Reconocimiento de Entidades con Nombre

Tutor
Diego Garat

Estudiantes
Claudia García
Yamandú González

Informe de Proyecto de Grado
Facultad de Ingeniería
Universidad de la Republica
Año 2005

RESUMEN

Una entidad con nombre es una expresión lingüística que denota a una persona (PER), un lugar (LOC), una organización (ORG), etc. Por ejemplo:[PER Wolff] , actualmente periodista en [LOC Argentina], jugó con [PER Del Bosque] a finales de los años setenta en el [ORG Real Madrid]. Identificar entidades con nombre en textos ha demostrado ser un componente importante de tareas como la extracción de información, corrección ortográfica, etc. Dentro de las opciones posibles para realizar esta tarea, el uso de técnicas de aprendizaje permite obtener sistemas más flexibles a cambios de contexto (por ejemplo, cambios en el idioma). En este proyecto se hace un estudio de diversas técnicas de aprendizaje automático para la identificación y clasificación de entidades con nombre, para luego implementar algunas de estas técnicas en una herramienta. Para esto se estudian soluciones propuestas en las conferencias CoNLL ("Conference on Computational Natural Language Learning"). Estas conferencias constituyen un espacio en el que se discuten y presentan trabajos de investigación en el área de la lingüística computacional, utilizando técnicas de aprendizaje automático. Se demuestra que la combinación de varios clasificadores simples, integrando diversas técnicas, provee mejores resultados que un individual, lográndose resultados satisfactorios para el clasificador implementado. Estos resultados son comparados con las soluciones propuestas en las CoNLL, superando alguna de estas. Además, se comparan los resultados con dos sistemas basados en reglas donde se obtienen resultados similares en el reconocimiento.

Finalmente, como una forma de ver como se comporta el clasificador implementado en un corpus distinto al provisto en las CoNLL, se clasifican los artículos de las páginas del sitio web de la ONG Choike (www.choike.org), para el cuál se alcanzan buenos resultados.

Palabras Claves: *Reconocimiento de Entidades con Nombre (NER), Clasificación de Entidades con Nombre (NEC), Boosting, K-NN, Árboles de Decisión, Método de Votación.*

AGRADECIMIENTOS

A nuestras familias por acompañarnos y apoyarnos en este camino.

A nuestros jefes y compañeros de trabajo por la paciencia y la compresión prestada.

A Isabel, Maria y Lorena, especialmente a estas dos últimas que ayudaron en la corrección de este documento.

A Rafael, Camila y Pepe por facilitarnos sus computadoras para correr las distintas pruebas realizadas.

A Diego por su paciencia y disponibilidad.

INDICE

1 Introducción

1.1 Problema

Una entidad con nombre es una expresión lingüística que denota a una persona (PER), un lugar (LOC), una organización (ORG), etc. Por ejemplo:

> [PER Wolff] , actualmente periodista en [LOC Argentina], jugó con [PER Del Bosque] a finales de los años setenta en el [ORG Real Madrid].

El espectro de tareas que consideran estas entidades es amplio, destacando dos en especial: Reconocimiento de Entidades con Nombre (NER) y Clasificación de Entidades con Nombre (NEC).

La identificación de entidades ha demostrado ser una tarea preliminar muy útil en los sistemas de procesamiento del lenguaje natural, por ejemplo, en problemas como la extracción y recuperación de información. También se destaca en sistemas de respuestas automáticas, sistemas de restauración de la capitalización de un texto escrito solo en mayúsculas o minúsculas, corrección ortográfica (evitando accidentalmente corregir nombres que son confundidos con palabras mal escritas), etc.

El reconocer entidades con nombre es una tarea no trivial, dado que una misma secuencia de palabras en distintos contextos puede representar o no una entidad con nombre. Más compleja aún es su clasificación, dado que a una misma entidad le pueden corresponder distintas clasificaciones de acuerdo a su contexto. Por ejemplo, la entidad con nombre 'Washington' puede denotar un lugar o una persona.

1.2 Motivación

Muchos esfuerzos de investigación previa en construir sistemas reconocedores de entidades con nombre han recaído en el uso de reglas escritas mano. El funcionamiento de estos sistemas se basa en la definición de un rico conjunto de reglas gramaticales y/o de inmensos diccionarios de datos, etc. Estas soluciones tienen dos desventajas fundamentales. La primera es que la mayoría de estos sistemas trabajan bien bajo ciertas circunstancias capturadas por los diseñadores al momento de escribir las reglas. Sin embargo, tanto el idioma como las necesidades de sus usuarios tienden a evolucionar con el paso del tiempo. Segundo, no es claro qué recursos adicionales son requeridos para adaptar el sistema a otros lenguajes.

A pesar de que es asumido que mejoras en el procesamiento del lenguaje natural serán hechas integrando información lingüística y técnicas estadísticas, la realidad es que el lenguaje es muy diverso. Esto lleva a sistemas muy estáticos y dependientes del contexto, por lo que surge la idea de emplear técnicas de aprendizaje automático en el reconocimiento de entidades con nombre. El objetivo de esto es crear sistemas más flexibles a cambios de contexto, ya sea cambios en el idioma o en la naturaleza de los textos a procesar: no es la misma realidad la que se presenta en artículos de prensa que en libros de poesía.

El campo del aprendizaje automático esta muy relacionado con la pregunta de cómo construir software que automáticamente mejore con la experiencia. Estos sistemas cuentan con la ventaja de ser capaces de adquirir e integrar conocimiento en forma autónoma y de aprender de la experiencia pasada para así mejorarse a sí mismos, aumentando su eficiencia y efectividad.

En años recientes varios algoritmos han sido desarrollados, los cuáles son efectivos para distintos tipos de tareas, así como ha habido avances importantes en el conocimiento teórico sobre el aprendizaje automático. Muchas aplicaciones prácticas que encuentran efectiva la utilización de técnicas de aprendizaje automático han comenzado a surgir; estas van desde la minería de datos (data-mining), con el fin de detectar fraudes en las transacciones de tarjetas de crédito, hasta programas que filtran información y aprenden las preferencias de usuarios.

Para problemas de reconocimiento en el área de la lingüística, los algoritmos basados en aprendizaje automático han obtenido resultados notoriamente mejores que otros enfoques utilizados hasta el momento. En la actualidad existen varios trabajos que utilizan aprendizaje automático para el reconocimiento y clasificación de entidades con nombre. En particular resulta de gran interés concentrar la atención en los trabajos presentados en las conferencias CoNLL en los años 2002 y 2003 donde se atacó este problema.

1.3 Antecedentes

CoNLL (Conference on Computational Natural Language Learning) es una conferencia internacional, en la que se discuten y presentan trabajos de investigación en el área de la lingüística computacional, donde estos deben utilizar técnicas de aprendizaje automático. Todos los años durante estas conferencias, se realizan lo que se denominan “tareas compartidas”: se plantea un problema a resolver, los equipos implementan una solución, y luego se comparan los distintos resultados.

Durante las CoNLL de los años 2002 [13] y 2003 [14], la tarea compartida consistió en el reconocimiento de entidades con nombre y su posterior clasificación en alguna de las siguientes categorías: personas (PER), lugares (LOC), organizaciones (ORG) y misceláneos (MISC). Está última categoría engloba todas las entidades que no son comprendidas por las tres primeras

Antes de las CoNLL 2002, diferentes sistemas han sido evaluados en la tarea de reconocer entidades con nombre en Sixth Message Understanding Conference (MUC6) en 1995 [12]. EL objetivo en las MUC6 era reconocer entidades con nombre para el idioma ingles. Los sistemas presentados obtuvieron buenos resultados, con la gran desventaja que muchos de éstos eran totalmente dependientes del lenguaje. Luego de las MUC6, sistemas reconocedores de entidades con nombre fueron desarrollados para lenguajes europeos y asiáticos. Al lo menos dos trabajos han aplicado a su sistema a diferentes lenguajes. Por ejemplo, Palmer y Day [9] utilizaron métodos estadísticos para encontrar entidades con nombre en artículos en los idiomas chino, inglés, francés, japonés, portugués y español.

En esta tarea, los autores encontraron que era distinta para cada lenguaje, aunque gran parte de la tarea podía realizarse mediante métodos simples. Cucerzan and Yarowsky [5] usaron pistas morfológicas y contextuales para identificar entidades con nombre en ingles, griego, indio, rumano y turco..

Dada la importancia del reconocimiento de entidades con nombre y la dependencia del lenguaje a la que estaban sujetos los sistemas existentes resulta de gran interés encontrar técnicas que permitan reconocer entidades con nombre lo más independientemente del lenguaje posible, siendo este el fin perseguido en las CoNLL 2002 y 2003.

Respecto a las CoNLL se puede agregar que los resultados del 2003 son ampliamente mejores que los del 2002. Esto es debido a que se logran grandes mejoras al combinar varios algoritmos de aprendizaje, en lugar de utilizar uno solo como se hizo en el 2002. Dentro de los trabajos que dieron mejores resultados las técnicas de aprendizaje mas utilizadas fueron *Boosting*, Transformation Based Learning (TBL), enfoques de máxima entropía, K-NN y cadenas de Markov. Las formas empleadas para combinar los distintos clasificadores van desde la más simple, como ser votación por mayoría, hasta las más complejas, como pueden ser votación ponderada, *stacking*, etc... Para profundizar en las soluciones propuestas, así como en los resultados obtenidos en las conferencias mencionadas, ver los Anexos I y II.

En el contexto de este proyecto se estudian las soluciones propuestas y se implementa una solución propia, para luego ser comparada con los resultados obtenidos en las conferencias del 2002 para el idioma español.

1.4 Aprendizaje y NEC

Para entender el entorno en el que se desarrolla el proyecto es necesario dar una explicación más refinada del concepto de algoritmo de aprendizaje, comenzando por definir que es una instancia.

Una instancia es la representación que se elige para aquel objeto que se quiere predecir su comportamiento o clase. Por ejemplo, en el contexto de este proyecto una palabra representada por sus dos primeras letras (prefijo), sus dos últimas letras (sufijo) y por un bit que indica si esta comienza por mayúsculas sería una instancia para la cuál se intenta determinar si pertenece o no a una entidad con nombre. Una forma de representar las instancias (la forma usada en el transcurso de este documento) es mediante un conjunto de atributos, donde cada uno tiene un rango de posibles valores (discretos o no), más un atributo en especial que determina la clase. Esta representación es de la forma $<a_1,\ldots,a_n,a_c>$, donde, $a_1,..a_n$ denotan los valores tomados por los atributos de la instancia y a_c denota la clase a la que pertenece. Entonces dada una instancia cuya clase se desconoce y un conjunto de instancias para las cuáles si se conoce, determinar la clase de la primera a partir de las segundas es la incógnita que se trata de resolver mediante aprendizaje automático.

Un algoritmo de aprendizaje funciona básicamente en dos etapas. Una primera etapa de entrenamiento y una segunda de clasificación. En la etapa de entrenamiento se provee al algoritmo de un conjunto de instancias (por ejemplo, textos donde las entidades con nombre están identificadas) para las cuáles se conoce su clase (lugar, persona, etc.) con el fin de que ésta información sea procesada y oficie de "experiencia" para el algoritmo.

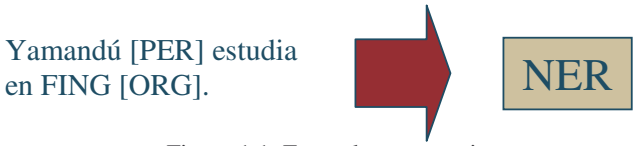

Figura 1.1: Etapa de entrenamiento

La etapa de clasificación consiste en aplicar el algoritmo previamente entrenado, sobre un conjunto de instancias para las cuáles no se conoce su clase, de forma que ésta sea determinada en base a la experiencia pasada.

Figura 1.2: Etapa de clasificación

La mayor ventaja de estas técnicas es que tienen la capacidad de aprender y luego aplicar el conocimiento adquirido sin conocer la naturaleza de lo que se está procesando. En otras palabras, permiten establecer patrones y comportamientos sobre los datos sin saber de dónde provienen o qué significan en el mundo real, aunque se debe considerar que los datos son provistos por personas u otros sistemas que los manipulan, agregando cierto conocimiento implícito con el fin de mejorar los resultados.

Debido a estas ventajas, se encuentra conveniente la aplicación de aprendizaje automático al reconocimiento de entidades con nombre: además de dar una mayor (aunque no total) independencia del lenguaje, es fácil integrar nuevas fuentes de conocimientos, como ser diccionarios.

1.5 Objetivo

Este proyecto tiene como objetivo el estudio de diversas técnicas de aprendizaje automático para la identificación y clasificación de entidades con nombre. Se busca implementar algunas de estas técnicas en una herramienta (clasificador).

Se demuestra que la combinación de varios clasificadores simples provee mejores resultados que un clasificador individual, logrando resultados satisfactorios para el clasificador implementado, superando otras soluciones propuestas.
El resto del proyecto consiste en comparar el clasificador obtenido con dos sistemas basados en reglas fruto de otros proyectos de grado. Por último como una forma de ver como se comporta el clasificador implementado en otros casos distintos al corpus provisto por las CoNLL, se clasifican los artículos de las páginas del sitio Web de Choike[1].

1.6 Organización del documento

En el capítulo 2 se explican aspectos generales de la solución propuesta, así como algunos conceptos que son usados en todo el documento.

En los capítulos 3, 4 y 5 se explican en detalle los clasificadores implementados, las diferentes decisiones de diseño que se debieron tomar en pro de mejorar el desempeño del mismo y se exponen los resultados obtenidos durante el desarrollo de dicho clasificador.

En él capitulo 6 se presenta el clasificador combinado y los resultados obtenidos al probar distintas formas de combinar los clasificadores presentados en los capítulos 3, 4 y 5.

