TextRank como motor de aprendizaje en tareas de etiquetado *

Fermín Cruz, José A. Troyano, Fernando Enríquez y **F. Javier Ortega**
Dep. de Lenguajes y Sistemas Informáticos
Universidad de Sevilla
Avda. Reina Mercedes s/n
41012 Sevilla
troyano@lsi.us.es

Resumen: Este trabajo trata de cómo adaptar el método TextRank para que funcione de manera supervisada. TextRank es un método basado en grafos que aplica las ideas de PageRang al PLN. Nuestra principal aportación es la definición de un método que permite crear un grafo que incluye información extraída de un corpus de entrenamiento. Hemos comparado los resultados de nuestro método en distintas tareas PLN con los obtenidos por herramientas especializadas en el etiquetado. El rendimiento de nuestro sistema se acerca bastante al de estas herramientas llegando incluso a superar a algunas de ellas en varias tareas.
Palabras clave: Ordenación de Grafos, Corpus anotado, Aprendizaje Supervisado

Abstract: In this paper we investigate how to adapt the TextRank method to make it work in a supervised way. TextRank is a graph based method that applies the ideas of PageRank to NLP tasks. Our main contribution is the definition of a method that allows to apply TextRank to a graph that includes information generated from a training tagged corpus. We have compared the results that our method achieves in several NLP tasks with those obtained with specialized tagging tools. The performance of our system is quite near to these tools, improving the results of some of them in several tasks.
Keywords: Graph Ranking, Tagged corpus, Supervised Learning

1. Introducción

En muchas aplicaciones relacionadas con el Procesamiento del Lenguaje Natural los grafos se revelan como la representación más adecuada. De hecho, desde el momento en el que un texto es fragmentado en palabras y se establece algún tipo de relación entre dichas palabras, disponemos de una representación en forma de grafo. Sin embargo, esta conexión entre PLN y grafos no siempre está presente en los modelos que se utilizan para resolver muchos de los problemas relacionados con el tratamiento de textos. Así, en las visiones generativas (basadas en gramáticas) del PLN el modelo de representación dominante suele ser el árbol, como consecuencia directa del concepto de árbol de derivación.

En las propuestas estadísticas (basadas en corpus) hay más variedad de modelos, pero no son muchos los que explotan la conexión entre grafo y lenguaje. En este sentido, técnicas como los árboles de decisión, el modelado de máxima entropía, el aprendizaje basado en ejemplos o el aprendizaje basado en transformaciones descansan sobre representaciones bastante alejadas de los grafos. Por otro lado, técnicas como los modelos de Markov y las redes neuronales sí se basan en representaciones en forma de grafo, aunque en estos modelos no se utilizan directamente algoritmos relacionados con la teoría de grafos.

Recientemente están apareciendo propuestas que dan más protagonismo a los grafos en el proceso de entrenamiento, e incluso empiezan a surgir *workshops* (como (Radev, 2006)) que incluyen como tema principal la aplicación de algoritmos basados en grafos al PLN.

Una propuesta interesante es la de TextRank (Mihalcea, 2004), un algoritmo basado en la misma idea que originalmente usó Google para establecer un *ranking* de páginas (Brin, 1998). El algoritmo TextRank ha sido aplicado con bastante éxito a diver-

* Parcialmente financiado por el Ministerio de Ciencia y Tecnología (TIN2004-07246-C03-03).

sas tareas del PLN como son la extracción de resúmenes, la extracción de palabras clave o la desambiguación de significados. A pesar de ser un algoritmo no supervisado alcanza en dichas tareas resultados similares a los obtenidos en la literatura por sistemas supervisados.

El objetivo principal de este trabajo es investigar cómo se puede utilizar el algoritmo TextRank de una manera supervisada. Para ello hay que introducir información recopilada desde un corpus de entrenamiento (que al ser el aprendizaje supervisado estaría etiquetado) en el grafo que represente un determinado problema que posteriormente sería procesado por el algoritmo TextRank. La intuición nos dice que dada la flexibilidad que ha mostrado dicho algoritmo en su versión no supervisada, el hecho de introducir la información proveniente del corpus anotado no tendría que estropear la bondad del algoritmo. La clave está en encontrar una representación para cada problema que saque el máximo rendimiento a la potencia del algoritmo TextRank.

