Procesamiento del Lenguaje Natural, num. 29 (2002), pp.39-45 recibido 24-05-2002; aceptado 03-07-2002

Resolución de la ambigüedad mediante redes neuronales

Mª Teresa Martín Valdivia
Universidad de Jaén
Av. Madrid 37, 23071
maite@ujaen.es

Manuel García Vega
Universidad de Jaén
Av. Madrid 37, 23071
mgarcia@ujaen.es

L. Alfonso Ureña López
Universidad de Jaén
Av. Madrid 37, 23071
laurena@ujaen.es

Resumen: La resolución de la ambigüedad léxica es una tarea importante dentro del procesamiento del lenguaje natural. En este trabajo, se presenta un nuevo enfoque basado en redes neuronales para aumentar la precisión en la tarea de desambiguación. Concretamente, se utiliza el algoritmo de aprendizaje basado en el modelo de Kohonen conocido con el nombre de algoritmo de aprendizaje por cuantificación vectorial LVQ (Learning Vector Quatization). Los resultados obtenidos demuestran que la utilización de dicho algoritmo competitivo supera notablemente a otros algoritmos ampliamente utilizados en otros trabajos.
Palabras clave: Redes Neuronales Artificiales, algoritmo LVQ, WSD, SemCor

Abstract: Word Sense Disambiguation is an important task in Natural Language Processing. This work shows a new approach based on neuronal networks that increases the precision in WSD task. The learning algorithm is based on the Kohonen model, known with the name of the LVQ algorithm (Learning Vector Quatization). The results demonstrate that the use of this competitive algorithm performes better than other algorithms widely used in other works.
Keywords: Artificial Neural Networks, LVQ algorithm, WSD, SemCor

1 Introducción

El objetivo de este trabajo es la resolución de la ambigüedad léxica la cual aparece cuando las palabras presentan diferentes significados. La tarea de desambiguación (Word Sense Disambiguation, WSD) consiste en identificar el sentido de una palabra en un determinado contexto dentro de un conjunto de candidatos determinado. La desambiguación no es un fin en sí misma, sino una tarea intermedia muy necesaria para algunas tareas del Procesamiento del Lenguaje Natural (PLN) como Recuperación de Información (IR), Traducción Automática (MT), Categorización de Textos (TC), Extracción de Información (IE)...

En los últimos años se han propuesto varios enfoques de distinta naturaleza para WSD, que podemos clasificarlos de acuerdo con la fuente de conocimiento utilizada. Algunos enfoques se basan en la utilización de algún tipo de base de datos léxica (Xiaobin y Spakowicz, 1995). Otros, hacen uso de corpus de texto no-anotados (Pedersen y Bruce, 1997) o de corpus anotados semánticamente como colección de entrenamiento (Bruce y Janyce, 1994). Finalmente, recientes trabajos proponen la combinación de varias fuentes de conocimiento (Wilks y Stevenson, 1998; Ureña, Buenaga, Gómez, 2001), como bases de datos léxicas, corpus de texto, algunas heurísticas, colocaciones...

En esta comunicación se presenta un nuevo enfoque para resolver la ambigüedad léxica que hace uso de redes neuronales artificiales. Una Red Neuronal Artificial (RNA) es un modelo de procesamiento de información que está inspirado en un sistema nervioso biológico (McClelland y Rumelhart, 1986). Este se

ISSN 1135-5948

compone de un gran número de elementos de procesamiento interconectados (neuronas) trabajando en armonía para resolver problemas específicos. Las RNA aprenden con ejemplos de manera que la configuración de la red para una aplicación específica se realiza a través de un proceso de aprendizaje. Las RNA se han aplicado a un gran número de problemas reales de complejidad considerable. Su ventaja más importante está en resolver problemas que son demasiado complejos para tecnologías convencionales, problemas que no tienen un algoritmo de solución o que su algoritmo de solución es muy difícil de encontrar. Estos problemas incluyen reconocimiento de patrones y pronósticos, clasificación de datos, optimización... Sin embargo, en el ámbito del procesamiento del lenguaje natural aún no han sido suficientemente explotadas.

Precisamente en este trabajo, se propone la utilización de una RNA para aumentar la precisión en problemas de desambiguación del sentido de las palabras. Concretamente, se trata de una red basada en el modelo de Kohonen (Kohonen, 1995). El modelo de Kohonen es una red competitiva. Las redes competitivas se basan en la idea de utilizar reglas de aprendizaje en las que las distintas unidades de procesamiento (neuronas) deben competir por la oportunidad de aprender (entrenarse). Esto se refiere a que la unidad que produce la mayor salida se considera como la "ganadora" y tienen la capacidad de inhibir a las otras unidades, de manera que únicamente los pesos de la unidad ganadora serán ajustados (modificados).