El capítulo 7, se comparan los resultados del clasificador aquí presentado con dos clasificadores basados en reglas.

En el capitulo 8, se evalúan los resultados obtenidos al clasificar los artículos del sitio Web de la ONG Choike con el clasificador presentado.

En el capitulo 9, se presentan las conclusiones de este trabajo junto con una reseña de trabajos a futuro, que los autores creen se podrían seguir a partir de este proyecto.

[1] Choike es una ONG uruguaya que trata temas sociales del cono sur, su sitio web es www.choike.org

2 Aspectos generales

Como se explica en los siguientes capítulos la solución propuesta es la combinación de varios clasificadores simples, aunque estos son distintos entre sí, existen varios aspectos comunes que se explican en este capítulo. Además los clasificadores fueron entrenados y testeados con los mismos conjuntos (corpus) de entrenamiento, por lo cuál también se explica que datos proveen y como estos están presentados.

Primero se presentan los conjuntos de datos usados durante el desarrollo, luego se hace una reseña general respecto a la solución implementada y finalmente se explican las características en común entre los diferentes clasificadores.

2.1 Corpus

En las CoNLL se provee un juego de archivos por cada lenguaje propuesto, tanto para el entrenamiento como para el testeo del clasificador. En este proyecto se utiliza el corpus provisto por los organizadores de las CoNLL 2002 para el idioma español. El corpus se divide en tres archivos con textos obtenidos de Reuters [10]: Un archivo de entrenamiento, un archivo de desarrollo y un archivo de testeo. En primer lugar, con el fin de ajustar los parámetros de los métodos de aprendizaje, se usan el primer y segundo corpus para entrenar y testear respectivamente. Esta etapa puede constar de varias iteraciones (entrenar, ajuste, testear), hasta obtener el juego de parámetros con el cuál se logra el mejor desempeño. Luego para evaluar el desempeño de la técnica se testea el clasificador con el tercer corpus, previamente (en caso de ser necesario) se entrena con el primer conjunto de datos. El resultado de este último test será comparado con el obtenidos por un sistema base provisto por las CoNLL.

Todos los archivos de datos contienen una palabra por línea con líneas vacías representando límites de oraciones. Adicionalmente cada línea contiene una etiqueta cuyo estado indica si la palabra esta dentro de una entidad con nombre o no. La etiqueta también indica el tipo de entidad con nombre. Por ejemplo, para la oración:

Wolf B-PER
, O
un O
periodista O
en O
Argentina B-LOC
, O
visito O
a O
Del B-PER
Bosque I-PER

Las palabras etiquetadas con O están por fuera de una entidad con nombre. La etiqueta B-XXX es usada para la primera palabra en una NE de tipo XXX y I-XXX es usada para todas las restantes palabras en una NE de tipo XXX. Los datos contienen entidades de cuatro tipos: personas (PER), organizaciones (ORG), lugares (LOC) y misceláneos (MISC).

Lamentablemente sólo se puede tener acceso a los conjuntos de datos para el idioma español y el holandés, no resultando este último de interés para este proyecto de grado. En un principio, el objetivo consistía en aplicar la solución propuesta a los idiomas ingles y español, por lo que este proyecto queda limitado a este último. El conjunto de datos de entrenamiento contiene más de 240000 palabras, proporcionando así un texto de gran tamaño, lo que es fundamental para un correcto entrenamiento del algoritmo de aprendizaje. Los conjuntos de desarrollo y testeo contienen alrededor de 54000 palabras.

2.2 Definición del clasificador

Del análisis de los trabajos presentados en las CoNLL se desprende que los mejores resultados fueron obtenidos por los trabajos que combinaron el resultado de clasificadores basados en distintas técnicas de aprendizaje. Esto se debería a que los distintos algoritmos empleados tienen sesgo de aprendizaje distinto, y al combinarlos se atenúan los errores de clasificación cometidos por cada uno de ellos. Visto esto se decidió implementar un clasificador que combinara tres técnicas de aprendizaje automático (*Boosting*, K-NN y Árboles de Decisión). En otras palabras, se construye un clasificador por cada una de estas técnicas, los cuáles son combinados mediante algún método para obtener un único clasificador.

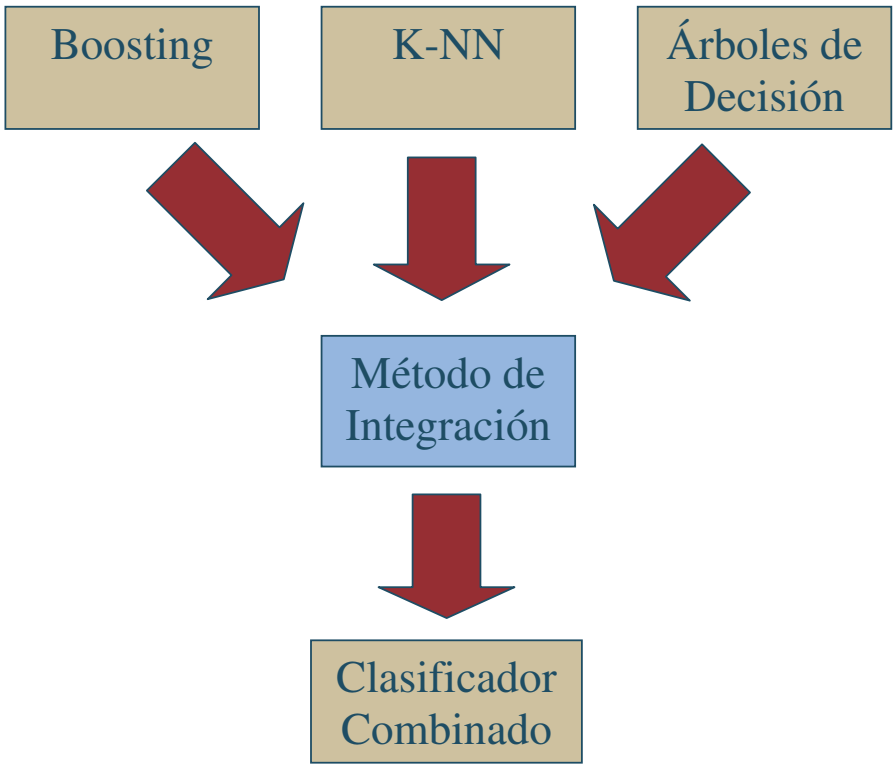

Figura 2.1: Bosquejo de la solución propuesta.

Clasificador basado en Boosting: En el año 2003 los resultados obtenidos por los trabajos basados únicamente en esta técnica de aprendizaje no estuvieron entre los primeros puestos, ya que estos fueron ocupados por clasificadores que consistían en la combinación de varias técnicas (*Boosting* entre ellas). Por el contrario en el año 2002, donde uno de los idiomas a reconocer era el español, los trabajos que obtuvieron los mejores resultados principalmente utilizaban este método.

Clasificador basado en K-NN: Esta técnica fue elegida debido a que obtuvo buenos resultados en ambas CoNLL y a que es una estrategia fácil de entender e implementar. A diferencia de *Boosting*, K-NN utiliza las instancias en su representación original como medio principal de clasificación, lo que lleva a que esta técnica este comprendida en lo que se denomina métodos basados en memoria. En cambio *Boosting* procesa las instancias previamente para obtener un objeto que encapsula las características de estas.

Clasificador basado en árboles de decisión: En un principio no se tenía pensado integrar una tercera estrategia de aprendizaje a nuestro clasificador. Debido a que los resultados de combinar las dos primeras fueron malos se vio la necesidad de integrar un tercer clasificador que "desempatara". El por qué de usar árboles de decisión se debe a que esta técnica es de rápida implementación. De esta forma, se puede contar lo antes posible con resultados que permiten estudiar y mejorar la integración de clasificadores.

Un punto que no puede pasarse por alto es que para la implementación de todos los componentes de este proyecto se eligió java como lenguaje de programación. Esto se debe a que para dicho lenguaje se cuenta con un software (distribuido baja la licencia pública GNU), fruto del proyecto Weka [17]. Este paquete provee clases que implementan diversas técnicas de Data Mining y de Aprendizaje Automático, entre estas, K-NN, *Boosting* y Árboles de Decisión. También provee funcionalidades que facilitan el pre-procesamiento de datos, implementaciones de métodos de evaluación así como un ambiente que facilita la comparación de algoritmos de aprendizaje.

El desarrollo que se explicará más adelante, consistirá en el estudio y selección de aquellos atributos para una instancia que muestran mejor desempeño, sin detenerse en la implementación de las técnicas en sí.

La medida tenida en cuenta para evaluar los resultados del proyecto será la misma utilizada en las CoNLL:

$$F_{\beta=1} = \frac{(\beta^2 + 1) * precisión * recuperación}{\beta^2 * precisión + recuperación}$$

donde:

- *Precisión*: Porcentaje de Entidades con nombre encontradas por el sistema que son correctas.
- *Recuperación:* Porcentaje de Entidades presentes en el corpus que son encontradas por el sistema.
- $\beta = 1$.

2.3 Características empleadas en la función de aprendizaje

Como fue mencionado, la 'inteligencia' del clasificador viene dada por una función de aprendizaje. Ésta debe contemplar las características consideradas de interés para el problema que se desea aprender. En el caso de reconocimiento de entidades con nombre, éstas pueden venir dadas por partes de las palabras, si están escritas con mayúsculas, etc. A continuación describiremos las características más importantes empleadas por cada uno de los clasificadores implementados.

Morfología de la palabra: Consiste en indicar como está escrita la palabra: en mayúsculas, minúsculas ó una mezcla de éstas. Es de interés además si ésta se encuentra al comienzo de una oración. Esto último se debe a que en el idioma español una palabra comenzando con mayúsculas tiene grandes posibilidades de ser o pertenecer a una entidad con nombre. Notar también que si la palabra es el principio de una oración estas posibilidades disminuyen.

Contiene Digito: Tiene el objetivo de identificar entidades con nombre formadas parcial o totalmente por dígitos. Ejemplos de esto son, el número de emergencias '911', el buscador de Internet 'A9'.

Contiene *puntos:* Palabras conteniendo puntos pueden denotar entidades con nombre, como ser 'www.fing.edu.uy'. Dentro de las entidades con nombre conteniendo puntos, casos particulares son los acrónimos y las iniciales. Los primeros; por lo general, denotan organizaciones o países, como ser F.M.I., EE.UU. Por otro lado, entidades que denotan personas muchas veces son escritas con la inicial del nombre seguida del apellido, como ser el caso 'J. Perez'.

Es Romano: Varias entidades con nombre contienen números romanos, por ejemplo 'Carlos V' o 'XIX Feria del Libro'. El identificar éstos puede contribuir a clasificar entidades con nombre del tipo misceláneo.

Contiene guión: Sirve para encontrar entidades con nombre que contienen guiones, así como 'Corea-Japón' o 'sub-17'.

Símbolo de puntuación: En los archivos provistos por las CoNLL, los fines de oración son representados por una línea conteniendo un punto seguida de una en blanco. Ahora bien, existen otros caracteres en el idioma español denotan fin de oración (como ser '?'). Importa conocer tanto el fin como el comienzo de una oración, por esto, se indica si una palabra corresponde a alguno de los siguientes caracteres '?', '¿', '!', '¡', ',', ';'.

Es comilla: Con la finalidad de identificar entidades con nombre que se encuentran entre comillas, se indica cuando una palabra es una comilla doble o simple.

Es paréntesis: En los textos provistos por las CoNLL, aparecen secuencias de palabras como ser 'Fondo Monetario Internacional (FMI)' o 'Sao Paulo (Brasil)'. Notar que, palabras comenzando con mayúscula dentro de paréntesis que siguen a una entidad con nombre, también lo son. Por lo que se marca cuando una palabra es un paréntesis de apertura o cierre.

Es palabra funcional: Palabras de hasta tres letras en minúsculas entre dos palabras que son clasificadas como entidades con nombre, pueden en realidad ser parte de una entidad con nombre formada por ésta y las otras dos mencionadas. Un ejemplo de esto es 'Río *de* Janeiro'.

Largo de la palabra: Se observó en el texto de entrenamiento que palabras muy largas por lo general denotan partes de entidades con nombre. Así que se emplea el largo de las palabras hasta 8.

Pos tags: Éstos son características gramaticales de las palabras, pudiendo llegar a ser de gran ayuda al reconocer entidades con nombre. Se anotaron los conjuntos de entrenamiento, testeo y desarrollo provistos por los organizadores de las CoNLL con Freeling [2].

Es palabra disparadora: Estas son palabra que pueden indicar el comienzo de una entidad con nombre. Un ejemplo de una entidad conteniendo una entidad con nombre es 'doctor Pedro Suar', donde 'doctor' es una palabra disparadora. Otros ejemplos de éstas son: 'presidente', 'hospital', 'hotel', 'ingeniero', etc.

Afijos: Se utilizaron los afijos, que consisten en las primeras y últimas dos letras de la palabra (prefijo y sufijo) en minúsculas. Por ejemplo el prefijo "Ke" en una palabra (Kenia), puede denotar una entidad, debido a que estas dos letras no son comunes al comienzo de una palabra.

Pertenece a diccionario: Palabras que se encuentran en diccionarios de nombres propios, apellidos, lugares, organizaciones, etc., tienen una gran posibilidad de pertenecer o ser entidades con nombre.

Etiqueta de entidad de la palabra predecesora: Para muchas palabras, la clasificación a asignarle puede depender en gran medida de la asignada a la palabra que la precede. Por ejemplo, en la frase 'Luis Martínez es hijo', si 'Luis' es clasificada como comienzo de entidad, teniendo ésta información al clasificar 'Martínez' parecería más correcto asignarle una etiqueta de palabra intermedia a entidad que de comienzo.