La organización del resto del artículo es como sigue, en la siguiente sección se explicará el algoritmo TextRank, en la sección tercera mostraremos cómo hemos construido el grafo a partir de un corpus etiquetado, la sección cuarta incluirá el diseño experimental, la quinta los resultados, y por último la sección sexta estará dedicada a las conclusiones y líneas de trabajo futuro.

2. El algoritmo TextRank

La idea principal de TextRank es aplicar un algoritmo de ordenación basado en grafos a problemas relacionados con el PLN. En concreto, hace uso del famoso algoritmo PageRank (Brin, 1998), una de las claves que llevaron a Google a la posición de privilegio de la que actualmente disfruta en Internet. PageRank es utilizado para medir la importancia de cualquier página web en Internet en función de los enlaces que dicha página recibe, aunque también se han utilizado ideas similares en otros contextos como el análisis de redes sociales o de redes de referencias bibliográficas.

La formalización del algoritmo PageRank es bastante simple, dado un grafo $G = (V, E)$ donde V es un conjunto de vértices y E un conjunto de arcos dirigidos entre dos vértices, se definen en primer lugar dos operaciones $E(V_i)$ y $S(V_i)$ que calculan, respectivamente, el número de arcos que entran o salen del vértice V_i. A partir de estas dos operaciones básicas, se define la puntuación (o PageRank) de un determinado vértice con la siguiente fórmula:

$$P(V_i) = (1 - d) + \ d \sum_{j \in E(V_i)} \frac{1}{|S(V_j)|} \ P(V_j)$$

donde d es un factor de amortiguación que tiene como objetivo incluir en el modelo la probabilidad de que haya un salto aleatorio de un vértice del grafo a cualquier otro. En el contexto de la navegación en Internet, dicho factor representa la probabilidad de que un usuario acceda a una página a través de un enlace situado en la página actual, siendo por tanto $(1 - d)$ la probabilidad de que dicho usuario salte a una página aleatoria no enlazada con la página actual. En la definición original de PageRank se recomienda un valor de 0.85 para el factor d.

Partiendo de valores arbitrarios para las puntuaciones de los nodos de un grafo, se alcanza un punto de convergencia aplicando iterativamente la fórmula hasta que la mayor diferencia de las puntuaciones obtenidas para cada nodo, entre dos iteraciones, es menor que un determinado umbral. Una vez finalizado el algoritmo, la puntuación alcanzada por cada nodo representa la importancia del mismo, y puede ser utilizada como criterio para la toma de decisiones.

Esta fórmula se puede generalizar con facilidad para ser aplicada a grafos cuyos arcos tengan pesos. En este caso la puntuación de cada nodo se calcularía de la siguiente forma, donde p_{ji} es el peso del arco que va del vértice V_j al V_i:

$$P(V_i) = (1 - d) + \ d \sum_{j \in E(V_i)} \frac{p_{ji}}{\sum_{k \in S(V_j)} p_{jk}} \ P(V_j)$$

Para poder aplicar TextRank sólo es necesario obtener un grafo a partir de un texto, calcular a partir del grafo la puntuación de PageRank y utilizar dicha puntuación de los nodos para resolver cuestiones sobre las unidades textuales a las que se refieren dichos nodos. Este algoritmo se ha aplicado a tareas como la extracción de palabras clave y la generación de resúmenes (Mihalcea, 2004) o desambiguación de significados (Mihalcea,

2004b) con muy buenos resultados. En cada caso la forma de construcción del grafo es distinta. Por ejemplo, en la extracción de palabras clave los vértices son palabras y se establecen arcos entre vértices si hay coocurrencia entre las palabras que representan. Se entiende que hay coocurrencia si están juntas o a una distancia menor que un límite N establecido.

3. Obtención del grafo a partir de un corpus anotado

Las aplicaciones desarrolladas alrededor de TextRank han sido siempre no supervisadas. Es decir, construyen el grafo directamente a partir del texto de evaluación sin hacer uso de ningún corpus de entrenamiento anotado. A pesar de esto, TextRank consigue en las tres tareas antes mencionadas resultados similares a los obtenidos por sistemas que sí usan corpus anotados, y que por tanto realizan un aprendizaje supervisado. Quizá la razón de tan sorprendente hecho (se alcanzan los mismos resultados con menos información) haya que buscarla en la naturaleza de los tres problemas a los que se ha aplicado, que se ajustan muy bien al modelo de grafo.

