

3.4 Tiempos de aprendizaje

Las estadísticas que se presentan en la tabla 2 han sido obtenidas con el comando tim de UNIX en una SUN SPARC STATION 10 bajo SUNOS 4.1.3 y reflejan el tiempo d generación de la matriz de aprendizaje, contrastando la versión que utilizaba GAWK y C con la versión que hace uso de TCL/TK, FLEX, BISON y C.

Tamaño de fichero de aprendizaje		Primera versión	Versión actual
en palabras	en bytes aprox.	tiempo de usuario en seg.	tiempo de usuario en seg.
Sin texto	0	53,5	0,0
12	*1.400*	53,9	0.0
120	*14.000*	54.0	0,0
240	*28.000*	53,2	0,1
960	*111.900*	55,5	0,3
1.920	*223.700*	56,1	0,6
3.840	*447.400*	57,5	1,1
15.360	*1.800.000*	65.2	4,7
30.720	*3.600.000*	78,6	9,4
61.440	*7.000.000*	90,3	18.6

Tabla 2: Estadísticas de la fase de aprendizaje.

Como se puede apreciar en esta tabla, los tiempos se han reducido de tal forma que, e la actualidad, la generación de la matriz sigue una función linealmente creciente frente a tamaño del texto de aprendizaje.

3.5 Reducción y fusión de matrices de aprendizaje

Además de lo expuesto para la fase de aprendizaje, hemos desarrollado dos funcion adicionales —reducción y fusión— que pasamos a describir.

El módulo de reducción permite reducir una matriz de aprendizaje de tal forma qu sólo nos quedemos con la información de aprendizaje que supere o iguale una calidad qu el propio usuario puede especificar en términos de porcentajes de aparición. Esto provoc que si se consulta una matriz de aprendizaje reducida, para deshacer una ambigüeda que se presente, puedan darse dos situaciones:

1. o bien, la eliminación de la ambigüedad no es posible, lo que se debería a que en matriz inicial ninguna alternativa supera la calidad solicitada por el usuario;
2. o bien, el proceso de desambiguación sólo tiene en cuenta la situación presentad por la matriz: la entrada de la matriz sólo refleja la alternativa que supera la calid impuesta por el usuario.

[3]herramientas de GNU equivalentes a LEX y YACC [5].

En el primero de los casos anteriores, la matriz reducida no nos permite discernir qué etiqueta es la más probable (se elegirá la primera para continuar con el proceso). En el segundo, se elegirá de forma definitiva la alternativa que presente la matriz reducida, siempre que dicha alternativa sea una de las posibles para la palabra que presentó la ambigüedad.

Lo que se consigue con esta reducción de la matriz de aprendizaje es una mayor agilidad a la hora de elegir entre las posibles etiquetas de una palabra: o no se puede deshacer la ambigüedad, o bien sólo hay que determinar si la única alternativa que se refleja al indexar la matriz, se encuentra entre las varias posibilidades de etiquetación que presenta la palabra. En definitiva, lo que se logra es mejorar el tiempo de respuesta y tomar una elección absoluta.

El segundo módulo permite fusionar, en una única matriz final, varias matrices que hayan sido generadas con anterioridad. En realidad, a través de este bloque se posibilita una suma de matrices, o lo que es equivalente, una suma de aprendizajes.

Como punto final, se deben recalcar las ventajas que ofrece guardar en un fichero la matriz que se genera:

- Cada estilo literario puede corresponderse, en el sistema, con una matriz que se haya generado a partir de textos de ese mismo estilo. Ponemos tanto énfasis en los diferentes estilos de los textos porque, obviamente, los patrones de escritura difieren de una novela a un artículo periodístico, por poner dos ejemplos. Los datos que se desprendan del aprendizaje sobre un estilo y otro serán también claramente diferentes.
- El propio usuario puede tener varias matrices en las que la información léxica de las etiquetas que se toma en consideración difiera de unas a otras. Las matrices resultan propicias para aplicar sobre ellas operaciones, como la de suma o fusión de matrices, de la que ya hemos hablado.