A continuación, a modo de ejemplo se mostrará cómo se calculan algunas de las características para una frase extraída de los textos del corpus. Dada la gran cantidad de características descritas, en este ejemplo se trabajará solamente con algunas de ellas. Éstas son: morfología de la palabra *(case)*, contiene dígitos (*dígito*), afijos (*prefijo y posfijo*), símbolo de puntuación (*puntuación*), es paréntesis (*paréntesis*), largo de la palabra (*largo*) y pos tags (*pos*).

La característica c*ase* (que captura la morfología) para palabras no nulas (contienen al menos un carácter), vale:

0: Si la palabra no esta formada solo por letras.
1: Si todas las letras son mayúsculas
2: Si todas las letras son minúsculas.
3: Empieza con mayúsculas seguida de minúsculas y es principio de oración.
4: Empieza con mayúsculas seguida de minúsculas y no es principio de oración.
5: Es una mezcla de mayúsculas y minúsculas.

Para las características *puntuación y dígito* para palabras no nulas, toma valor 1 si se cumple, 0 en caso contrario.

La característica *paréntesis* para palabras no nulas toma los siguientes valores: 2 si la palabra es paréntesis de apertura, 1 si es paréntesis de cierre; 0 en caso contrario.

Las características afijos y prefijos en caso de que los caracteres que la forman no sean letras o números toma el valor 0.

Para todas las características, si la palabra es nula toma valor –1.

La frase elegida para procesar es: 'Sao Paulo (Brasil), 23 may. En un balance que'.

Calcularemos primero el vector de características para la palabra 'Sao'. 'Sao' comienza con mayúsculas seguidas de minúsculas y es principio de oración, por lo que la característica *case* toma el valor 3. No contiene dígitos, no es signo de puntuación y no es un paréntesis, por lo que éstas características toman valor 0. Luego, sus dos primeras letras son 'sa' y sus dos últimas 'ao', siendo éstos su prefijo y su posfijo respectivamente. El largo de la palabra es 2 y su pos tag según Freeling es NC. En la tabla 2.2 se muestra la frase procesada:

palabra	Case	Digito	prefijo	posfijo	puntuación	Paréntesis	Largo	Pos	Clasif
Sao	3	0	sa	oa	0	0	3	NC	B_NE

Tabla 2.1: El vector de características para 'Sao' es el siguiente.

palabra	Case	Digito	prefijo	posfijo	puntuación	Paréntesis	Largo	Pos	Clasif
Sao	3	0	sa	ao	0	0	2	NC	B_NE
Paulo	4	0	pa	lo	0	0	5	VMI	I_NE
(	0	0	0	0	0	0	1	Fpa	O
Brasil	4	0	br	il	0	0	6	NC	B_NE
)	0	0	0	0	0	0	1	Fpt	O
,	0	0	0	0	0	0	1	Fc	O
23	0	1	23	23	0	0	2	Z	O
May	2	0	ma	ay	0	0	3	NC	O
.	0	0	.	.	1	0	1	Fp	O
	-1	-1	-1	-1	-1	-1	-1	-1	O
En	4	0	en	en	0	0	2	SP	O
Un	2	0	un	un	0	0	1	DI	O

Tabla 2.2: Vector de características para cada una de las palabras del texto seleccionado

2.4 Contexto de las palabras

Ahora bien, resulta de interés, no solamente considerar la palabra procesada, sino también su contexto. Esto se debe a que la misma palabra en distintos contextos puede pertenecer o no a una entidad con nombre. Por ejemplo, en la secuencia de palabras: "Río de Janeiro", la palabra "de" es parte de una entidad con nombre; pero en la secuencia "Saltó de la cama" la palabra "de" ni representa ni es parte de una entidad con nombre. También existen entidades con nombre que dependiendo del contexto, es la clasificación que se les debe asignar. Por ejemplo, la entidad con nombre 'Washington' puede corresponder a una persona pero también puede denotar un lugar.

En consecuencia, las palabras no fueron consideradas en forma aislada sino que fue tenido en cuenta su contexto. O sea, cada una de las instancias de la función de aprendizaje es formada por secuencias de palabras. Cabe señalar que las ventanas son locales a cada oración: cuando se llega a un carácter que denota el fin de ésta, las palabras de la ventana que la siguen se vuelven nulas (palabras sin caracteres). En forma análoga, al comenzar una oración, las palabras de la ventana anteriores a la oración considerada se transforman en nulas.

Veremos ahora el contexto para la palabra 'Sao'. Ésta es comienzo de oración, por lo que las dos palabras que la preceden son nulas. El resto de su contexto ésta dado por las dos palabras que la siguen: 'Paulo' y '('.

Luego de procesar 'Sao', se debe procesar 'Paulo'. El contexto de ésta viene dado por: sus dos predecesoras las cuales son una palabra nula y 'Sao'; y las que la siguen en la oración '(' y 'Brasil'. Luego, para construir la ventana de la siguiente palabra, se hace un desplazamiento hacia la izquierda de la actual y se agrega como palabra final de ésta la siguiente de la oración. Se procesa en forma análoga hasta llegar a la palabra '.', ésta denota fin de oración por lo que las palabras que la siguen en la ventana se vuelven nulas. En la siguiente figura se ilustra esto:

w-2	w-1	w	w+1	w+2
		Sao	Paulo	(
	Sao	Paulo	(	Brasil
Sao	Paulo	(	Brasil	)
Paulo	(	Brasil	)	,
(	Brasil	)	,	23
Brasil	)	,	23	may
)	,	23	may	.
,	23	may	.	
23	may	.		
		En	un	balance
	En	un	balance	que

Figura 2.2: Ventanas de 5 palabras para de la secuencia palabras seleccionada.

La entrada para el clasificador, es un vector formado por los vectores de características para cada una de las palabras de la ventana. Ver la siguiente tabla:

La ventana para la palabra 'Sao' es la siguiente:

		Sao	Paulo	(

Figura 2.3: Ventana de la para la palabra 'Sao'.

La entrada del clasificador seria:

car_{vacia}	car_{vacia}	car_{Sao}	car_{Paulo}	$car_{)}$	NE_w

Figura 2.4: Entrada del clasificador.

Donde:

car_w_{-2}	-1	-1	-1	-1	-1	-1	-1	-1	-1
car_w_{-1}	-1	-1	-1	-1	-1	-1	-1	-1	-1
car_w	3	0	sa	ao	0	0	0	2	NC
car_w_{+1}	4	0	pa	lo	0	0	0	5	VMI
car_w_{+2}	0	0	0	0	0	0	0	1	Fpa
NE_w	B_NE								

Figura 2.5: Representación de la ventana para el clasificador.

3 Clasificador basado en *Boosting*

El primero de los clasificadores implementados usa como técnica de aprendizaje *boosting*. Esta técnica que fue elegida debido a los buenos resultados obtenidos por los trabajos que la emplearon en las CoNLL.

Se comenzara por dar una breve explicación del algoritmo de *boosting*. Luego se explicará la aplicación de esta técnica al reconocimiento de entidades con nombre. Finalmente se discutirán los resultados del clasificador implementado, y se comparan los resultados obtenidos con los trabajos presentados en las CoNLL 2002 que utilizaron esta técnica.

3.1 *Boosting*

Boosting se refiere a un método general y efectivo que busca producir una regla de predicción muy precisa a partir de la combinación de varias reglas sencillas. Trabaja corriendo un algoritmo de aprendizaje débil[2] reiteradas veces (rondas) sobre los datos de entrenamiento levemente alterados. Luego, se combinan las salidas de éstos en un clasificador final. Así, se llega a un resultado mejor que el que daría cada uno de los clasificadores por separado.

AdaBoost es una implementación de *boosting*. Este fue presentado por Freund y Schapire [6] en 1995. El pseudo código se encuentra en la figura 3.1.

El funcionamiento de Adaboost se muestra en la figura 3.2. Primero se asigna igual peso a cada uno de los ejemplos del conjunto entrenamiento. En cada ronda, un clasificador débil es entrenado con los datos del conjunto de entrenamiento y sus pesos asociados. El paso siguiente consiste en verificar cuales ejemplos del conjunto entrenamiento fueron clasificados correctamente y cual no. Para los que bien clasificados, se disminuye su peso; para el resto es aumentado. De ésta forma, Adaboost se asegura que en la siguiente ronda, el clasificador débil a entrenar hará mayor hincapié en los ejemplos mal clasificados.

En cada ronda se obtiene como resultado un clasificador débil, la eficacia de éste es medida mediante α (ver figura 3.1). Finalmente, se construye una hipótesis final que es la combinación de éstos. Ésta se realiza mediante votación ponderada[3] de los clasificadores débiles, siendo el peso de cada uno de éstos α.

[2] Un clasificador débil de aprendizaje consiste en un clasificador fácil de construir pero que no da muy buenos resultados

[3] Una votación ponderada consiste en tener varios algoritmos cada uno con un peso asociado. Dadas las salidas de los distintos algoritmos cada una pesa en la decisión final tanto como el peso asignado al algoritmo que la dio.

Dados…

- $E = \{(x_1, y_1),..,(x_n, y_n)\}$ conjunto de ejemplos clasificados donde $x_i \in X$ e $y_i \in Y$
- El algoritmo de aprendizaje débil debe poder manejar conjuntos de ejemplos ponderados. Lee un conjunto E y una distribución D. El algoritmo intenta encontrar una hipótesis h con mínima probabilidad de estar clasificando mal, dado un ejemplo sacado de X con respecto de D.

El algoritmo…
$D_t(i)$ denota el peso de un ejemplo i en la ronda t.

1. Inicialización, asignar a cada ejemplo $(x_i, y_i) \in E$ el peso $D_1(i) = 1/n$
2. desde $t = 1$ hasta T
   a. Ejecutar el algoritmo de aprendizaje débil con el conjunto de ejemplos E y pesos s dados por D_t
   b. Obtener una hipótesis débil $h_t : X$
   c. Actualizar los pesos para todos los ejemplos.

$$D_{t+1}(i) = \frac{D_t(i)e^{-\alpha_t y_i h_t(x_i)}}{Z_t}$$

   Donde $\alpha_t \in R$ es el peso elegido y Z_t es un valor de normalización para que D_{t+1} sea un distribución.

   α_t debe ser elegido de modo de minimizar Z_t
3. Devolver la hipótesis final generada a partir de las hipótesis de las rondas de la 1 a la T.

$$H(x) = sign(f(x)) = sign(\sum_{t=1}^{T} \alpha_t h_t(x))$$

Figura 3.1: Algoritmo de Adaboost

A continuación se mostrará con un ejemplo como trabaja *boosting*.

El conjunto entrenamiento viene dado por los siguientes ejemplos:
[Juan, B-PER]
[estudia, O]
[*boosting*, O]
[., O]

Se asigna igual peso a cada una de las instancias (1/4):
{[Juan, B-PER], 1/4}
{[estudia, O], 1/4}
{[*boosting*, O], 1/4}
{[., O], 1/4}

Con el conjunto entrenamiento así modificado se entrena el primero de los clasificadores débiles h_1. Luego éstos ejemplos son clasificados con h_1, se determinan cuales son los ejemplos bien y mal clasificados y se calcula α_1.

Supongamos ahora que la clasificación dada por h_1 para los datos es: [Juan, B-ORG], [estudia, O], [*boosting*, B-PER], [., O]. Las palabras 'Juan' y 'boosting' han sido mal clasificadas, por lo que se deben modificar los pesos de cada ejemplo de la siguiente forma: aumentar el peso de 'Juan' y '*boosting*', disminuir los pesos de 'estudia' y '.'. El conjunto de entrenamiento con los nuevos pesos va a ser la entrada del clasificador h_2. Se trabaja en forma similar hasta obtener tantos clasificadores como se desee. Finalmente se combinan en un clasificador final mediante votación ponderada teniendo en cuenta los α_i calculados.

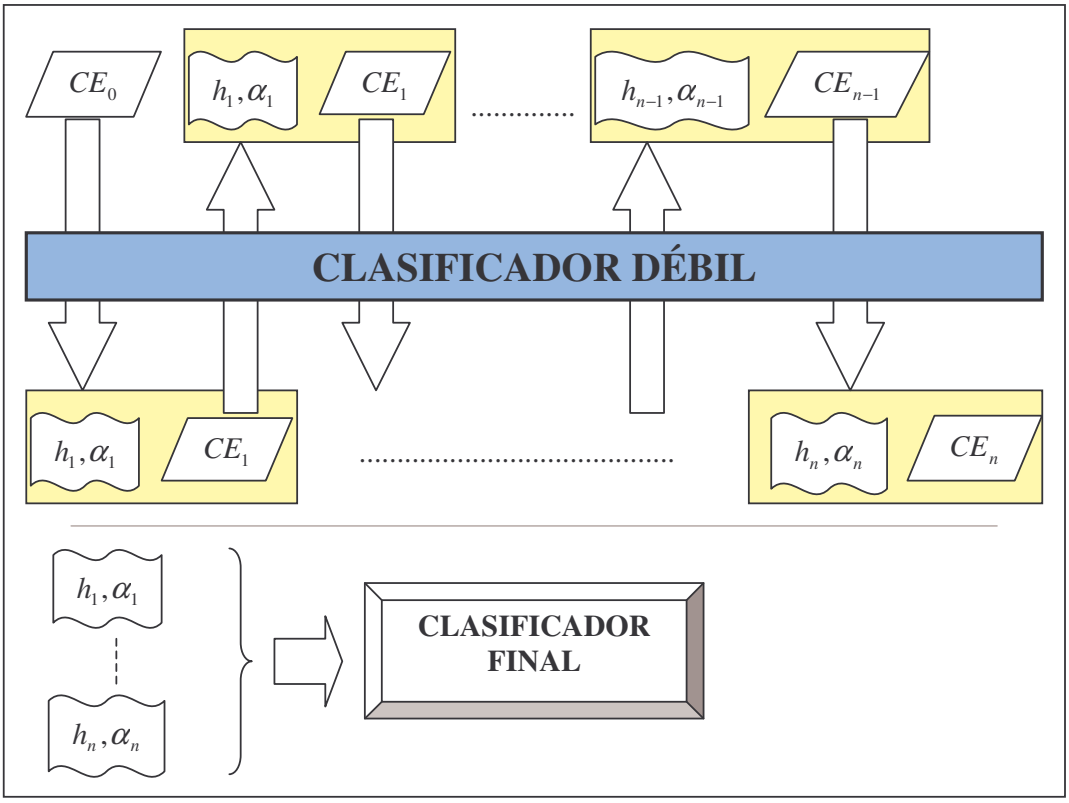

Figura 3.2: Funcionamiento de Adaboost con n rondas. Dado el conjunto entrenamiento, se asignan pesos iguales a cada una de las instancias CE_0. Se entrena el clasificador débil seleccionado, obteniéndose: el clasificador h_1, el conjunto entrenamiento con los pesos modificados CE_1 y el peso de h_1 en el clasificador final α_1. En general, en la ronda i entrenando con CE_{i-1} se van a obtener: el clasificador h_i, su peso en el clasificador final α_i y el conjunto entrenamiento CE_i. Finalmente a partir de los h_i y sus pesos α_i se construye el clasificador final.