Parece por tanto que la cercanía de la tarea al modelo de grafo es una condición indispensable para que TextRank se pueda utilizar como motor de aprendizaje. En este punto la pregunta es, ¿será posible aplicar este algoritmo a otras tareas? Nosotros no hemos encontrado aproximaciones a otras tareas clásicas en el PLN como el etiquetado POS, el análisis sintáctico o la extracción de información. Precisamente el objetivo inicial de este trabajo era explorar otras vías de aplicación de este algoritmo al mismo tiempo que intentar encajar su utilización en un marco supervisado que aproveche la información disponible en un corpus de entrenamiento anotado.

No tiene porqué haber una única manera de representar una tarea PLN en forma de grafo para poder aplicar un algoritmo de *ranking* como TextRank. Nosotros hemos elegido una lo suficientemente general como para que pueda ser utilizada en cualquier tarea en la que se asocien etiquetas a palabras. Los vértices de nuestros grafos estarán compuestos por dos informaciones $V = (w, t)$, w es una palabra y t una etiqueta. Si una palabra es ambigua (puede tener varias etiquetas) se crearán tantos vértices como posibles etiquetas pueda tener. La idea principal de nuestra aproximación es construir un grafo con este tipo de vértices para cada frase, aplicar TextRank a dicho grafo y asociar a cada palabra de la frase la etiqueta correspondiente a su vértice mejor puntuado. Si una palabra aparece más de una vez en una frase, se crean vértices diferenciados para que estas múltiples instancias no interfieran entre sí.

Para los arcos del grafo hemos optado por la coocurrencia, de manera que habrá un arco desde un vértice $V_i = (w_i, t_i)$ a otro vértice $V_j = (w_j, t_j)$ si la palabra w_j aparece en la frase después de la palabra w_i. El grafo será por tanto dirigido.

Por último, la información del corpus de entrenamiento aparecerá en el grafo mediante pesos en los arcos. Hemos experimentado con distintas medidas y la que mejor resultado nos ha dado ha sido una combinación de las probabilidades de emisión $P(w|t)$ y transición $P(t|t')$ utilizadas en los modelos ocultos de Markov basados en bigramas. Dichas probabilidades son estimadas a partir de la frecuencia de aparición de etiquetas y palabras en el corpus de entrenamiento:

$$P(w|t) = \frac{C(w,t)}{C(t)} \qquad P(t|t') = \frac{C(t',t)}{C(t')}$$

donde $C(t)$ es el número de veces que aparece la etiqueta t en el corpus de entrenamiento, $C(w,t)$ el número de veces que la palabra w aparece etiquetada con t y $C(t',t)$ el número de ocasiones en las que la etiqueta t' aparece antes de la etiqueta t.

En el modelo de Markov estas probabilidades se utilizan para calcular el mejor etiquetado para una frase maximizando la siguiente probabilidad:

$$P(t_{1,n}|w_{1,n}) = \prod_{i=1}^{n} P(w_i|t_i)P(t_i|t_{i-1})$$

Nosotros utilizaremos la probabilidad $P(w_i|t_i)P(t_i|t_{i-1})$ como peso del arco que va del vértice $V_{i-1} = (w_{i-1}, t_{i-1})$ al vértice $V_i = (w_i, t_i)$ del grafo, y dejaremos a TextRank que calcule la importancia de cada vértice. A diferencia del modelo de Markov en el que se consideran los distintos caminos como competidores y se busca el que maximice la anterior expresión, la aproximación

de TextRank es más colaborativa ya que se combinan las probabilidades desde distintos arcos para determinar el ranking de un determinado vértice.

3.1. Extensiones al modelo inicial

Muchas de las extensiones clásicas de los modelos de Markov pueden ser incluidas en nuestro grafo de una u otra forma. Por ejemplo, podemos redefinir la expresión con la que se calcula el peso de un arco para que incluya en la medida la probabilidad de trigramas y unigramas, además de bigramas, o podemos generar nuevos arcos adicionales a los explicados en nuestro diseño base.