El modelo de Kohonen presenta dos variantes: Mapa auto-organizativo (*Self-Organizing Map*) o SOM y Aprendizaje por Cuantificación Vectorial (*Learning Vector Quantization*) o LVQ. Aunque ambos utilizan un aprendizaje competitivo, la diferencia radica en que el modelo SOM utiliza un método de aprendizaje no supervisado, mientras que el algoritmo LVQ es supervisado.

Como recurso lingüístico se ha usado el corpus SemCor. Puesto que este corpus está etiquetado con los sentidos de WordNet, lo que posibilita el uso de aprendizaje supervisado, se utilizará el algoritmo LVQ para entrenar el contexto de cada uno de los sentidos de una palabra.

Con este trabajo, se pretende presentar una primera aproximación para comprobar si el aprendizaje competitivo y reforzado es viable en tareas de WSD. Como se vio en (García, Martín y Ureña, 2001), este tipo de aprendizaje supervisado funciona satisfactoriamente en otras tareas del PLN.

El resto del artículo se organiza de la siguiente manera. En la sección 2 se presenta el algoritmo de aprendizaje LVQ. A continuación, se describe el desambiguador construido y su entrenamiento. En la sección 4 se detallan los experimentos realizados y los resultados obtenidos. Por último, la sección 5 expone las conclusiones y los trabajos futuros.

2 La red neuronal LVQ

La red basada en el algoritmo de aprendizaje LVQ ha sido utilizada con éxito en muchas aplicaciones, fundamentalmente en análisis de imágenes, reconocimiento de patrones, problemas clasificación... En (Kohonen, 1995) se presenta una larga lista de dichas aplicaciones incluyendo algunas relacionadas con el procesamiento del lenguaje natural, como por ejemplo, el reconocimiento y análisis del habla. Recientemente, el equipo de Kohonen ha desarrollado el sistema conocido como WEBSOM que permite la recuperación de documentos mediante mapas auto-organizativos (Kohonen et al., 2000). En (García, Martín y Ureña, 2001) se muestra cómo la utilización del modelo LVQ en tareas de categorización de texto aumenta considerablemente la precisión obtenida con otros algoritmos de aprendizaje usualmente utilizados en categorización de texto (Lewis et al., 1996).

El algoritmo LVQ está especialmente diseñado para resolver problemas de clasificación de patrones. Se trata de clasificar un conjunto de patrones de entrada en un número finito de clases de manera que cada clase está representada o caracterizada por un vector prototipo. Para ello, el modelo LVQ utiliza un aprendizaje supervisado que permite organizar las unidades de entrenamiento en el espacio de salida mediante regiones que actuarán como clasificadores de los datos de entrada. Se trata de un aprendizaje competitivo que permite reforzar positivamente (premiando) o negativamente (castigando) los pesos de las conexiones dependiendo de que la clasificación haya sido realizada correcta o incorrectamente.

La entrada al sistema son vectores n-dimensionales (x_i) y la salida es una representación del espacio de entradas conocida como vectores prototipo (w_k). Para cada clase,

k, se tiene al menos un vector de pesos w_k. En cada iteración, se selecciona un vector de entrada x_i y se compara con todos los vectores prototipo w_k utilizando como medida de similitud, la distancia euclídea $\|x_i\text{-}w_k\|$. El vector prototipo más cercano a la entrada x_i se proclamará ganador y se ajustará para adaptarse de la mejor manera posible a la entrada. Así, para encontrar el vector prototipo ganador w_c habrá que seleccionar aquel que cumpla

$$\|x_i - w_c\| = \min_k \{\|x_i - w_k\|\} \quad (1)$$

La clase c, con la que está asociado el vector w_c, será la ganadora de la competición, y por tanto, se podrá comprobar si el procedimiento ha clasificado correctamente sin más que comparar c con la clase a la que pertenece el vector de entrada x_i. Suponiendo que d es la clase a la que pertenece x_i, la regla de aprendizaje utilizada por el algoritmo LVQ es la siguiente

$$w_c = \begin{cases} w_c + \alpha(t)\cdot(x_i - w_c) \text{ si } c = d \\ w_c - \alpha(t)\cdot(x_i - w_c) \text{ si } c \neq d \end{cases} \quad (2)$$

Sólo se ajustan los pesos de la unidad ganadora, permaneciendo sin modificación el resto de vectores prototipo. Si la clase ganadora c de la competición es correcta, es decir, coincide con la clase del vector de entrada d, los pesos de w_c se modifican acercándose al vector de entrada (se premia el acierto en la clasificación). Si por el contrario, la clasificación es incorrecta, se modifican alejándolo del vector de entrada (se castiga el error en la clasificación).