3.2 Aplicación de la técnica al reconocimiento de entidades

Para la implementación de este clasificador se utilizó "decision stumps" como algoritmo. Éstos representan un regla que pregunta por un único atributo (ver ejemplo en figura 3.3). Tanto para la implementación de *boosting*, como para la de "decision stumps" se utilizaron las bibliotecas provistas por el Proyecto Weka.

if esDigito(<palabra>) devolver SI, sino devolver NO

Figura 3.3: Regla representando un 'decision stump'. Verifica sí una palabra es un dígito, devolviendo 'SI' si es así y 'NO' en otro caso.

El reconocimiento de las entidades con nombre se realizó en 2 etapas. La primer etapa consistió en reconocer las entidades con nombre indicando si son el comienzo o una palabra intermedia de la entidad con nombre. La segunda etapa consistió en clasificar las entidades reconocidas en la etapa anterior según las categorías establecidas: lugares, personas, organizaciones y misceláneos.

El conjunto de características utilizado en la función de aprendizaje fue básicamente el recopilado del análisis de los trabajos basados en *boosting* presentados en las CoNLL 2002 y 2003. Entre estos se encuentran afijos, pos tags, morfología de la palabra, pertenencia a diccionarios, etc. Esta información se encuentra en detalle en el anexo IV.

Se comenzó con un conjunto pequeño de características al que progresivamente se le agregaron características más complejas o que se creyeran convenientes, a partir del análisis de los resultados obtenidos. Si bien los resultados fueron calculados con el script de evaluación provisto por las CoNLL, también se realizó un análisis manual de los resultados para detectar cuáles eran realmente las carencias del clasificador implementado.

Para tener en cuenta el contexto de las palabras a clasificar se utilizaron ventanas de tamaño 5 (la palabra siendo clasificada, las dos que la preceden y las dos que la siguen).

Tanto para la etapa de reconocimiento como para la de clasificación los resultados obtenidos se encuentran en el anexo III, 'Tablas y resultados'.

Etapa de Reconocimiento

En la etapa de reconocimiento se utilizó una técnica similar a la empleada por el reconocedor de entidades con nombre empleado por Carreras en sus trabajos presentados en las CoNLL 2002 y 2003[1,2]

El reconocedor implementado consistió en anotar las palabras con alguna de las siguientes etiquetas: O si la palabra no corresponde a una entidad con nombre, B_NE si la palabra corresponde al principio de una entidad con nombre, I_NE si la palabra corresponde a palabras internas de una entidad con nombre.

Prueba 1

El conjunto inicial de características empleadas está compuesto por las siguientes: morfología de la palabra, contiene dígitos, contiene puntos, es acrónimo, es inicial, es número romano, contiene guión, símbolo de puntuación, es comilla, es paréntesis, es palabra funcional, largo de la palabra y su pos tag.

La característica morfología de la palabra empleada en la etapa de reconocimiento dependiendo del caso puede tener cuenta los pos tags. En el caso de que todos los caracteres que conforman la palabra sean letras; se evalúa si éstas son: todas mayúsculas, minúsculas ó una mezcla de éstas. Además, si la palabra es comienzo de oración y está formada por letras, estando la primera mayúscula y el resto minúsculas, se verifica sí su pos tag corresponde a un conjunto de pos tags de entidad con nombre. Se analizó el conjunto de datos de entrenamiento conformándose un conjunto con los pos tags que pertenecen a comienzos de oraciones y son entidades con nombre; estos son: NC (nombre común), NP (nombre propio), AQ (adjetivo calificativo), VMI(verbo principal)).

Los resultados obtenidos son los siguientes. Para el reconocimiento de entidades con nombre se obtuvo como resultado general 71,36 para el conjunto de desarrollo (esp.testA) y 73.52 para el conjunto de testeo (esp.testB). Los resultados se muestran en detalle en las tablas 1.1 y 1.2 del anexo III.

Para ambos conjuntos de datos, el porcentaje de recuperación obtenido al reconocer etiquetas de palabras intermedias a una entidad con nombre es muy bajo en comparación con los otros resultados.

Del análisis manual de los textos clasificados, se observó que existen problemas al reconocer entidades con nombre de más de una palabra. Se observa que palabras intermedias de entidades con nombre son etiquetadas como comienzo de nuevas de entidades, un ejemplo de esto se muestra en la figura 3.4.

Palabra	*Etiqueta correcta*	*Etiqueta predicha*
Compañía	B_NE	B_NE
Riograndense	I_NE	I_NE
de	I_NE	I_NE
Telecomunicaciones	*I_NE*	*B_NE*

Figura 3.4: Ejemplo de error de reconocimiento

Prueba 2

Para solucionar el problema detectado en la prueba anterior, se agregó como característica a la función de aprendizaje, la etiqueta de entidad predicha para la palabra anterior a la procesada.

Las ventanas pasan a ser a tener la forma de la que se muestra en la figura 3.5.

Comparando los resultados obtenidos para el reconocimiento de etiquetas de entidad con nombre con los obtenidos en la prueba anterior (figura 3.6) se observa que:

1) El porcentaje de recuperación al reconocer palabras intermedias a una entidad con nombre aumentó notoriamente, más de un 20% en ambos conjuntos de datos. Además se observa un aumento de 6% en la precisión al reconocer comienzos de entidades con nombre. Este último aumento se debe a que muchas de las palabras intermedias a una entidad con nombre antes eran etiquetadas equivocadamente como comienzos de entidad con nombre, bajando así la precisión al etiquetar comienzos de entidades con nombre.

2) La precisión al reconocer palabras intermedias de una entidad con nombre disminuyó más de un 10% en esp.testA y 6% en esp.testB. Sin embargo, la precisión al reconocer principios de entidades con nombre aumentó en alrededor de un 6%. Estos cambios están dados por secuencias de palabras que no son entidades con nombre pero son incorrectamente etiquetados como si lo fueran. Si bien aún continúan siendo incorrectamente etiquetadas, el etiquetado de estas secuencias de palabras es "coherente", contando sólo con una etiqueta de comienzo de entidad con nombre, seguida de varias etiquetas de palabra interna a una entidad con nombre, en lugar de ser una mezcla de etiquetas de comienzo y fin de entidad con nombre. Un ejemplo de esto se encuentra en la figura 3.8.
3) Restringiéndose a los valores de F, estos aumentan tanto para reconocer comienzos como para palabras intermedias a una entidad con nombre, llevando a un resultado general también mejor que en la prueba anterior.

Comparando los resultados de reconocimiento de entidades con nombre (figura 3.7), los resultados mejoran más para el conjunto de datos esp.testB que para esp.testA, aumentando aproximadamente 6 puntos.

car_{w-2}	car_{w-1}	car_{w}	car_{w+1}	car_{w+2}	NE_{w-1}	NE_{w}

Figura 3.5: Ventana de características tenidas en cuenta en la prueba 2.

Prueba 1

esp.testA

	Precisión	Recuperación	F
B_NE	76.19%	90.67%	82.80
I_NE	92.00%	52.18%	66.69
Overall	80.40%	74.31%	77.23

esp.testB

	Precisión	Recuperación	F
B_NE	78.62%	91.65%	84.64
I_NE	92.99%	49.66%	64.74
Overall	82.24%	73.84%	77.82

Prueba 2

esp.testA

	Precisión	Recuperación	F
B_NE	82.59%	83.43%	83.01
I_NE	81.79%	72.48%	76.85
Overall	82.28%	78.78%	80.49

esp.testB

	Precisión	Recuperación	F
B_NE	85.07%	85.86%	85.47
I_NE	86.51%	72.67%	78.99
Overall	85.62%	80.27%	82.86

Figura 3.6: Resultados al reconocer etiquetas de comienzo e intermedias a entidades con nombre para las pruebas 1 y 2.

Prueba 1

	Precisión	Recuperación	F
esp.testA	65.20%	78.81	71.36
esp.testB	67.78%	80.33%	73.52

Prueba 2

	Precisión	Recuperación	F
esp.testA	76.50%	77.25%	76.87
esp.testB	79.73%	80.44%	80.08

Figura 3.7: Resultados del reconocimiento de entidades con nombre para las pruebas 1 y 2.

Palabra	*Etiqueta verdadera*	*Etiqueta que se predecía antes*	*Etiqueta predicha*
subrayó	O	B_NE	B_NE
Hoy	O	I_NE	I_NE
Da	O	B_NE	I_NE
La	O	I_NE	I_NE

Figura 3.8: Ejemplo de errores cometidos en la etapa de reconocimiento

Prueba 3

Una característica utilizada en las CoNLL que mejoró los resultados de todos los trabajos basados en *boosting* que la emplearon, fue la utilización de los afijos de las palabras. Para cada palabra de la ventana se agregaron como características de la función de aprendizaje sus posfijos y prefijos de largo hasta dos caracteres.

Observando los resultados al reconocer etiquetas de entidades (tabla 1.5 anexo III). Para la categoría comienzo de entidad, se observó un leve aumento tanto en la precisión como en la recuperación, aumentando F en 1 punto. Para el caso de las palabras internas a una entidad con nombre, se observó una disminución en la precisión en 1 punto, sin embargo la recuperación aumentó en 4 puntos, aumentado F en casi 2 puntos.

Sin embargo, observando los resultados al reconocer entidades con nombre (tabla 1.6 anexo III) los resultados se mantienen prácticamente incambiados aumentado el resultado general para esp.testA y disminuyendo para esp.testB respectivamente.

Resumiendo, el agregar los afijos de las palabras no queda claro si mejora los resultados, al menos no como era esperado. Teniendo en cuenta que existe un porcentaje de error de reconocimiento que puede ser mayor a la variación de resultados con respecto a la prueba anterior, los afijos no serán removidos del conjunto de características.

Prueba 4

Otra de las características que dio buenos resultados en el único trabajo de las CoNLL 2002 que la empleo (resultando en el trabajo ganador) y luego empleada en todos los trabajos presentados en las CoNLL 2003, fue la utilización de diccionarios.

Se agregó como característica a la función de aprendizaje, si la palabra actual pertenece a alguno de los siguientes diccionarios: de lugares, formado por 1515 palabras correspondientes a países, ciudades, ríos, etc.; de personas, conformado por 1979 palabras correspondientes a nombres propios y apellidos; de organizaciones, conteniendo 1140 palabras correspondientes a organizaciones de las más distintas categorías, como ser clubes de fútbol, marcas de autos, etc.

Estos diccionarios fueron construidos en base a en listas de lugares, personas y organización obtenidos en Internet. También se agregaron a los diccionarios alrededor de un 20% de las entidades con nombre correspondientes a personas, lugares y organizaciones del conjunto de datos de entrenamiento de las CoNLL 2002 para el idioma español. Se intentaron seleccionar las entidades “más comunes”: de las entidades

presentes en el conjunto de entrenamiento se seleccionó un subconjunto con las que se considera más factibles encontrar en cualquier texto a clasificar. Por ejemplo, se prefiere agregar al diccionario de personas la palabra 'Pedro' a agregar 'Hermenegildo'.

La razón por la que se agregaron entidades con nombre del conjunto entrenamiento a los diccionarios es que la fuente de estos textos es la misma de los textos contra los que se evaluara el clasificador. Se espera, entonces, que algunas de las entidades con nombre del primero aparezcan en estos últimos. Cabe destacar que no fueron agregadas todas las entidades con nombre del conjunto entrenamiento (para que el escenario en el que se lleva a cabo el entrenamiento no sea muy distinto al de evaluación). En la etapa de entrenamiento, todas las entidades con nombre serían encontradas en los diccionarios, mientras que en la etapa de reconocimiento de entidades en textos no antes vistos ésta situación es poco probable que ocurra.

Las ventanas pasan a ser de la forma que se muestra en la figura 3.9.

Los resultados obtenidos se encuentran en las tablas 1.7 y 1.8 del anexo III. En ambos conjuntos de datos aumentó la precisión al reconocer entidades con nombre en más de 4 puntos.

car_{w-2}	car_{w-1}	car_w	car_{w+1}	car_{w+2}	dic_w	NE_{w-1}	NE_w

Figura 3.9: Ventana de características tenidas en cuenta en la prueba 4.