A partir del modelo básico anteriormente expuesto, hemos realizado una serie de pruebas para encontrar variantes con mejor comportamiento. El camino que ha seguido nuestra investigación en este sentido ha sido el siguiente:

- En los primeros experimentos, usábamos la probabilidad de bigramas y de emisión de palabras para ponderar los arcos.
- Posteriormente, añadimos aristas que unían los nodos representantes de una palabra con los nodos de las palabras dos posiciones a la izquierda y derecha. De esta forma, tratábamos de añadir un conocimiento menos local a la red. Con este esquema conseguimos peores resultados que con el anterior.
- La tercera evolución supuso el uso de trigramas en lugar de bigramas. Para ello fue necesario cambiar la construcción del grafo, de forma que para cada palabra se crease un nodo por cada posible etiqueta de la palabra anterior y cada posible etiqueta de la palabra en cuestión. El problema de usar sólo la probabilidad de trigramas es que la frecuencia de aparición de los trigramas en un corpus es mucho menor, encontrándonos con muchas combinaciones de etiquetas no observadas en el corpus. Se necesita por tanto suavizar estos resultados mediante la inclusión de las probabilidades de bigramas y unigramas.
- Investigamos si el proceso de combinación de bigramas y trigramas se podría llevar a cabo simplemente mediante la inclusión de aristas distintas para cada probabilidad, y dejando que el algoritmo de TextRank calculara una puntuación para cada nodo en función de dichas probabilidades. Pero tras distintos experimentos, llegamos a la conclusión de que la mejor solución era usar una sola arista ponderada con una interpolación lineal de los tres tipos de probabilidad. Para el cálculo de los coeficientes de correlación de las probabilidades hemos usado el mismo algoritmo que TnT.

El modelo que mejor se ha comportado y que hemos utilizado por tanto en nuestros experimentos ha sido este último. Para el tratamiento de las palabras desconocidas hemos optado por una estrategia muy simple: las etiquetas más usuales en el corpus son asignadas como candidatas a las palabras desconocidas, y la probabilidad de emisión es calculada en base a la terminación de dichas palabras. En el proceso de extracción de estadísticas del corpus de entrenamiento, se calculan las probabilidades de emisión de las palabras con terminaciones más comunes, y dichas probabilidades son usadas para el cálculo de las probabilidades de emisión de palabras desconocidas en el proceso de etiquetado. Este tratamiento puede ser todavía muy refinado para llegar al nivel de herramientas como TnT, y es de esperar que la inclusión de dichas técnicas mejore sensiblemente los resultados obtenidos en nuestro trabajo, sobre todo ante corpus de entrenamiento pequeños.

4. Diseño experimental

En esta sección presentaremos el diseño experimental que hemos seguido para aplicar nuestras ideas a dos corpus anotados con etiquetas POS (*Part Of Speech*). Para esta tarea se pueden encontrar recursos suficientes y numerosas aproximaciones con las que comparar los resultados. El conjunto de etiquetas suele ser mediano (entre 50 y 100), hay palabras que sólo pueden ser etiquetadas con una etiqueta y otras para las que hay varias posibilidades.

Los dos corpus utilizados están escritos en inglés, uno es el corpus Susanne y otro un extracto del corpus Penn compuesto por sus cuatro primeras secciones, en la tabla 1 se pueden ver los tamaños de las respectivas particiones de entrenamiento y test para ambos corpus.

La diferencia más significativa entre los

	Entren.	Test	Etiquetas
Susanne	141140	15482	131
Penn	198550	46461	37

Tabla 1: Tamaños de los corpus.

dos corpus utilizados es el número de etiquetas. El corpus Penn tiene un conjunto de etiquetas bastante pequeño de sólo 37 etiquetas, mientras que el corpus Susanne casi triplica esa cifra con 131. En la práctica esto se traduce en que la tarea de etiquetar a partir del corpus Susanne es mucho más difícil que con el Penn, tal y como se podrá comprobar en los resultados.