4 Fase operacional

Una vez finalizada la fase de aprendizaje, poseemos una matriz generada a partir de un texto desambiguado, que nos permitirá abordar la presente fase operacional.

El código necesario para esta fase se ha incorporado en un programa que permite visualizar los resultados del analizador léxico y usar la opción de supresión de ambigüedades. En caso de invocar esta opción, se deberá especificar la matriz de aprendizaje que se usará para el proceso. Esta matriz se cargará en memoria y el sistema ya estará listo para proceder a eliminar las ambigüedades léxicas que se encuentren en el tratamiento del texto que se está estudiando.

El analizador léxico muestra todas las posibles alternativas de una palabra, y al final de cada etiqueta da un número que indica la seguridad de esa etiqueta según el aprendizaje realizado. Cuanto más bajo sea este número[4], más alta será la confianza que ofrece esa etiqueta, siempre de acuerdo con la matriz de aprendizaje que haya indicado el usuario.

Existen situaciones, como hemos explicado en apartados anteriores, en que la matriz no nos permite deshacer las ambigüedades. En estos casos se escogerá la primera etiqueta para poder seguir con el proceso.

[4] el límite inferior es 1.

Hay que destacar que algunos elementos léxicos, como los pronombres enclíticos o las preposiciones de varias palabras[5], no ofrecen la posibilidad de varias etiquetaciones, es decir, son casos en que no existe ambigüedad. Este hecho ha de tenerse en cuenta. También hay que tener presentes otros casos en los que hay que hacer una gestión especial de los tokens que devuelve el analizador léxico. Por ejemplo, la palabra *velo* presenta dos posibilidades de etiquetación:

1. *ve* (verbo ver) + *lo* (pronombre enclítico)
2. *velo* (paño utilizado para cubrir la cara)

Si se escoge la segunda alternativa, la siguiente llamada al analizador léxico deberá ignorarse, puesto que devolverá un pronombre enclítico que no tiene sentido ante la elección tomada. Esto no ocurrirá si, guiados por la matriz de aprendizaje, escogemos la primera alternativa. En las líneas siguientes presentamos éste y otros ejemplos similares tal y como se presentan en la salida de nuestro analizador léxico. El analizador ha sido invocado con la opción de desambiguación y con una matriz de aprendizaje creada con anterioridad:

1. Palabra introducida: *velo*

```
velo
=> ["ve", Verbo, 2da, sg, presente, imperativo, prioridad 1, "ver"]
   ["ve", Verbo, 2da, sg, presente, imperativo, prioridad 1, "ir"]
   ["velo", Verbo, primera, sg, presente, indicativo, prioridad 4, "velar"]
   ["velo", Sustantivo comun, masc, sg, "velo"]
=> ["lo", Pron Per encl ato, 3ra, sg, acus, masc, prioridad 1, "'el"]
```

También podríamos tener, si así lo indicase el aprendizaje, la salida siguiente:

```
velo
=> ["ve", Verbo, 2da, sg, presente, imperativo, prioridad 3, "ver"]
   ["ve", Verbo, 2da, sg, presente, imperativo, prioridad 3, "ir"]
   ["velo", Verbo, primera, sg, presente, indicativo, prioridad 3, "velar"]
   ["velo", Sustantivo comun, masc, sg, prioridad 1, "velo"]
```

2. Palabra introducida: *ténselo*

```
t'enselo
=> ["ten", Verbo, 2da, sg, presente, imperativo, prioridad 1, "tener"]
   ["t'ense", Verbo, 2da, sg, presente, imperativo, prioridad 1, "tensar"]
=> ["se", Pron Per encl ato, 3ra, sg y pl, acus y dat, masc y fem, prioridad 1,
=> ["lo", Pron Per encl ato, 3ra, sg, acus, masc, prioridad 1, "'el"]
```

La etiquetación, basada en la matriz de aprendizaje, también podría ser como indica a continuación:

```
t'enselo
=> ["t'ense", Verbo, 2da, sg, presente, imperativo, prioridad 2, "tensar"]
   ["ten", Verbo, 2da, sg, presente, imperativo, prioridad 2, "tener"]
=> ["lo", Pron Per encl ato, 3ra, sg, acus, masc, prioridad 1, "'el"]
```

[5] tradicionalmente llamadas locuciones prepositivas, del tipo de cara a, en aras de, etc.