En la mayoría de los trabajos presentados en las CoNLL 2002 y 2003, no solamente para los clasificadores basados en *boosting*, se utilizaron ventanas de tamaño 5 o 7. En este trabajo se han utilizado ventanas de tamaño 5. Se experimento, entonces, con el tamaño de la ventana para observar como influye esto en el resultado final y la conveniencia de trabajar con otro tamaño de ventana. Se utilizarán los mismos atributos que en la *prueba 4,* por ser para los cuales se obtuvieron los mejores resultados.

Prueba 5

Se utilizó ventana de tamaño 3 (la palabra considerada, la que la precede y la que la sigue)

Las ventanas pasan a ser de la forma que se muestra en la figura 3.10.

Tanto los resultados de precisión como de recuperación caen abruptamente siendo el valor de F aproximadamente 1 punto, confirmado que el contexto de una palabra es fundamental a la hora de decidir si es o no una entidad con nombre. Queda claro entonces que se necesita un contexto más rico que sólo una palabra a cada lado para obtener resultados aceptables.

car_{w-1}	car_w	car_{w+1}	dic_w	NE_{w-1}	NE_w

Figura 3.10: Ventana de características tenidas en cuenta en la prueba 5.

Prueba 6

Se utilizó una ventana de tamaño 7 (la palabra considerada, las 3 que la preceden y las 3 que la siguen)

Las ventanas pasan a ser de la forma que se muestra en la figura 3.11.

Comparando los resultados con los obtenidos en la prueba 4, aumentó muy poco el resultado para ambos conjuntos de datos (0,5 puntos aproximadamente), siendo mayor el aumento de F para el juego de datos esp.testA. En ambos conjuntos de datos disminuye la precisión y aumenta la recuperación, en 2 y 1 puntos respectivamente.

Dado que el resultado no aumentó en gran medida al aumentar el tamaño de ventana, no se considera de interés seguir aumentándolo.

car_{w-3}	car_{w-2}	car_{w-1}	car_w	car_{w+1}	car_{w+2}	car_{w+3}	dic_w	NE_{w-1}	NE_w

Figura 3.11: Ventana de características tenidas en cuenta en la prueba 6.

	Precisión	Recuperación	F
esp.testA	80.62%	80.05%	80.34
esp.testB	82.47%	81.85%	82.16

Tabla 3.1: Resultados del reconocimiento de entidades con nombre prueba 6.

Ahora que se ha llegado a un valor bueno de F (mayor a 80 puntos), resulta de interés analizar manualmente los errores que se están produciendo en el reconocimiento, para así modificar el conjunto de características con algunas más especificas que ataquen los errores detectados. Dado que el conjunto de datos esp.testA es el conjunto de datos propuesto por las CoNLL para ajustar el clasificador, se examinaron los errores cometidos sobre este conjunto de datos.

Los errores de etiquetado que pueden ocurrir al reconocer entidades con nombre son los siguientes:

1- Etiquetar una palabra como parte de una entidad con nombre cuando no lo es (etiquetar como B_NE o I_NE cuando corresponde O)
2- Etiquetar como “no entidad con nombre” una palabra que sí lo es (etiquetar con O cuando corresponde B_NE o I_NE)
3- Etiquetar como comienzo de entidad con nombre una palabra que es intermedia a ésta y viceversa (etiquetar B_NE cuando corresponde I_NE y etiquetar I_NE cuando corresponde B_NE)

Uno de los problemas detectados es que las palabras disparadoras no están siendo reconocidas como entidades con nombre. Dado que éstas son parte de las entidades, se agregó una característica indicando si es palabra disparadora. Esto produjo una baja en los resultados de reconocimiento de alrededor un punto. Se intento entonces, en lugar de agregar ésta característica, modificar el atributo ‘pertenece al diccionario’ para que considerara las palabras disparadoras. En este caso los resultados también disminuyeron. Quizás el problema viene dado por la calidad del diccionario de palabras disparadoras empleadas. El problema también podría ser que todas las palabras de más

de dos letras en minúsculas se clasifican como no pertenecientes a una entidad con nombre independientemente de las otras características de la ventana.

Un problema fácil de solucionar es el del carácter 'ü'. Éste no estaba siendo considerado en el conjunto de afijos validos ni a la hora de verificar si una palabra esta formada solamente por letras. Se agregó este carácter al conjunto de letras validas, resultando en una correcta clasificación de las palabras que lo contienen (eran dos, una perteneciente a una entidad con nombre y otra no). Con este cambio, estas palabras se clasificaron correctamente, pero no afectó significativamente los resultados globales.

Algunos problemas de etiquetado pueden provenir de la característica que analiza la morfología de la palabra. Se experimentó modificando esta característica, para que en el caso de que una palabra no este formada solamente por letras, tomara distintos valores dependiendo de sí empieza con mayúscula y contiene caracteres que se encontraron en las entidades mal clasificadas como ser guiones, puntos, comas, números, etc. Se probó también con distintas combinaciones de la característica de morfología y eliminando diferentes combinaciones de las características 'es guión', 'es inicial', 'es puntuación' ya que estás característica eran consideradas muchas veces en las distintas pruebas de la característica de morfología. De estas pruebas ninguna logro mejorar los resultados, llevando a una baja de hasta 4 puntos para el conjunto de datos esp.testA.

En general se puede observar que ninguna de las distintas modificaciones realizadas al conjunto de características llevó a resultados considerablemente mejores. De todas formas, el resultado nunca estuvo por debajo del obtenido por la primera prueba realizada, donde se usaba un conjunto de características muy básico.

Otra forma posible de mejorar los resultados es realizando un post procesamiento a los textos clasificados. El problema es que esto eliminaría en cierta medida la automatización del reconocimiento, cayendo en características más especificas del lenguaje. Por lo que se estarían agregando reglas especificas para solucionar los errores de reconocimiento encontrados para el idioma español, siendo prácticamente seguro que estas reglas no servirán a la hora de reconocer textos en otro idioma.

Si bien un par de los trabajos presentados en las CoNLL 2002 realizaron post procesamiento de los textos clasificados, en el caso de este reconocedor se prefirió no hacerlo, para así preservar lo más posible la independencia del lenguaje.

En el anexo IV, 'Errores de reconocimiento en el clasificador basado en *boosting*' se detallan los errores encontrados.

Conclusiones para el reconocimiento

El conjunto de características con el que se obtuvieron los mejores resultados es para los de la prueba 4 utilizando una ventana de tamaño 7 (tabla 3.2).

	Precisión	Recuperación	F
esp.testA	80.62%	80.05%	80.34
esp.testB	82.47%	81.85%	82.16

Tabla 3.2: Resultados de la prueba 6

Al igual que en las CoNLL, se observa que un conjunto grande de características lleva a mejores resultados. Las características que se destacaron en las CoNLL por dar buenos resultados también lo hicieron en el reconocedor implementado: diccionarios, pos tags, morfología de la palabra, etc.

De los trabajos presentados en las CoNLL 2002 (donde se reconocieron entidades con nombre en español), el trabajo que obtuvo mejores resultados fue el de Carreras [3]. El clasificador de Carreras, al igual que este trabajo, separa la etapa de reconocimiento de la de clasificación de entidades con nombre. Además, empleó un conjunto de características muy rico y una ventana de tamaño 7, obteniendo los resultados mostrados en la tabla 3.3.

Conjunto de datos	*Uso diccionarios*	*Precisión*	*recuperación*	*F*
esp.testA	Si	92.45	90.88	91.66
esp.testA	No	77.91	76.59	77.24
esp.testB	No	79.27	79.29	79.28

Tabla 3.3: Resultados obtenidos por Carreras en las CoNLL 2002 en la etapa de reconocimiento de entidades con nombre.

Para el caso en que no se utilizaron diccionarios los resultados obtenidos por el reconocedor aquí presentado son mejores; obteniendo un valor de F de aproximadamente 3 puntos mayor para ambos conjuntos de datos (esp.testA 80.34 y esp.testB 82.16).

Sin embargo, si se comparan los resultados utilizando diccionarios, el clasificador presentado tiene una performance mucho menor. Probablemente, esto se deba a que la calidad de los diccionarios utilizados por Carreras sea mejor.

No es posible comparar los resultados de reconocimiento con los otros trabajos presentados en las CoNLL 2002 basados en *boosting* pues realizan ambas tareas (clasificación y reconocimiento) simultáneamente, sin discriminar en resultados de reconocimiento.

Etapa de Clasificación

Este clasificador se aplica a los resultados obtenidos en la etapa de reconocimiento. La entrada del clasificador va a ser la salida del reconocedor, utilizando el clasificador resultado de la prueba 6. Al igual que en el reconocimiento se utilizó un clasificador basado en *boosting* sobre 'decision stumps'.

Las entidades con nombre reconocidas en la etapa anterior (etiquetadas con B_NE y I_NE) van a ser clasificadas en las categorías LOC, PER, ORG y MISC, conservándose el comienzo de la etiqueta anteriormente predicha (B o I).

Al igual que en el reconocimiento se comenzó con un conjunto de características pequeño el cual fue adaptándose según los resultados obtenidos. En esta etapa, el contexto de las palabras fue dado por una ventana de tamaño 5.

Prueba 1

Se comenzó con una función de aprendizaje con las siguientes características para cada una de las palabras de la ventana.

Un vector de 3 bits, donde el primer bit indica si la palabra pertenece al diccionario de lugares, el segundo indica si pertenece al diccionario de personas y el tercero indica si pertenece al diccionario de organizaciones (los diccionarios los mismos empleados en la etapa de reconocimiento).

También se emplearon los pos tags de las palabras de la ventana, obtenidos mediante Freeling.

Al igual que en el reconocimiento, se utilizó una característica de morfología de la palabra. Ésta indica si los caracteres que conforman la palabra son todas letras, y en caso de que así sea, si estas son todas mayúsculas, todas minúsculas o si es una mezcla éstas.

Las ventanas a considerar son de la forma que se muestra en la figura 3.12.

En general se obtiene un resultado de 50 puntos aproximadamente, los resultados desglosados por categoría se encuentran en la tabla 1.12 del anexo III. Observando los resultados obtenidos, llama la atención ver el bajo resultado para la categoría MISC, tanto al considerar el valor aislado, como comprándolo con el resultado obtenido para las otras categorías (entre 3 y 4 veces menor al obtenido para la categoría que la sigue). Se observa también que los resultados para el conjunto de datos esp.testB son 7 puntos más altos que para el conjunto esp.testA. Esto era en cierta medida esperable, dado que partimos de resultados de reconocimiento que para el conjunto esp.testB eran mejores que para esp.testA.

car_{w-2}	car_{w-1}	car_w	car_{w+1}	car_{w+2}	NE_w

Figura 3.12: Ventana de características tenidas en cuenta en la prueba 1.

Prueba 2

Al conjunto de atributos anterior se agregó como característica los afijos de largo hasta dos de las palabras de la ventana.

Al igual que en la etapa de reconocimiento, los resultados prácticamente no difieren de la prueba anterior, aumentando tan solo 0.2 puntos.

Realizando un análisis manual de los datos se observan errores de clasificación como el que se muestra en la figura 3.13.

Se tiene que para entidades de más de una palabra, se asignan distintas clasificaciones a las palabras que la componen. Una posible forma de solucionar esto es agregar al conjunto de características la etiqueta de entidad predicha para la palabra que precede a la considerada. El problema de esto es que llevaría a que la primera palabra de la entidad fuera la que decidiese la clasificación de toda la entidad. Igualmente se utiliza

una ventana de tamaño 5, al reconocer la primer palabra de la entidad también se está teniendo en cuenta a las dos que la siguen, por lo que este problema no parecería ser tan grave.

Palabra	Etiqueta correcta	Etiqueta predicha
Antonio	B-PER	B-PER
Viana	I-PER	I-LOC
Baptista	I-PER	I-ORG

Figura 3.13: Ejemplo de error de clasificación. A una misma entidad con nombre se asignan categorías distintas.

Prueba 3

Se agregó como característica la etiqueta de entidad predicho para la palabra que precede a la actualmente procesada.

Las ventanas a considerar pasan a ser de la forma que se muestra en la figura 3.14.

Los resultados mejoran notoriamente (tabla 1.14 del anexo III), aumenta tanto la precisión como la recuperación para todas las categorías, aumentando F en casi 10 puntos para ambos conjuntos de datos. Los resultados mejoran en mayor medida para la categoría PER y en menor medida para LOC. La categoría MISC continua siendo la más difícil de clasificar, estando los resultados muy alejados de la categoría que la sigue.

car_{w-2}	car_{w-1}	car_{w}	car_{w+1}	car_{w+2}	NE_{w-1}	NE_{w}

Figura 3.14: Ventana de características tenidas en cuenta en la prueba 3.

Prueba 4

Una característica que podría mejorar los resultados es saber si las palabras de la ventana son una palabra disparadora. Se agregó un vector de 3 bits para cada una de las palabras de la ventana. En este vector, cada bit indica si la palabra se encuentra en el diccionario de palabras disparadoras de personas, lugares u organizaciones respectivamente.

Los resultados se encuentran en la tabla 1.15 del anexo III. Si bien el resultado mejoró, esto fue en poca medida (1,5 para esp.testA y 0,74 esp.testB). Los resultados para la categoría MISC siguen siendo muy bajos.

Prueba 5

Dados los bajos resultados obtenidos al clasificar la categoría MISC, se conformó un diccionario de palabras que pueden estar contenidas o son una entidad con nombre correspondiente a la categoría MISC. La forma en que se construyó este diccionario es similar a la que se empleó al construir los de las otras categorías.

Se transformo el vector de 3 bits que indicaba si la palabra se encontraba en los diccionarios para las categorías LOC, PER y ORG en un vector de 4 bits. Los tres primeros bits quedan como antes y el último indica si la palabra se encuentra en el diccionario de palabras misceláneas.