4.1. Otros sistemas

Para comparar los resultados obtenidos con nuestra versión supervisada de TextRank hemos entrenado los dos corpus con herramientas especializadas en tarea del etiquetado POS. En concreto hemos utilizado los siguientes sistemas:

- **TnT** (Brants, 2000), es uno de los más utilizados, está basado en modelos de Markov, es muy rápido y suele obtener muy buenos resultados.
- **TreeTagger** (Schmid, 1994), está basado en árboles de decisión, para cada palabra genera un registro de una base de datos que es posteriormente utilizada para la obtención del árbol de decisión.
- **MBT** (Daelemans, 2004), realiza el entrenamiento mediante aprendizaje basado en ejemplos, una implementación eficiente de la técnica de los vecinos más cercanos.
- **fnTBL** (Ngai, 2001), es una implementación eficiente del método TBL (Brill, 1995) basado en la generación de reglas de transformación guiada por el descubrimiento de errores.
- **MaxEnt** (Baldridge, 2005), que emplea modelado de máxima entropía para integrar, en forma de restricciones, el conocimiento del problema que proporciona el corpus de entrenamiento.

Además, disponemos de un etiquetador simple que nos servirá de línea base para nuestros experimentos. En él, se asocia a cada palabra la etiqueta más repetida para ella en el corpus de entrenamiento.

4.2. TextRank inverso

Además del método supervisado para TextRank presentado en la sección 3 de este artículo, hemos incluido en nuestro grupo de experimentos dos variantes sobre la idea original.

La primera variante, que denominaremos TextRank inverso (TextRankI), consiste en calcular las probabilidades de transición en sentido contrario, procesando el texto de derecha a izquierda. De esta forma, $P(t|t')$ refleja la probabilidad de que la etiqueta t' aparezca después de la etiqueta t, y se estima con la siguiente fórmula a partir de los ejemplos del corpus de entrenamiento:

$$P(t|t') = \frac{C(t,t')}{C(t')}$$

En general, los resultados de esta variante son peores que el tratamiento natural (de izquierda a derecha), pero aportan una visión alternativa del problema que aprovecharemos en la siguiente variante.

4.3. TextRank combinado

La segunda variante (TextRankC) consiste en la combinación de los resultados de TextRank y TextRank inverso. Hemos aplicado la técnica de *stacking*, que consiste en utilizar los resultados de una primera etapa de aprendizaje para entrenar un clasificador de segundo nivel. Con ello se consigue combinar las opiniones producidas por los etiquetadores de la primera etapa de aprendizaje de una manera muy flexible.

La técnica de *stacking* suele dar muy buenos resultados cuando se combinan opiniones complementarias. Esto se suele conseguir o bien utilizando distinto material de entrenamiento, o bien utilizando distintos clasificadores sobre un único conjunto de datos de entrenamiento. Este esquema ha sido aplicado con éxito en diversas tareas, en concreto en el ámbito del Procesamiento del Lenguaje Natural existen trabajos que aplican una doble etapa de decisión para el etiquetado POS (Halteren, 2001), la desambiguación de significados (Florian, 2002), el análisis sintáctico (Henderson, 1999) o el reconocimiento de entidades con nombre (Florian, 2003).

En nuestro caso, para cada palabra del corpus de entrenamiento hemos creado un registro que contiene las tres etiquetas mejor situadas según TextRank y TextRank inverso

para dicha palabra, así como las puntuaciones obtenidas por cada una de las propuestas. El registro se completa con la etiqueta real que es extraída directamente del corpus de entrenamiento.

Con todos los registros obtenidos del corpus de entrenamiento, entrenamos un árbol de decisión que determinará la etiqueta a asignar a una palabra en función de las propuestas de TextRank y TextRank inverso. La figura 1 muestra todos los elementos que participan en el esquema, dentro del recuadro en trazo discontinuo se incluye el proceso que permite convertir el corpus de entrenamiento en base de datos de entrenamiento gracias a la aplicación de los modelos previamente entrenados para TextRank. Un proceso similar se aplica al corpus de test, aunque no se ha detallado en el gráfico para no complicarlo en exceso. Una vez que se dispone de ambas bases de datos, se sigue el esquema clásico de entrenamiento/aplicación de cualquier proceso de aprendizaje automático cuya salida es utilizada para componer el corpus de test anotado.

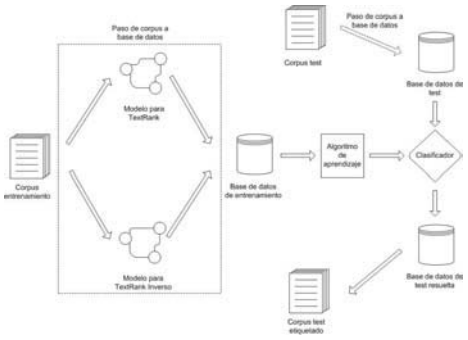

Figura 1: Proceso de Stacking

5. Resultados

Antes de empezar con estos experimentos sabíamos que iba a ser difícil obtener mejores resultados que las herramientas con las que nos compararíamos. La razón: estas herramientas están muy especializadas en la tarea del etiquetado POS e incluyen heurísticas especiales para resolver de la mejor manera posible este problema. Los resultados no nos han sorprendido, aunque nuestra aproximación ha dado muestras de saber adaptarse bien a este problema.