$\alpha(t)$ es el ratio de aprendizaje, siendo $0<\alpha(t)<1$, una función monótona decreciente del tiempo. La inicialización de $\alpha(t)$ se hace con un valor cercano a cero, y decrece linealmente de manera que al final del proceso de aprendizaje su valor es prácticamente nulo (Kohonen, 1995).

3 WSD

A continuación, se va a presentar un desambiguador basado en el Modelo de Espacio Vectorial (MEV). Cada sentido de una palabra es representado con un vector, así como el contexto de la palabra a desambiguar.

Los pesos para los distintos términos se calculan de manera análoga a (Salton, 1983) con el estándar *tf•idf*. Estos vectores serán la entrada del algoritmo de entrenamiento LVQ, para obtener finalmente los vectores prototipo con los pesos definitivos de cada sentido y para todas las palabras.

El vector que representa el contexto de cada palabra a desambiguar deberá compararse con cada uno de los vectores de sus sentidos, según la similitud del coseno:

$$sim(w_k, x_i) = \frac{w_k \cdot x_i}{|w_k| \cdot |x_i|} \quad (3)$$

El sentido representado por el vector de mayor similitud será el designado como sentido desambiguado (sentido ganador).

3.1 Recursos lingüísticos

Para nuestros experimentos se ha usado SemCor 1.6, tanto para el entrenamiento, como para la evaluación. Se trata del Brown Corpus etiquetado con los sentidos de WordNet 1.6. Concretamente, la partición *Brown-2* ha sido utilizada como entrenamiento, mientras que con la partición *Brown-1* se han realizado los experimentos de evaluación.

Para aplicar el MEV es necesario definir el tamaño del contexto. En las pruebas realizadas se va a usar el párrafo como unidad semántica con el fin de aumentar la precisión en el proceso de desambiguación (Ureña et al., 1998) realizando los experimentos únicamente sobre los sustantivos de todo el corpus.

```
<contextfile concordance=brown>
<context filename=br-a01 paras=yes>
<p pnum=1>
<s snum=1>
<wf cmd=ignore pos=DT>The</wf>
<wf cmd=done rdf=group pos=NNP lemma=group
wnsn=1 lexsn=1:03:00::
pn=group>Fulton_County_Grand_Jury</wf>
<wf cmd=done pos=VB lemma=say wnsn=1
lexsn=2:32:00::>said</wf>
<wf cmd=done pos=NN lemma=friday wnsn=1
lexsn=1:28:00::>Friday</wf>
<wf cmd=ignore pos=DT>an</wf>
.
.
.
<wf cmd=done pos=NN lemma=irregularity wnsn=1
lexsn=1:04:00::>irregularities</wf>
<wf cmd=done pos=VB lemma=take_place wnsn=1
lexsn=2:30:00::>took_place</wf>
<punc>.</punc>
</s>
</p>
```

Figura 1: Fichero br-a14 de SemCor

Para cada nombre, se debe detallar su contexto con todas las palabras que aparecen en el párrafo donde figura, destacando si se trata de nombres, verbos, adjetivos o adverbios e

incluyendo el sentido de cada una de ellas. Toda esta información aparece expresamente en SemCor, como se ve en la Figura 1.

Llamaremos dominio a todos los contextos referidos a un mismo término. Así pues, cualquier palabra dada aparecerá, seguramente, en más de una ocasión, dando lugar a un dominio con tantos contextos como ocurrencias tenga en el corpus, incluyendo la mayoría de los sentidos de la palabra en cuestión.

Cada uno de los dominios generados debe ser actualizado, calculando los pesos de las palabras que lo componen según el MEV.