3. Palabra introducida: *sáltelo*

```
s'altelo
=> ["sal", Verbo, 2da, sg, presente, imperativo, prioridad 1, "salir"]
   ["s'alte", Verbo, 2da, sg, presente, imperativo, prioridad 1, "saltar"]
=> ["te", Pron Per encl ato, 2da, sg, acus y dat, masc y fem, prioridad 1, "t'u"]
=> ["lo", Pron Per encl ato, 3ra, sg, acus, masc, prioridad 1, "'el"]
```

Otra posibilidad que se podría haber dado es:

```
s'altelo
=> ["s'alte", Verbo, 2da, sg, presente, imperativo, prioridad 1, "saltar"]
   ["sal", Verbo, 2da, sg, presente, imperativo, prioridad 1, "salir"]
=> ["lo", Pron Per encl ato, 3ra, sg, acus, masc, prioridad 1, "'el"]
```

La situación que se da en la fase operacional se podría expresar como se indica en la figura 2.

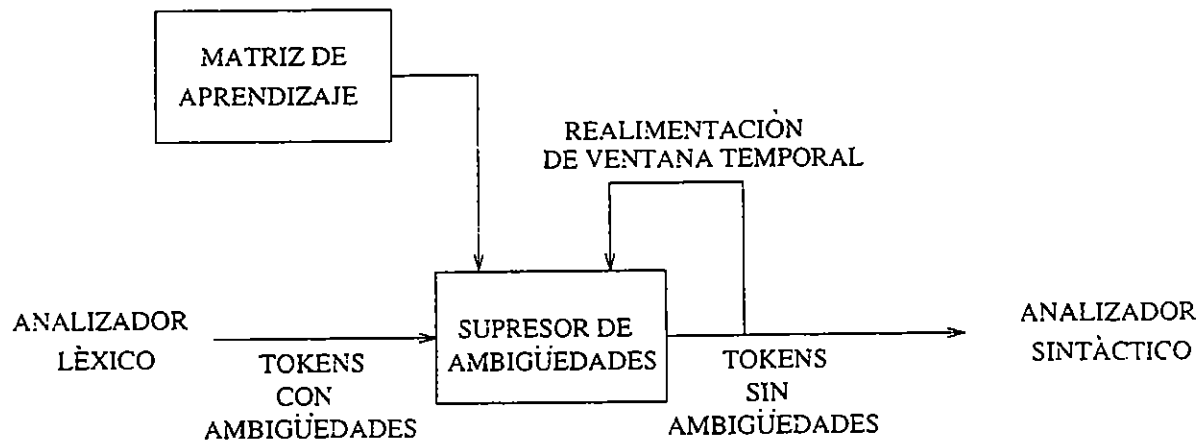

Figura 2: Visión del sistema en la fase operacional.

4.1 Un ejemplo de prueba

En este apartado presentamos una prueba de la fase operacional, que ha sido realizada tomando como texto de aprendizaje un fragmento de la obra Crónica de una muerte anunciada de Gabriel García Márquez [2], constituido por 500 palabras y que fue previamente desambiguado a mano.

La primera tarea fue configurar el supresor estadístico de ambigüedades tomando como información léxica la mostrada en la figura 1. Esta configuración originó cerca de 400 etiquetas léxicas, que fueron las consideradas tanto para el aprendizaje como para la fase operacional. La tarea siguiente fue la llamada al analizador léxico, con la opción de desambiguación y la matriz generada a partir del texto de aprendizaje anterior.

El resultado que se obtuvo, sobre un fragmento de cerca de 550 palabras de la obra citada, fue aproximadamente de un 95% de acierto en la elección de la etiqueta correcta. Si bien este porcentaje resulta alentador, es evidente que con el tamaño del texto de aprendizaje que se ha seleccionado, las estadísticas no son significativas (ver tabla 3). Todo el proceso se ha desarrollado con una base de datos léxica constituida por 4.500 lemas.