Los resultados generales aumentan en gran medida (Tabla 1.16 del anexo III), especialmente para el conjunto de datos esp.testA donde F aumenta en 8 puntos, para esp.testB los resultados aumentan casi 4 puntos.

Cabe destacar, que si bien continúan siendo bajos los resultados al reconocer la categoría MISC, el valor de F para ésta aumenta en 16 puntos para el primer juego de datos y en 11 para el segundo.

Los resultados para el conjunto de datos esp.testA mejoran para todas las categorías aumentando en general 8 puntos.

Los resultados se encuentran en la tabla 3.4. Si bien estos aumentan en general 4 puntos para el conjunto de datos esp.testB, este aumento no se da en todas las categorías. La categoría ORG baja 5 puntos, y PER 1; observándose una baja en la precisión y un aumento en la recuperación. Estos cambios en los resultados de PER y ORG pueden deberse a que muchas de las secuencias de palabras mal clasificadas como entidades eran clasificadas como MISC. Éstas ahora son clasificadas como ORG o PER, explicándose así también la baja en la recuperación y el aumento de precisión en la categoría MISC.

esp.testA				esp.testB			
	Precisión	Recuperación	F		Precisión	Recuperación	F
LOC	62.61%	65.28%	63.92	LOC	73.87%	55.81%	63.58
MISC	57.08%	28.09%	37.65	MISC	51.81%	29.41%	37.52
ORG	66.15%	58.33%	62.00	ORG	54.88%	82.59%	65.94
PER	62.92%	81.91%	71.17	PER	54.88%	82.59%	65.94
Overall	63.67%	63.43%	63.55	Overall	64.61%	64.32%	64.46

Tabla 3.4: Resultados de la clasificación de entidades con nombre prueba 5.

Se debe tener en cuenta, que el clasificador implementado está trabajando sobre un texto donde solamente el 82% de las entidades con nombre fueron reconocidas correctamente. Para ver que tan bueno es el clasificador implementado, se etiquetaron correctamente los conjuntos de datos esp.testA y esp.testB, para luego correr el clasificador implementado sobre ellos.

Como era de esperarse los resultados obtenidos son mejores que los obtenidos sobre los textos anotados por el reconocedor implementado (tabla 3.5). Para la categoría que mejoran más los resultados es MISC (22 puntos). Esto último, nos hace notar que para el reconocedor es la categoría más difícil de detectar, llevando a que sea la categoría más difícil de clasificar.

esp.testA				esp.testB			
	Precisión	Recuperación	F		Precisión	Recuperación	F
LOC	58.90%	76.24%	66.46	LOC	69.30%	67.90%	68.59
MISC	58.67%	55.51%	57.04	MISC	64.29%	55.59%	59.62
ORG	71.28%	66.04%	68.56	ORG	69.96%	72.36%	71.14
PER	70.26%	82.16%	75.75	PER	63.80%	85.85%	73.20
Overall	66.48%	71.80%	69.04	Overall	67.73%	72.18%	69.89

Tabla 3.5: Resultados de la clasificación de entidades con nombre sobre los textos con las entidades con nombre perfectamente reconocidas.

3.3 Conclusiones para el clasificador

Al igual que en el reconocimiento, se observó que un conjunto grande de características lleva a mejores resultados. Se encuentra fundamental el uso de diccionarios y la etiqueta de entidad predicha para la palabra anterior.

Como fue observado en los resultados de ambas CoNLL, la categoría más difícil de clasificar es MISC (no solamente para el idioma español). En el clasificador presentado, se mejoró el resultado para esta categoría utilizando un diccionario de palabras correspondientes a entidades con nombre misceláneas. Se observa además que el principal problema de esta categoría es su reconocimiento.

Los resultados generales superan ampliamente los dados por el sistema base, tanto en precisión como en recuperación. Se destaca el valor de precisión donde el resultado es el doble. Sin embargo, con respecto a los trabajos presentados en las CoNLL 2002, el clasificador construido solo obtiene mejor valor de F general que el trabajo de McNamee[7] (basado en cadenas de Markov). Obteniéndose solamente mejor precisión que el trabajo de McNamee y el sistema base. En cuanto a la recuperación, solamente se supera al sistema base. Comparando en detalle con los resultados obtenidos por McNamee, se supera el resultado para todas las categorías menos PER.

En comparación con los trabajos basados en *boosting* de las CoNLL 2002, los resultados obtenidos son mucho más bajos (difiriéndose del más cercano en 7 puntos). Notar además que, por más que se clasificaran textos con las entidades con nombre perfectamente reconocidas, los resultados subirían solamente un puesto en la tabla de resultados, no superándose ninguno de los trabajos basado en *boosting*. Se concluye, entonces, que las principales carencias del clasificador están dadas por la etapa de clasificación.

A continuación, se hará una comparación detallada con los trabajos presentados basados en *boosting*. El objetivo es descubrir las razones por las cuales éstos clasificadores performan mejor que el presentado. En los anexos I y II se profundiza en los trabajos presentados en las CoNLL.

Articulo	Precisión	Recuperación	$F_{\beta=1}$
Carreras	81,38 %	81,40%	81,39
Florian	78,70 %	79,40%	79,05
Cucerzan	78,19%	74,14%	77,15
Wu	75,85 %	77,38%	76,61
Burguer	71.79%	72.70%	72.25
Tjong Kim Sang	76,00%	75,55%	75,78
Patrick	74.32%	73.52%	73.92
Jansche	74,03%	73.76%	73,98
Malouf	73.93%	73.39%	73.66
Tsukamoto	69,04 %	74,12%	71,49
Black	68,78%	66,24%	67,49
Clasificador presentado	64.61%	64.32%	64.46
McNamee	56,28%	66,51%	60,97
Sistema base	26,27%	56,48%	35,86

Tabla 3.6: Resultados de los trabajos presentados en las CoNLL 2002 para el idioma español
Naranja: Trabajos basados en *boosting*.
Amarillo: Clasificador presentado.

Comparando con Tsakamoto [16] (resultados en la tabla 3.7), si bien él obtiene mejores resultados generales para todas las categorías, éstos no difieren mucho para las categorías MISC y ORG.

Español	Precisión	Respuesta	$F_{\beta=1}$
LOC	70,60%	73,25%	71,08
MISC	41,83%	42,94%	42,38
ORG	68,21%	76,93%	72,31
PER	80,23%	84,49%	82,31
Overall	69,04%	74,12%	71,49

Tabla 3.7: Resultados obtenidos por Tsakamoto para esp.testB

Wu [18] obtiene mejores resultados de precisión y de recuperación para todas las categorías (los resultados se encuentran en la tabla 3.8). El resultado final de él difiere en 12 puntos a favor de él. En la categoría en la que difieren menos los trabajos es ORG (7 puntos) y la que más es PER (18,5 puntos).

Español	Precisión	Respuesta	$F_{\beta=1}$
LOC	79,15%	77,40%	78,26
MISC	55,76%	44,12%	49,26
ORG	74,73%	79,21%	76,91
PER	80,20%	89,25%	84,48
Overall	75,85%	77,38%	76,61

Tabla 3.8: Resultados obtenidos por Wu para esp.testB

Comparando con los resultados de Carreras [3] (tabla 3.9), tanto en precisión como recuperación, el trabajo de él se comporta mejor, difiriéndose en el valor de F 16 puntos. Desglosando por categoría, en la que se asemejan más los resultados es para ORG (10 puntos), mientras que para la cual los resultados son más distintos es para PER (23 puntos).

Cabe destacar que la diferencia de resultados comienza a observarse en la etapa de reconocimiento. En ésta, Carreras obtiene un resultado 11,34 puntos mejor. O sea, se parte de una situación en desventaja a la hora de comparar las clasificaciones. De todas formas, comparando el resultado de Carreras con el del clasificador presentado sobre el conjunto de datos correctamente reconocido, él sigue obteniendo mejores resultados.

Español	Precisión	Respuesta	$F_{\beta=1}$
LOC	86,76%	79,43%	82,47%
MISC	60,19%	54,35%	58,73%
ORG	81,21%	82,43%	81,81%
PER	84,71%	93,47%	88,87%
Overall	81,38%	81,40%	81,39%

Tabla 3.9: Resultados obtenidos por Carreras

Comparando uno a uno los trabajos, no se identifica una característica que pueda ser la determinante de por que los trabajos basados en *boosting* presentados en las CoNLL 2002 den mejores resultados. Los tres trabajos de las CoNLL son distintos entre sí, teniendo a su vez similitudes y diferencias con este trabajo.

Si bien el trabajo de Carreras podría ser él mas parecido, dado que divide la clasificación en dos etapas y utiliza un conjunto de características muy similares, parte de bases muy distintas.

La primera es que cada etapa es llevada a cabo por varios clasificadores binarios, los cuales son posteriormente combinados, en lugar de utilizar clasificadores multiclase. Ahora bien, en las CoNLL 2003 Carreras [1] presenta un trabajo muy similar al que había presentado en el 2002, diferenciándose de este ultimo en que cambia los clasificadores binarios por dos clasificadores multiclase (uno para cada etapa). Esto lleva a pensar que los clasificadores binarios utilizados en el 2002 no eran una buena elección, siendo éste un punto a favor del clasificador presentado.

Otras dos diferencias encontradas con el trabajo de Carreras son: primero la utilización de árboles de decisión de profundidad fija como algoritmo de aprendizaje débil (en lugar de *decisión stumps*); segundo un conjunto características más amplio en la etapa de clasificación. En la etapa de reconocimiento, ambos trabajos utilizan un conjunto rico de características, encontrándose entre éstas: pertenece a diccionarios, pos tags, afijos, palabras disparadoras, morfología de las palabras, es comilla, es número romano, es inicial, es acrónimo, etiquetas predichas por el clasificador/ reconocedor actual, etc. Sin embargo en la etapa de clasificación Carreras además utiliza palabras disparadoras, bolsa de palabras[4], sufijos de largo hasta cuatro caracteres y palabras funcionales.

Finalmente, se encuentra otra gran diferencia: el clasificador de Carreras, toma las secuencias de palabras que según BIO (reconocedor empleado) son una entidad con nombre como unidad. Clasificando ésta en un solo paso. En cambio, en este trabajo se procesa palabra a palabra. Pudiendo, entonces, ser las arriba mencionadas las causas de la gran diferencia entre los resultados de ambos clasificadores.

[4] Una bolsa de palabras consiste en las palabras de la ventana no es teniendo en cuenta el orden

En cuanto a los trabajos presentados por Wu y Tsakamoto, estos son mas parecidos entre sí que con el de Carreras, trabajando en una sola etapa y utilizando un clasificador multiclase. Además, los primeros utilizan el mismo tamaño de ventana y el mismo algoritmo de aprendizaje débil.

Una diferencia no menor entre el trabajo aquí presentado y los de Wu y Tsakamoto, es que ambos realizan el reconocimiento y la clasificación en una sola etapa. Llevando a cometer errores en una sola tarea. Comparando los resultados de estos dos contra los del clasificador presentado sobre los datos sin errores de reconocimiento, los de este último siguen siendo más bajos. Por esto, no resulta correcto atribuir los bajos resultados obtenidos al encadenamiento de errores el realizar la tarea en dos etapas.

Analizando los conjuntos de característica de los tres trabajos. Wu y Tsakamoto se observa que utilizan un conjunto de características pobre. Por lo que se podrían atribuir los mejores resultados de ellos a que ambos utilizan conjuntos de características y ventanas más pequeñas. Esto contradice los resultados obtenidos en las pruebas efectuadas en este trabajo, donde se observó que conjuntos características y contextos más grandes llevan a mejores resultados.

Una última diferencia encontrada con el trabajo de Tsakamoto, es que él utiliza stacking[5] para, mejorar los resultados obtenidos. Sin embargo el trabajo de Wu, siendo muy similar a este, no utiliza stacking y obtiene un resultado 5 puntos mejor.

[5] Stacking consiste en aplicar una secuencia de clasificadores, donde la salida de uno es tomada como entrada del siguiente.

4 Clasificador basado en K-NN

El segundo clasificador se basa en *K Nearest Neightbour*, técnica que fue elegida debido a los buenos resultados obtenidos en las CoNLL y a su fácil comprensión e implementación. K Nearest Neightbour (K-NN) está comprendido en lo que se denominan métodos basados en memoria. Esta categoría incluye a todas aquellas técnicas que se valen de las instancias en su representación original como medio principal de clasificación. Esto significa que la clasificación de una instancia cuya clase es desconocida será determinada en función de otras instancias cuya clase sí se conoce (parte de la experiencia del método de aprendizaje).

Se comenzará por dar una breve reseña sobre que consiste K-NN, luego se explicará la aplicación de esta técnica al reconocimiento de entidades con nombre. Finalmente, se discutirán los resultados del clasificador implementado, y se comparará con otro clasificador presentado en las CoNLL basado en la misma técnica.

4.1 K-NN (K Nearest Neightbour)

Como la sigla lo expresa consiste en buscar los *k* instancias más cercanas (cuya clase se conoce) a una instancia *i* cuya clase no se conoce, y estima, la clase de *i* en base a la clase de las *k* instancias.

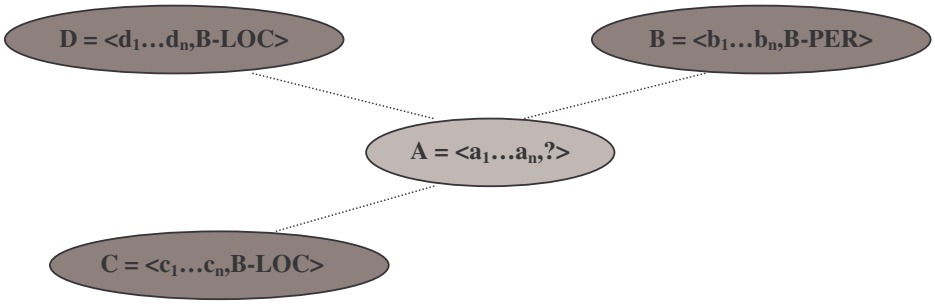

Figura 4.1: *k* instancias mas cercanas a una instancia *i* cuya clase no se conoce (*k* = 3).