Una vez calculados los pesos, se dispone de un conjunto de vectores (vectores x_i) que forman el espacio de entrada de nuestra red neuronal (Figura 2). Como se aprecia, la palabra *access* aparece con dos sentidos. El primer sentido tiene una sola ocurrencia, por lo que le corresponde un vector, mientras que el segundo sentido aparece dos veces, figurando dos vectores. Cada vector tiene una secuencia de pares (término, peso) de longitud variable, determinada por el número de palabras que tiene el párrafo donde aparece. Además, cada término tiene tres componentes, la palabra, el *pos* y el sentido.

access#1 13 access\1#1 0.258199 all-weather\5#1 0.258199 feeding\1#2 0.258199 floor\1#1 0.258199 heavy\3#2 0.258199 next\5#1 0.258199 possible\3#1 0.258199 problem\1#1 0.258199 provide\2#1 0.258199 road\1#1 0.258199 snowfall\1#1 0.258199 so\4#2 0.258199 year\1#1 0.258199
access#2 16 access\1#2 0.099288 administrative\3#1 0.269023 appeal_board\1#1 0.269023 be\2#3 0.269023 deny\2#3 0.269023 due_process\1#1 0.269023 file\1#1 0.269023 … petitioner\1#1 0.269023 proceedings \1#1 0.269023 rebut\2#1 0.269023 statement\1#1 0.269023
access#2 73 2\5#1 0.072280 access\1#2 0.026676 allotment\1#1 0.289121 allotment\1#2 0.072280 alone\4#1 0.072280 already \4#1 0.072280 also\4#1 0.072280 amended\3#1 0.072280 amount\1#1 0.072280 assist\2#1 0.072280 avoid\2#2 0.072280 base\1#6 0.072280 base\2#1 0.072280 basic\3#1 0.072280 basis \1#1 0.072280 call\2#1 0.072280 capacity\1#4 0.072280 ceiling\1#3 0.144560 design\2#1 0.144560 use\2#1 0.072280 vocationally\4#1 0.072280 wealth\1#1 0.144560

Figura 2: Dominio de entrada

3.2 Entrenamiento

Aunque estrictamente los vectores iniciales que necesita el algoritmo LVQ deben de ser aleatorios, en el caso que nos ocupa (la desambiguación), el número de vectores de los dominios (el conjunto de vectores de entrenamiento) puede ser anormalmente pequeño. En este caso, al terminar el entrenamiento, algunos de los pesos de los vectores prototipo (los vectores de pesos w_k) no tendrán ajustados sus valores, con lo que en realidad se está introduciendo ruido.

En su lugar, se van a inicializar a cero todos los pesos, dejando actuar a la red neuronal de forma que en el entrenamiento de cada vector, introduzca los pesos adecuados dependiendo de si la elección del ganador ha sido o no correcta.

Como contrapartida a esta decisión, se obtendrán vectores con algunos pesos negativos representando a las palabras que se hallaban en vectores desambiguados de manera incorrecta durante el entrenamiento. Esto no supone ningún problema, puesto que el cálculo de la similitud con el coseno sigue siendo correcto.

Con el *Brown-2* se generan 29.941 vectores de entrenamiento que se corresponden con 6.819 dominios, de los que 1.483 se corresponden con palabras polisémicas, que inicialmente tienen asignados los pesos que se obtienen de aplicar directamente el MEV. Así pues, cada uno de los vectores de entrada será introducido en la red neuronal sucesivas veces. Conforme se va iterando, los pesos de los vectores prototipo se modifican en menor medida en función del ratio de aprendizaje $\alpha(t)$, comenzando con un valor de 0,3 y decrementándose linealmente según la siguiente ecuación

$$\alpha(t+1) = \alpha(t) - \frac{\alpha(0)}{P} \qquad (4)$$

donde P es el número total de pasadas del entrenamiento para un dominio determinado. Según (Kohonen et al., 1996), el número de pasadas P suele ser suficiente con 40 veces el número de vectores prototipos. Por lo tanto, para cada dominio P se fijará a 40xNS_i siendo NS_i el número de sentidos para el dominio i.

Se trata de encontrar el sentido ganador calculando la menor distancia euclídea entre el vector de entrada y los vectores prototipo de la red LVQ. Una vez elegido el ganador, sus pesos serán ajustados utilizando la ecuación 2.

Una vez finalizado el proceso de aprendizaje, los vectores prototipo contienen los pesos óptimos para cada uno de los sentidos de cada término (Figura 3). En esta ocasión sólo aparece un vector por sentido, siguiendo el mismo esquema que el explicado anteriormente en la Figura 2.