Category	Type	Etiquetaciones correctas	Etiquetaciones erróneas
Adjective		29	*Verb:* 1
Adverb	*nuclear*	11	*Verb:* 1
	otros	5	
Article		44	
Conjunction	*coordinate*	12	
	subordinate	23	*Verb:* 1 *Relative:* 1
Demonstrative		3	
Indefinite		23	*Numeral cardinal:* 4 *Verb:* 2
Interjection		0	
Interrogative		0	
Numeral	*cardinal*	7	*Verb:* 1
	otros	0	
Peripheral	*todas*	0	
Preposition		91	*Verb:* 3
Personal Pronoun	*tonic*	3	
	proclitic atonic	19	*Article:* 2
	enclitic atonic	2	
Possessive		5	
Punctuation Mark	*todas*	47	
Relative		7	*Conjunction subordinate:* 6
Substantive hline	*common*	95	*Verb:* 3 *Adjective:* 1
	proper	10	
Verb		66	otros *Verb:* 3

Tabla 3: Estadísticas de la fase operacional.

Las estadísticas de la tabla 3 nos sugieren varias reflexiones:

1. Algunas categorías, como por ejemplo *Demonstrative*, *Possessive* y *Punctuation Mark*, no plantean complicaciones de elección entre varias alternativas.
2. Habrá que centrar los esfuerzos en eliminar los errores referentes a las categorías *Relative* y *Conjunction subordinate*[6]. Tanto en este tipo de error, como en otros que pudieran aparecer, se puede proceder de varias maneras:
   (a) Manejando textos de aprendizaje mayores que, en breve, tendrá disponibles nuestro equipo. Esto es lo que se ha venido haciendo hasta ahora.
   (b) Seleccionando, con ayuda de un equipo de lingüistas, las etiquetas más adecuadas para llevar a cabo la desambiguación, a través de la ventana de configuración presentada en la figura 1.
3. La selección mencionada en el punto anterior debe estar orientada a minimizar la elección de *Verb* como etiqueta, ya que ésta constituye la mayor fuente de errores o selecciones incorrectas. Esto puede lograrse considerando más información de otras categorías en la configuración del desambiguador.
4. La desambiguación mediante reglas, proyectada para el futuro, deberá centrarse fundamentalmente en el estudio de las situaciones en que el desambiguador actual no permite reducir el margen de error.

[6]los errores se deben, fundamentalmente, a la aparición de la partícula *que*.

5 Discusión: Elección de un método estadístico frente a otras posibilidades

En la eliminación de ambigüedades, en nuestro sistema, se han manejado métodos estadísticos. Más adelante, se considerará la posibilidad de utilizar reglas de desambiguación. El método actual de supresión de ambigüedades guiará, de alguna forma, al que se desarrolle posteriormente.

En el pasado se ha dedicado una gran cantidad de esfuerzos al problema de la etiquetación de textos. En líneas generales, ha habido dos aproximaciones [6]:

- basada en reglas (Klein y Simmons 1963; Brodda 1982; Paulussen y Martin 1992; Brill et al. 1990);
- probabilística (Bahl y Mercer 1976; Debili 1977; Stolz, Tannenbaum y Carstensen 1965; Marshall 1983; Leech, Garside y Atwell 1983; Derouault y Merialdo 1986; DeRose 1988; Church 1989; Beale 1988; Marcken 1990; Merialdo 1991; Cutting et al.).

Hoy en día se afirma con frecuencia que la segunda aproximación es preferible. Esta opinión se debe al coste que tiene escribir "a mano" un conjunto de reglas con la cobertura suficiente para obtener resultados que igualen los de los sistemas estadísticos. Éstos, evitando el "trabajo manual" llegan a obtener unos resultados de hasta un 96% de acierto. Sin embargo, el etiquetador de Brill constituye un contraejemplo: a partir de un conjunto de patrones de reglas escritos a mano, este sistema extrae automáticamente las instancias de las reglas y obtiene resultados tan buenos como los de un etiquetador estadístico. Otro ejemplo que contradice la afirmación anterior es el sistema diseñado por Voutilainen [8]. Está basado en el formalismo sintáctico *Constraint Grammar* y obtiene un resultado del 99% para el inglés. Atro Voutilainen, para conseguir este nivel de resultados, diseñó a mano aproximadamente 1000 reglas, lo cual le supuso cerca de un año de trabajo.