Para hallar las instancias más cercanas, es necesario definir antes que nada una función de distancia, la cuál se calcula en base a los atributos de las instancias que se deseen incluir en esta función (exceptuando la clase, por supuesto). Un ejemplo de función puede ser la distancia euclidiana, para la cuál dadas dos instancias $<a_1,\ldots,a_n>$ y $<b_1,\ldots,b_n>$ la función es la siguiente:

$$d(<a_1\ldots a_n>,<b_1\ldots b_n>) = (\Sigma(a_i-b_i)^2)^{1/2}$$

Donde $a_1,..a_n,b_1,..b_n$ denotan los valores tomados por los atributos de cada instancia.

Teniendo las instancias y la función de distancia se seleccionan las *k* instancias cuya distancia a *i* sea la mínima. Algunas variantes de la técnica original establecen que las *k* instancias pueden tener distinto peso al momento de ser tomadas en cuenta para obtener la clase de la instancia *i*. Dentro de estas ponderaciones se pueden distinguir tres (que

son las más comunes), las cuáles se tomaron en cuenta la momento de ajustar el clasificador basado en esta técnica:

- Sin peso: Cualquiera de las *k* instancias pesa lo mismo, por lo cuál la clase de *i* se decide de acuerdo a la clase dominante en las *k* instancias (mayoría simple).
- Peso basado en similaridad: Consiste en ponderar el voto de cada instancia por el coeficiente $w = 1-d$, donde *d* es la distancia a *i*. De esta forma el voto de las instancias más cercanas es el que tiene mayor peso.
- Peso basado en la inversa de la distancia: En esta caso $w = 1/d$, al igual que el punto anterior el voto de las instancias más cercanas tiene mayor peso.

Para los últimos dos casos la clase se determina agrupando las k instancias en base a la clase que poseen, y luego se calcula la suma ponderada (*s*) de acuerdo a *w*; aquella clase cuya suma sea el mayor será la asignada a *i*. Se aclara que en la técnica sin peso podría interpretarse que $w = 1$ (a partir de ahora se usara esta notación para referirnos a esta).

Por ejemplo: se tiene la instancia A cuyo clase no se conoce, $k=3$ y las 3 instancias más cercanas son *B* con $d_B = 0.2$, C con $d_C = 0.4$ y $d_D = 0.5$. Las clases de estas instancias son B-PER, B-LOC y B-LOC respectivamente. Se toma como ponderación el peso basado en similaridad $w = 1/d$ entonces:

El peso del voto para la categoría B-PER esta dada por la instancia *B* entonces

$$w_B = 1/d_B = 1/0.2 = 5$$

El peso del voto para la categoría B-LOC esta dada por la suma de *C* y *D* entonces

$$w_C + w_D = 1/d_C + 1/d_D = 1/0.4 + 1/0.5 = 2.5 + 2 = 4.5$$

Por lo tanto en este ejemplo la clase otorgada a A será B-PER por tener el peso mayor.

Así como las *k* instancias pueden tener peso de acuerdo a su cercanía los atributos también pueden ser ponderados, o sea, ciertos atributos pueden influir mas fuertemente que otros. Un ejemplo de esto se vería reflejado en la siguiente función de distancia:

$$d(<a_1 \ldots a_n>,<b_1 \ldots b_n>) = (\Sigma(w_i a_i - w_i b_i)^2)^{1/2} = (\Sigma(w_i(a_i - b_i))^2)^{1/2} = (\Sigma w_i^2 (a_i - b_i)^2)^{1/2}$$

Donde w_i es el peso asignado al atributo i.

Lamentablemente el proyecto Weka (utilizado para la implementación del clasificador) no considera que los atributos de una instancia puedan tener diferentes ponderaciones. Esto se puede "remediar" permitiendo que haya tantas ocurrencias del atributo como se quiera en una instancia. En este caso la función de distancia sería:

$$d(<a_1 \ldots a_n>,<b_1 \ldots b_n>) = (\Sigma n_i (a_i - b_i)^2)^{1/2}$$

En este caso n_i denota el número de ocurrencias del atributo i.

4.2 Aplicación de la técnica al reconocimiento de entidades

En términos generales este clasificador, al contrario del basado en boosting, funciona en una sola etapa; en otras palabras, reconoce y clasifica al mismo tiempo. Esto tiene como ventaja importante que se pierde menos tiempo en su implementación. Se corre con la desventaja de que se hace una menor distinción de las características que pueden ser importantes para una u otra etapa. Sin embargo, este aspecto también juega a favor, ya que se evita la propagación de errores, propia de los sistemas que funcionan en más de una etapa.

Otro aspecto general a tomar en cuenta es que el conjunto de atributos elegidos permanece incambiado durante todo el proceso de desarrollo del clasificador, aunque sí varía la forma de utilizarlos.

En este clasificador cada instancia esta representada por una ventana de 5 o 7 palabras. Para cada palabra de la ventana las características tomadas en cuenta fueron básicamente las mencionadas en el capítulo dos: Afijos de largo dos, si la palabra contiene dígitos o si la misma es un símbolo de puntuación, pos tags y morfología de la palabra. La morfología es representada por atributos que indican si la palabra comienza con mayúsculas, si todas sus letras son mayúsculas, si sólo algunas son mayúsculas o si son todas minúsculas. A este conjunto también se agrega como parte de la morfología si la palabra se encuentra al comienzo de una oración.

Para las palabras anteriores a la que se quiere clasificar también se incluyen dos atributos: La clase predicha para estas palabras y si ésta existe. Esta última característica indica si la palabra considerada es un "blanco" o no. Dado el siguiente fragmento de texto:

w1 debate NC I-MISC
w2 peninsular AQ I-MISC
w3 . Fp O
w4
w5 Por SP O

Figura 4.2: Fragmento de texto con una línea en blanco.

En algún instante de la clasificación la ventana comprenderá las palabras del fragmento anterior, como se observa, la palabra *w4* esta conformada por una línea en blanco. Como esta palabra es diferente del resto se considera necesario un atributo que lo distinga. El atributo tomará el valor 0 para *w4* (ya que no existe palabra alguna) y 1 para el resto de las palabras del fragmento anterior. Esta característica también es incluida en la representación de las palabras posteriores a la que se quiere clasificar.

A partir de ahora se presentarán los resultados de los tests hechos sobre el clasificador y las consecuentes mejoras que dichos resultados generaron. Se aclara que, a menos que se señale lo contrario, los valores de k y w usados son 1 y 1 respectivamente y el tamaño de la ventana será 5.

Prueba 1

Los resultados obtenidos en una primera etapa para el idioma español se muestran en la siguiente figura:

esp.testA			
	Precisión	Recuperación	F
LOC	53.98%	68.12%	60.23
MISC	39.47%	29.89%	34.02
ORG	62.04%	58.06%	59.98
PER	73.20%	59.00%	65.34
Overall	60.44%	57.72%	59.05

esp.testB			
	Precisión	Recuperación	F
LOC	66.47%	62.92%	64.64
MISC	39.02%	30.29%	34.11
ORG	61.63%	58.50%	60.02
PER	74.27%	72.65%	73.45
Overall	64.05%	60.07%	62.00

Tabla 4.1: Resultados de la prueba 1.

Se puede observar que el clasificador obtiene porcentajes de recuperación similares (alrededor del 60%) para las categorías ORG y PER y para LOC muestra un porcentaje considerablemente más alto (cercano al 70%). Por otra parte el clasificador muestra carencias al momento de detectar entidades que no pertenecen a ninguna de las categorías anteriores (MISC). Una explicación de esto es que, para las primeras tres categorías la palabra que precede a una entidad, puede determinar de forma clara la clasificación de esta. Por ejemplo, cuando una frase se refiere a un lugar, es común encontrar la palabra "en" antes; en el caso de organizaciones es común encontrar un artículo como "el" o "la"; y en el caso de una persona esta palabra podría ser un oficio, por ejemplo "Doctor". Sin embargo, para las entidades misceláneas, no se ve claramente alguna clase de palabra precedente que la distinga. Por ejemplo la frase "en Internet" perfectamente podría provocar que "Internet" sea interpretada como un lugar.

Se procedió entonces a estudiar qué cambio podría implementarse para mejorar estos resultados. Además se observó que se había cometido un pequeño error en la representación de los afijos que eventualmente podría mejorar los resultados: no se tomaban en cuenta aquellos prefijos que incluyeran acentos o la letra *ñ*. En consecuencia la clasificación anterior, estos prefijos tomarían el valor *O* (por defecto).

Prueba 2

Luego de agregar al rango de prefijos aquellos que incluyeran acentos y aquellos que incluyeran a la *ñ*, se ejecutaron los tests nuevamente obteniendo los siguientes resultados:

esp.testA			
	Precisión	Recuperación	F
LOC	55.73%	71.07%	62.47
MISC	37.13%	28.54%	32.27
ORG	61.83%	58.88%	60.32
PER	76.38%	62.44%	68.71
Overall	61.46%	59.54%	60.48

esp.testB			
	Precisión	Recuperación	F
LOC	68.80%	68.36%	68.58
MISC	40.00%	31.18%	35.04
ORG	63.82%	60.86%	62.30
PER	74.80%	75.10%	74.95
Overall	65.92%	63.25%	64.55

Tablas 4.2: Resultados de la prueba 2.

Como se muestra en la tabla 4.2, se logró una mejoría de aproximadamente un 2% en la recuperación general para el esp.testA. Sin embargo si se observa más en detalle, se puede apreciar que para la categoría MISC la recuperación bajo un 1.5%. Esto es consecuencia de tomar en cuenta los acentos y las eñes, dado que estos ocurren con mayor frecuencia en el resto de las categorías, provocando que el clasificador asocie una entidad de clase MISC a otra categoría.

Ajuste de k y w

Luego de desarrollar el clasificador se procede a ajustar los parámetros *k* y *w* con el fin de mejorar el resultado. Para estos ajustes se entrena con un conjunto de menor tamaño (25000 palabras) y se testea sobre un conjunto menor también (10000 palabras), con el fin de acelerar las pruebas.

Se hicieron pruebas variando *k* entre los valores 1, 3, 5, 7 y 9 y *w* entre los valores 1, 1-*d* y 1/*d* sobre los conjuntos que se menciono anteriormente A continuación se muestran los resultados obtenidos para $k = 3$.

$k = 3$			
	Precisión	Recuperación	F
LOC	62.60%	52.74%	57.25
MISC	17.39%	5.19%	8.00
ORG	48.46%	37.08%	42.01
PER	62.29%	48.66%	54.64
Overall	56.01%	43.44%	48.93

Tablas 4.3: Mejor resultado de la pruebas de ajuste de *k* y *w*.

Observando los resultados, se puede establecer una especie de patrón que se cumple a medida que crece *k* (tabla 4.3): Se alcanza el F máximo en 3 y luego comienza a disminuir. Esto se debe a que a la vez que la precisión aumenta la recuperación disminuye. También se observa que en la mayoría de los casos, para un mismo valor de *k*, el valor de *w* prácticamente no influye en la precisión o la recuperación, aunque se señala que los mejores resultados se obtienen para $w \neq 1$. La diferencia poco significativa para un mismo valor de *k* se puede atribuir al hecho de que este no es suficientemente grande como para que *w* influya de forma considerable. Vale la pena señalar que, para casi todas las categorías, la mejor recuperación se obtiene para $k = 1$.

Para corroborar lo observado en los resultados anteriores se entrena al clasificador con el conjunto completo y se testea para $k = 3$.

esp.testA			
	Precisión	Recuperación	F
LOC	55.15%	69.54%	61.52
MISC	36.57%	22.02%	27.49
ORG	63.74%	54.74%	58.90
PER	76.61%	59.54%	67.00
Overall	62.28%	56.09%	59.02

esp.testB			
	Precisión	Recuperación	F
LOC	70.31%	65.31%	67.72
MISC	40.22 %	21.89%	28.35
ORG	63.96%	58.57%	61.15
PER	76.53%	73.20%	74.83
Overall	67.38%	60.16%	63.57

Tablas 4.4: Pruebas con *k*=3 con los conjuntos completos.

Al entrenar y clasificar con los conjuntos completos correspondientes podemos ver que el mejor resultado sigue siendo para $k = 1$, aunque la diferencia (al igual que en las pruebas anteriores) entre los 2 resultados no es significativa.

Ajuste de tamaño de ventana

Las pruebas anteriores (como se menciono al principio) fueron con ventana de tamaño 5, como segundo ajuste realizaron pruebas con ventana de tamaño 7 con conjuntos parciales de entrenamiento y testing, el mejor resultado obtenido es el que se muestra a continuación:

$k = 5, w \neq 1$			
	Precisión	Recuperación	F
LOC	62.93%	44.18%	51.91
MISC	9.52%	2.60%	4.08
ORG	52.97%	32.64%	40.39
PER	63.39%	31.70%	42.26
Overall	56.97%	33.50%	42.19

Tablas 4.5: Mejor resultado con ventana de tamaño 7.

Al igual que en las pruebas anteriores, se observa que F aumenta a medida que crece k, hasta encontrar un máximo ($k = 5$ y $w \neq 1$ en este caso), a partir del cual F comienza a disminuir. También se encuentra el mismo comportamiento para un k fijo y variando w, para el cuál existe un mínimo aumento de F para $w \neq 1$.