Con el fin de comparar los resultados obtenidos con el modelo LVQ, se han incluido experimentos con otros dos algoritmos de aprendizaje sobre el mismo conjunto de datos. Se trata del algoritmo de Rocchio (Rocchio, 1971) y el algoritmo de Widrow-Hoff (Widrow y Sterm, 1985) que han sido ampliamente utilizados en otros trabajos (Lewis et al., 1996). Para cada uno de los algoritmos, los parámetros se han ajustado según el estudio realizado en (Ureña, 2000).

access#1 31 access\1#1 0.258197 access\1#2 -0.000382 appeal_board\1#1 -0.001035 area\1#1 0.258197 be\2#1 0.258197 deny\2#3 -0.001035 … road\1#1 0.258197 snowfall\1#1 0.258197 so\4#2 0.258197 statement\1#1 -0.001035 year\1#1 0.258197
access#2 86 access\1#2 0.087952 administrative\3#1 0.186831 allotment\1#1 0.205910 allotment\1#2 0.051477 051477 alone\4#1 0.051477 already\4#1 0.051477 also\4#1 0.051477 … stipulate\2#1 0.051477such\5#1 0.051477 support\1#1 0.051477 thickly\4#2 0.051477 use\2#1 0.051477 0.051477 wealth\1#1 0.102955

Figura 3: Dominio generado por el algoritmo LVQ

4 Evaluación

Para la evaluación de nuestro sistema se han seleccionado los contextos completos de todos los sustantivos que aparecen en la partición *Brown-1*.

El número promedio de vectores por dominio es de 3,37, y la media de sentidos es de 2,73, lo que justifica la inicialización a cero de los vectores prototipo de la red LVQ.

El problema fundamental que se plantea al evaluar es que el entrenamiento de la red neuronal LVQ introduce características en los vectores del dominio que no son compatibles con la métrica escogida para los pesos de las palabras.

Cada uno de los vectores prototipos incluyen todas las palabras de todos los contextos de un determinado dominio.

Las palabras del vector de entrada provocan la actualización de los pesos del vector ganador y nada impide que dicho vector ganador sea distinto en cada iteración, haciendo que la frecuencia inversa sea muy alta para la mayoría de los términos de los dominios.

Esto nos lleva a un empobrecimiento de los pesos de las palabras, hasta el punto de que, precisamente, aquellas palabras que realmente marcan la diferencia con pesos muy positivos en los vectores adecuados y muy negativos en los que se desambiguaron incorrectamente, aparecen en la totalidad de los vectores de los dominios, por lo que se tornan irrelevantes en el cálculo de los pesos de los vectores de evaluación.

Es necesario, pues, cambiar el cálculo de los pesos. En nuestros experimentos, se ha adoptado la frecuencia normalizada para el cálculo de los pesos de las palabras de los vectores de evaluación. Para resolver los ceros que se obtienen en aquellas palabras que figuran en todos los contextos se suprime la frecuencia inversa del cálculo de los pesos. De esta forma, se mantiene la importancia de cada término con respecto a cada sentido y, al estar normalizada, de manera uniforme.

Sólo falta efectuar el cálculo de la similitud y resolver el sentido de mayor similitud.

En la Tabla 1 se muestran los resultados de la evaluación, con 4.997 sustantivos polisémicos, en medidas de precisión. En la evaluación se han omitido aquellas palabras del *Brown-1* que no aparecen en el entrenamiento (*Brown-2*) obteniendo así una cobertura del 100%. Observamos que el uso de la red neuronal LVQ mejora notablemente la precisión del resto de los algoritmos. Incluimos además una medida del promedio de las precisiones el número de sentidos de las palabras (precisión macroaveraging por sentidos).

	Precisión	MacroAv
Rocchio	55,81%	58,02%
Widrow-Hoff	58,87%	57,93%
LVQ	71,75%	71,63%

Tabla 1: Precisión y Macroaveraging para cada algoritmo

La precisión para el algoritmo LVQ ha sido de 71,75% lo que supone una mejora de casi un 16% sobre el algoritmo de Rocchio y de casi un 13% sobre el algoritmo de Widrow-Hoff.

Por otro lado, el promedio de las precisiones (precisión macroaveraging) con respecto al número de sentidos de las palabras de evaluación ha sido del 71,63% lo que supone una mejora del casi 14% con respecto a los algoritmos de Rocchio y Widrow-Hoff.