Utilizando un supresor de ambigüedades que opere con reglas, podemos encontrarnos con situaciones en que son aplicables varias de ellas. Para solventar este tipo de situaciones, es necesario asignar prioridades a las reglas de forma que la selección pueda hacerse incluso cuando se pueda aplicar más de una regla. La desventaja de esta solución es doble. Por una parte, complica el algoritmo y, por otra, surge el problema de cómo asignar las prioridades a las reglas para lograr una operatividad final.

Hoy en día, la tendencia mayoritaria es tomar como punto de partida los resultados de un desambiguador estadístico e intentar reducir el 4% de error subsistente[7], mediante información lingüística. Esta es la filosofía que se sigue dentro del proyecto en el que se enmarca este trabajo. Por ello, en el futuro se desarrollará un sistema que, valiéndose del conocimiento aportado por el supresor de ambigüedades actual, lo mejore.

Recientemente, se han propuesto trabajos que usan redes de neuronas artificiales (Benello, Mackie y Anderson 1989; Nakamura y Shikano 1989). Todo el proceso que se ha descrito aquí encaja con la forma de operar de las redes neuronales. Sin embargo, aún es pronto para plantearnos el desarrollo de un módulo de esas características mientras no podamos asegurar un buen rendimiento con la utilización de estas técnicas.

[7] con métodos puramente estadísticos, no se ha logrado superar el 96-97% de aciertos.

6 Conclusiones

A cada palabra, fuera de contexto, le puede corresponder, o bien una sola etiqueta, o bien un conjunto limitado de etiquetas. Un análisis léxico de una palabra dada nos proporciona el conjunto de etiquetas que le atañen.

Cuando una palabra ofrece varias alternativas de etiquetación, generalmente se puede escoger la etiqueta correcta usando estadísticas contextuales que definen la secuencia válida de etiquetas. El usuario puede, utilizando el sistema propuesto, seleccionar la información léxica más adecuada para refinar el aprendizaje.

Una aproximación probabilística del problema ofrece las siguientes ventajas:

- Proporciona un marco de trabajo teórico firme: el de la estadística.
- Los resultados de la aproximación son claros.
- Las probabilidades proporcionan un camino sencillo para eliminar las ambigüedades.
- Las probabilidades pueden estimarse automáticamente a partir de los datos.

Referencias

[1] Church, Mercer, *Introduction to the Special Issue on Computational Linguistics Using Large Corpora*, Computational Linguistics 1993, Volumen 19, Número 1, pp 1-24.

[2] G. García Márquez, *Crónica de una muerte anunciada*, Mondadori, Madrid, 1988.

[3] B.W. Kernighan, D.M. Ritchie, *El lenguaje de programación C*, Prentice-Hall Hispanoamericana S.A., 1991.

[4] B.W. Kernighan, R. Pike, *El entorno de programación Unix*, Prentice-Hall Hispanoamericana S.A., 1987.

[5] T. Mason, D. Brown, *Lex & Yacc*, O'Reilly & Associates, Inc.

[6] B. Merialdo, *Tagging English Text with a Probabilistic Model*, Computational Linguistics 1994, Volumen 20, Número 2, pp 155-171.

[7] J.K. Ousterhout, *Tcl and the Tk Toolkit*, Addison-Wesley Profesional Computing Series, 1994.

[8] A. Voutilainen, P. Tapanainen, *Ambiguity resolution in a reductionistic parser*, EACL'93, pp 394-403.

[9] M. Vilares Ferro, M.A. Alonso Pardo, *Towards Analyzers Based on Efficient Logical Frames*, Proc. of the *II* International Conference on Mathematical Linguistics, Tarragona, 1996.

[10] M. Vilares Ferro, J. Graña Gil, A. Pan Bermúdez, *Building Friendly Architectures for Tagging*, Proc. of SEPLN'96, Sevilla, 1996.