La tabla 2 muestra los resultados de todos los sistemas descritos en este artículo. Se muestra la medida *accuracy* que calcula el porcentaje de palabras del corpus de test que han sido etiquetadas correctamente.

Con respecto la línea base, TextRank supera con creces las establecidas para los dos corpus. En cuanto a la comparativa con otras herramientas, nuestro método (la versión TextRankC) sólo supera a TreeTagger y MBT con el corpus Susanne, se queda bastante cerca del resto de etiquetadores para dicho corpus y un poco más alejado cuando utilizamos el corpus Penn.

	Susanne	Penn
Línea base	79.15 %	80.01 %
TnT	93.61 %	95.48 %
TreeTagger	91.27 %	94.28 %
fnTBL	93.01 %	95.04 %
MBT	91.16 %	94.40 %
MaxEnt	93.09 %	95.47 %
TextRank	90.32 %	92.14 %
TextRankI	89.84 %	91.70 %
TextRankC	91.51 %	93.28 %

Tabla 2: Resultados de los experimentos.

Pese a obtener peores resultados que la mayoría de las herramientas, hay que destacar que nuestro método está aún en una fase muy preliminar. Por ejemplo, no se incluye ninguna heurística especial para las palabras desconocidas. Es de esperar que los resultados se acerquen más a medida que se incluya en la construcción del grafo este tipo de información, con la que cuenta la mayoría de las herramientas con las que lo hemos comparado. También hay que destacar que el método se ha comportado mejor en una tarea más dura (corpus Susanne con más etiquetas) que en una más sencilla (corpus Penn con menos etiquetas entre las que decidir).

5.1. Otras tareas PLN

Nuestro método puede ser aplicado a cualquier tarea siempre que el corpus que la describa sea de dos columnas (palabra-etiqueta). Gracias a la notación IOB es posible especificar tareas en las que se etiquetan grupos de palabras con corpus de dos columnas. De esta forma hemos podido aplicar TextRank supervisado a las siguientes tareas:

- Reconocimiento de entidades con nombre para el español (NER-E), usando el

corpus distribuido para la tarea propuesta en CoNLL 2002.

- Reconocimiento de entidades biomédicas (NER-B), usando el corpus distribuido para la tarea propuesta en COLING 2004.
- Análisis sintáctico superficial (Chunk), usando el corpus distribuido para la tarea propuesta en CoNLL 2000.

En la tabla 3 se pueden ver los tamaños de las respectivas particiones de entrenamiento y test para estas tareas. A pesar de que el conjunto de etiquetas es en todos los casos muy pequeño y que los corpus de entrenamiento son grandes, son tareas tan, o más complejas que las que plantearon los corpus Penn y Susanne ya que presentan un mayor grado de dependencia contextual.

	Entren.	Test	Etiquetas
NER-E	529413	105842	9
NER-B	985102	202078	11
Chunk	423454	94754	23

Tabla 3: Tamaños de los corpus de las tareas NER-E, NER-B y Chunk.

El diseño experimental es el mismo que el que empleamos para los corpus Susanne y Penn. En la tabla 4 se pueden ver los resultados de los distintos sistemas usando la medida *accuracy*.

	NER-E	NER-B	Chunk
Línea base	71.90 %	72.64 %	63.08 %
TnT	94.78 %	88.97 %	89.62 %
TreeTagger	90.58 %	84.79 %	84.40 %
fnTBL	94.30 %	90.49 %	89.54 %
MBT	94.38 %	88.71 %	90.61 %
MaxEnt	95.03 %	87.52 %	92.83 %
TextRank	92.72 %	86.75 %	87.34 %
TextRankI	90.85 %	87.78 %	78.84 %
TextRankC	92.93 %	89.71 %	89.24 %

Tabla 4: Resultados para las tareas NER-E, NER-B y Chunk.