En esta evaluación no se han tenido en cuenta aquellos términos que no se podían

desambiguar. Debido a la partición que se ha hecho de SemCor, hay palabras que no están en el entrenamiento pero sí en la evaluación, y viceversa. Además, también hay sentidos de palabras que están ausentes en el entrenamiento. En ambos casos, no se puede desambiguar. Como el objetivo consiste en conocer la bondad del algoritmo LVQ en medidas de precisión, simplemente se han eliminado estos casos cuya desambiguación es evidentemente errónea siempre y no aportan datos reales.

5 Conclusiones y trabajos futuros

En este trabajo, se presenta un nuevo enfoque basado en aprendizaje neuronal para resolver la ambigüedad léxica. Como muestran los resultados obtenidos en los experimentos realizados, la utilización del algoritmo LVQ mejora notablemente la precisión obtenida en la tarea de WSD.

Puesto que los resultados obtenidos en este trabajo son prometedores, se considera prioritario realizar una comparación de la red LVQ con RNA utilizadas en otros trabajos (Towell y Voorhees, 1998).

Consideramos como principal línea de investigación para otros trabajos, la introducción de técnicas estadísticas como la semántica latente que permitirá destacar los términos importantes de cada dominio con lo que es de esperar que nuestro sistema mejore.

Otro aspecto a tener en cuenta en trabajos futuros consiste en el uso de lexicones para enriquecer el espacio de entrada a la red neuronal LVQ o su uso combinado con otros recursos lingüísticos.

Por último, los buenos resultados obtenidos con el algoritmo LVQ, anima a utilizar esta red en otras tareas de PLN, así como la aplicación de otras redes neuronales a la tarea concreta de WSD.

Bibliografía

Bruce, R. y W. Janyce. 1994. Word sense disambiguation using decomposable models. En *Proceedings of 33rd Annual Meeting of the Association for Computational Linguistics* (ACL'94).

García, M., Martín, M.T., Ureña, L.A. 2001. Categorización de Textos Multilingües Basada en Redes Neuronales. *Procesamiento del Lenguaje Natural*, Revista nº 27, septiembre 2001.

Kohonen, T. 1995. *Self-organization and associative memory.* 2ª Edición, Springer.Verlag, Berlín.

Kohonen, T., Hynninen, J., Kangas, J., Laaksonen, J., Torkkola, K. 1996. Informe técnico, LVQ_PAK: The Learning Vector Quantization Program Package. Helsinki University of Technology, Laboratory of Computer and Information Science, FIN-02150 Espoo, Finlandia.

Kohonen, T., Kaski, S., Lagus, K., Salojärvi, J., Honkela, J., Paatero, V., Saarela, A. 2000. Self oganizing of a massive document collection. IEEE Transaction on neural networks. Vol. 11, nº 3, Mayo 2000.

Lewis, D. D., R. E. Schapire, J. P. Callan y R. Papka. 1996. Training algorithms for linear text classifiers. En *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.*

McClelland, J. y D. Rumelhart. 1986. *Parallel Distributed Processing.* Volúmenes I y II. MIT Press, Cambridge.

Pedersen, P. y R Bruce. 1997. Distinguishing word senses in untagged text. En *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*.

Rocchio J.J. 1971. Relevance feedback in information retrieval. En *G. Salton, editor, The SMART Retrieval System: Experiments in Automatic Document Processing*. Prentice-Hall.

Salton, G. y M.J. McGill. 1983. *Introduction to Modern Information Retrieval.* McGraw-Hill, New York.

Towell, G. y E.M. Voorhees. 1998. Disambiguating highly ambiguous words. Computational Linguistic. vol. 24, nº1. pp. 125-145

Ureña L.A., Buenaga M. y Gómez J.M. 2001. *Integrating Linguistic Resources in TC through WSD*, Computers and the Humanities, vol. 35, nº 2.

Ureña, L.A. 2000. Resolución de la Ambigüedad Léxica en Tareas de Clasificación Automática de Documentos.

Tesis Doctoral, Departamento de Lenguajes y Sistemas Informáticos. Universidad de Granada.

Ureña, L.A., M. García, J.M. Gómez y A. Díaz. 1998. Integrando una base de datos léxica y una colección de entrenamiento para la desambiguación del sentido de las palabras. *Procesamiento del Lenguaje Natural*, nº 23.

Widrow, B. y S. Sterns. 1985. *Adaptative Signal Processing*. Prentice-Hall

Xiaobin, L. y S. Spakowicz. 1995. WORDNET-based algorithm for word sense disambiguation. En *Proceedings of the Fourteenth International Joint conference on Artificial*