Al igual que para los experimentos POS, la versión combinada de TextRank sigue superando a las otras dos. En cuanto a la comparativa con los otros sistemas, es claramente favorable en la tarea NER-B (donde supera a todos los sistemas salvo a fnTBL), está igualada en Chunk (supera a TreeTagger y está muy cerca de TnT y fnTBL) y más desfavorable para NER-E (supera a TreeTagger y del resto queda a una distancia entre 1.4 y 2.1 puntos). En general podemos decir que el método se adapta bien a estas tareas más contextuales, acercándose más al resto de sistemas que en los experimentos POS.

6. Conclusiones y trabajo futuro

En este trabajo hemos estudiado de qué manera se pueden aplicar las ideas del método no supervisado TextRank para que aproveche la información disponible en un corpus de entrenamiento anotado. Hemos definido un método de construcción de grafos basado en la coocurrencia de palabras, que incluye el conocimiento acumulado en el corpus de entrenamiento mediante probabilidades de emisión y transición similares a las utilizadas en un modelo de Markov.

Hemos realizado un estudio experimental para la tarea POS y hemos comparado los resultados con los obtenidos con herramientas especializadas en este tipo de etiquetado. Los resultados demuestran que el método obtiene un etiquetado satisfactorio, y aunque la comparativa es desfavorable con la mayoría de los casos, llegamos incluso a superar a una de esta herramientas especializadas cuando entrenamos con el corpus Susanne. Hemos reproducido los experimentos para otras tareas PLN como NER o Chunking obteniendo mejores resultados que en el POS en la comparativa con los otros sistemas.

Nuestro trabajo futuro pasa por estudiar formas alternativas de creación del grafo que exploten otras características además de la coocurrencia, e incluir en el método heurísticas que nos permitan presentar resultados más competitivos.

Bibliografía

Baldridge, J., Morton, T., Bierner, G.: Maxent, Mature Java package for training and using maximum entropy models. An *OpenNLP* project. (2005)

Brants, T.: TnT. A statistical part-of-speech tagger. In *Proceedings of the 6th Applied NLP Conference (ANLP00)*. USA (2000)

Brill, E.: Transformation-based error-driven learning and natural language processing:

A case study in part of speech tagging. In *Computational Linguistics* 21(4), (1995)

Brin, S., Page, L.: The anatomy of a large-scale hypertextual Web search engine. In *Computer Networks and ISDN Systems*. (1998)

Daelemans, W., Zavrel, J., van der Sloot, K., van den Bosch, A.: TiMBL Memory Based Learner, version 5.1, Reference Guide. ILK Research Group Technical Report Series no. 04-02. The Netherlands (2004)

Florian, R., Yarowsky, D., 2002. Modeling Consensus: Classifier Combination for Word Sense Disambiguation. In *Proceedings of EMNLP'02*, Philadelphia, pp 25–32.

Florian, R., Ittycheriah, A., Jing, H., Zhang, T., 2003. Named Entity Recognition through Classifier Combination. In *Proceedings of CoNLL-2003*, Canada, pp 168–171.

Halteren, v.H., Zavrel, J., Daelemans, W., 2001. Improving accuracy in word class tagging through the combination of machine learning systems. *Computational Linguistics 27*, pp 199–230.

Henderson, J.C., Brill, E., 1999. Exploiting diversity in natural language processing. Combining parsers. In *1999 Joint Sigdat Conference on Empirical Methods in Natural Language Processing and Very Large Corpora, ACL*, USA, pp 187–194.

Mihalcea, R., Tarau, P.: TextRank. Bringing Order into Texts. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Barcelona, Spain (2004)

Mihalcea, R., Tarau, P. Figa, E.: PageRank on Semantic Networks, with application to Word Sense Disambiguation. In *Proceedings of The 20st International Conference on Computational Linguistics* . Switzerland, Geneva (2004)

Ngai, G., Florian, R.: Transformation-based learning in the fast lane. In *Proceedings of North Americal ACL 2001*, (2001)

Radev, D., Mihalcea, R. (organizers): Graph-based Algorithms for Natural Language Processing. Workshop at *HLT/NAACL*. New York, USA (2006)

Schmid, H.: Probabilistic Part-of-Speech Tagging Using Decision Trees. In *International Conference on New Methods in Language Processing*. Manchester, UK (1994)