Los resultados obtenidos en esta etapa no justifican el uso de una ventana de tamaño 7, dado que muestran ser significativamente inferiores con respecto a los logrados con ventana de tamaño 5.

Pruebas con atributos de mayor peso

Como se mencionó al principio, se cuenta con cierta capacidad de dar mas peso a algunos atributos. En esta prueba se da mas peso al atributo que indica si una palabra comienza con mayúsculas, que se considera predice la presencia de una NE con mayor probabilidad que los demás. En este caso se procede a dar a este atributo el doble de peso que los demás, y así determinar su importancia. La siguiente prueba se realizo con el mismo subconjunto con el que se realizaron los ajustes de k y w.

$k = 1, w = 1$			
	Precisión	Recuperación	F
LOC	56.62%	58.56%	57.58
MISC	12.50%	6.49%	8.55
ORG	42.99%	36.03%	39.20
PER	54.73%	49.11%	51.76
Overall	49.07%	43.44%	46.09

Tablas 4.6: Resultado de prueba con mayor peso en atributo.

En comparación con la prueba 2, los resultados para cada categoría variaron levemente, pero el resultado general no. Dado esto y que el resto de los atributos prevén con menor probabilidad la presencia de una entidad, se mantiene la versión con igual peso en todos los atributos.

4.3 Conclusiones para el clasificador

Al igual que el clasificador basado en boosting y los trabajos presentados en las CoNLL, la categoría más difícil de clasificar es MISC, aunque en este caso no se puede distinguir si la carencia se encuentra en el reconocimiento o la clasificación. Otro parecido con boosting es que los resultados generales obtenidos son similares.

Algunos detalles, que pudieron influir de manera positiva en el desempeño de este clasificador fueron tenidos en cuenta pero no implementados. Por ejemplo el uso de diccionarios, los cuáles han demostrado ser de suma importancia para lograr buenos resultados, también el uso de palabras disparadoras y de palabras funcionales.

A continuación se hará una comparación con la solución presentada por Tjong Kim Sang [15], con la finalidad de remarcar que aspectos coinciden y cuáles difieren. Sobre todo hacer hincapié en estos últimos, que probablemente son la razón por la que los resultados difieren de manera importante; como se muestra en la tabla 4.7 el trabajo de Tjong Kim Sang obtuvo un resultado superior en 11 puntos con respecto al obtenido por el clasificador presentado.

Español	Precisión	Recuperación	$F_{\beta=1}$
LOC	76.01%	76.01%	76.01
MISC	63.70%	50.59%	56.39
ORG	76.45%	78.36%	77.39
PER	79.57%	81.09%	80.32
Overall	76.00%	75.55%	75.78

Tablas 4.7: Resultados obtenidos por Tjong Kim Sang sobre conjunto esp.testB.

Al igual que en el clasificador implementado, cada palabra es representada por sí misma y por las palabras anteriores y posteriores, con la diferencia de que se utiliza ventanas de tamaño 3 (7 palabras). También se utilizan afijos como parte del conjunto de atributos tomados en cuenta, sin embargo estos son de largo 3. Si bien las diferencias mencionadas son significativas, la más importante es que no se utilizan los pos tags. Esta característica, en teoría, mejora de manera considerable los resultados, por lo tanto se supone que los buenos resultados obtenidos por Tjong Kim Sang radican mas bien en cómo utilizó la técnica K-NN, no en como representaba las instancias.

Articulo	Precisión	Recuperación	$F_{\beta=1}$
Carreras	81,38 %	81,40%	81,39
Florian	78,70 %	79,40%	79,05
Cucerzan	78,19%	74,14%	77,15
Wu	75,85 %	77,38%	76,61
Burguer	71.79%	72.70%	72.25
Tjong Kim Sang	76,00%	75,55%	75,78
Patrick	74.32%	73.52%	73.92
Jansche	74,03%	73.76%	73,98
Malouf	73.93%	73.39%	73.66
Tsukamoto	69,04 %	74,12%	71,49
Black	68,78%	66,24%	67,49
Clasificador presentado	65.92%	63.25%	64.55
McNamee	56,28%	66,51%	60,97
Sistema base	26,27%	56,48%	35,86

Tabla 4.8: Resultados de los trabajos presentados en las CoNLL 2002 para el idioma español. Naranja: Trabajos basados en K-NN. Amarillo: Clasificador presentado.

Aunque los resultados de esta solución son mejores, tienen en común que la categoría PER es la mejor clasificada y MISC la peor (lo cuál podría indicar una característica de la técnica utilizada). También se observa que en el caso de la categoría PER los resultados son similares a los de este clasificador propuesto.

Los resultados muestran una mayor variación en el reconocimiento entre una categoría y otra. Por ejemplo en esta solución los valores de F para PER y MISC son 75 y 35 respectivamente, en cambio en el otro caso son 80 y 56 para las misma categorías.

5 Clasificador basado en Árboles de Decisión

El tercer clasificador implementado se basa en *Árboles de Decisión*, aunque esta técnica no es recomendable para tareas como el reconocimiento, es de rápida implementación. En este caso no se busca un clasificador cuyo resultado sea lo mas fiel posible, sino que se busca un tercer integrante que sea factor de desempate al momento de integrar los resultados de los clasificadores.

El aprendizaje basado en Árboles de Decisión es uno de los métodos de inferencia inductiva más usado. Tiene el objetivo de aproximar funciones discretas y cuenta con la característica de ser robusto ante datos que están sujetos a ruido o errores. Se encuentran entre las implementaciones más conocidas de este método ASSISTANT, C4.5 e ID3. Esta última es la usada para el clasificador que se presentara más adelante.

A continuación se explicará más formalmente esta técnica, luego se mostrará como se aplicó al reconocimiento de entidades y finalmente se hará un análisis de los resultados obtenidos por el clasificador implementado.

5.1 Árboles de Decisión

El aprendizaje basado en esta técnica usada tiene el fin de aproximar funciones discretas. Los árboles de decisión pueden ser interpretados como un como un conjunto de reglas de inferencia (de la forma "si-entonces"). Esta técnica es utilizada en una amplia gama de problemas como pueden ser diagnóstico médico o análisis de riesgos al otorgar créditos a solicitantes.

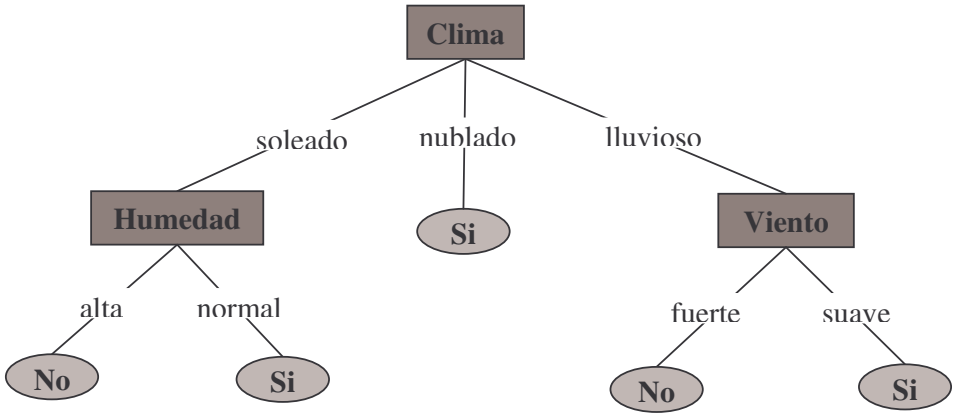

Figura 5.1: Ejemplo de árbol de decisión. Fuente: Machine Learning [8]

Como se muestra en la figura 5.1, cada nodo del árbol representa un atributo y cada rama que desciende de este representa uno de los posibles valores que el atributo puede tomar. Una instancia es clasificada recorriendo el árbol desde la raíz, evaluando el atributo que este nodo representa y descendiendo por la rama que corresponde al valor del atributo para dicha instancia, llegando así a un nuevo nodo. Este proceso se repite para el subárbol que comienza en el nuevo nodo hasta llegar a una hoja donde se encuentra la clasificación que se asignará a la instancia. Ilustraremos esto usando el árbol de la figura cuyo objetivo es evaluar si es recomendable o no jugar al tenis, dada la instancia

<Clima = soleado, Temperatura = caliente, Humedad = alta, Viento = fuerte>

El camino a recorrer de acuerdo a estos valores de atributos corresponde a la rama izquierda del árbol, obteniendo una clasificación negativa.

En general, los árboles de decisión representan una disyunción de conjunciones de condiciones sobre los atributos de las instancias. Cada camino desde la raíz hasta una hoja corresponde a una conjunción de condiciones y el árbol en sí a una disyunción de estas conjunciones. En el caso de la figura anterior la expresión correspondiente sería

(Clima = soleado $\wedge$ Humedad = normal) $\vee$
(Clima = nublado) $\vee$
(Clima = lluvioso $\wedge$ Viento = suave)

A continuación se explica como se construyen de acuerdo a la técnica ID3, la cuál es la implementación del clasificador que se presentará más adelante. Los árboles se forman de arriba hacia abajo decidiendo en cada nodo cuál es el atributo que debe ser evaluado. Para resolver cuál elegir, estos son probados mediante un método estadístico (llamado "Perdida de Información") para determinar que tan bien el atributo clasifica a las instancias del conjunto de entrenamiento por sí solo. El mejor es seleccionado, y se asigna a la raíz del árbol, luego se crea una rama para cada valor posible de este, provocando que el conjunto de entrenamiento sea dividido en tantos conjuntos como valores posibles existan. Este proceso se repite recursivamente para cada rama (usando el nuevo conjunto de entrenamiento asociado a esta), y así determinar el atributo que se asignará como raíz de este sub-árbol.

Un problema de este método es el *sobreajuste*: el árbol de decisión se adapta demasiado al conjunto de entrenamiento, por lo tanto si este no es un fiel representante de todo el universo de instancias, trae como consecuencia una mala clasificación. Para evitar esto, se introduce el concepto de *poda*: se eliminan aquellas ramas del árbol que tienen bajo *soporte*, o sea, las que cubren un porcentaje muy bajo del conjunto de entrenamiento. Al aplicar la técnica de poda, se logra "relajar" mas la función que el árbol representa y así cubrir un conjunto mayor de instancias.

Por último, se define *confianza* como la probabilidad condicional de que dado a se cumple b para la regla $a \rightarrow b$. Como se mencionó anteriormente, un árbol de decisión puede ser visto como un conjunto de reglas de inferencia, donde cada camino del árbol es una regla. Por lo tanto se podría establecer que ninguna regla (camino del árbol) tuviera menor confianza que s, siendo s arbitrario en el intervalo [13,14].

Las pruebas presentadas se realizan variando los parámetros s (confianza), p (poda medida en cantidad de casos mínimos permitidos) y el tamaño de la ventana.

5.2 Aplicación de la técnica al reconocimiento de entidades

Al igual que el clasificador basado en K-NN, funciona en una sola etapa. Además, usa los mismos atributos para representar a una instancia, con la salvedad de que el error cometido para los atributos de tipo "afijo" se permitió intencionalmente. El motivo es que este clasificador tiene altos consumos de memoria RAM (superando los 256 Mb, incluso llegando a requerir más de 512Mb); al reducir el rango de los afijos, se disminuye el tamaño del árbol. Tomando en cuenta que este clasificador se hizo con el fin de desempatar entre los dos primeros, no se consideró grave permitir este error.

A continuación se presentaran las diferentes pruebas realizadas, para las cuáles el clasificador fue entrenado con todo el conjunto de entrenamiento y aplicado todo el conjunto de esp.testA. Al igual que antes, se aclara que, al menos que se señale lo contrario, el tamaño de la ventana es de cinco palabras.

Ajustes de s y p (1)

En primera instancia se procede a realizar pruebas variando *s* entre 50%, 60%, 70% y 80% y *p* entre 10, 15 y 20. Los mejores resultados se obtuvieron para $s = 60\%$ y $p = 10$.

$s = 60\%$, $p = 10$			
	Precisión	Recuperación	F
LOC	51.76%	47.82%	49.71
MISC	32.58%	13.03%	18.62
ORG	60.44%	35.41%	44.66
PER	63.60%	44.76%	52.55
Overall	57.00%	38.56%	46.00

Tabla 5.1: Mejores resultados de pruebas de ajustes de *s* y p (1).

En estas pruebas (ver tabla 5.1 de anexo III), se observa que para una confianza fija, F aumenta a medida que disminuye la poda. Por otra parte si se fija la poda, F aumenta a la vez que aumenta la confianza. Esto sucede probablemente a que la confianza y la poda están relacionados (uno es afectado por el otro), lo que no se ha podido determinar es como las librerías de Weka manejan esta relación.

En general los resultados anteriores, muestran ser mejores para valores bajos de poda, lo cuál se comprobará en la siguiente prueba.

Ajustes de s y p (2)

En base a las conclusiones anteriores, la siguiente prueba se hace variando *p* entre 5 y 7, como se muestra a continuación los mejores resultados se obtienen para *s* = 60 y *p* = 5.

s = 60%, *p* = 5			
	Precisión	Recuperación	F
LOC	52.87%	52.28%	52.58
MISC	32.64%	14.16%	19.75
ORG	59.66%	41.24%	48.77
PER	66.30%	49.43%	56.63
Overall	57.89%	43.27%	49.52

Tabla 5.2: Mejores resultados de pruebas de ajustes de *s* y p (2).

Al igual que en la primer prueba al disminuir *p* para *s* fijo aumenta F, como también ocurre que si se fija *p,* F aumenta o se mantiene constante al aumentar *s*. Como se había supuesto al comienzo de esta prueba, los resultados fueron mejores, aumentando el valor de F general en un 3.5%. Los mejores aumentos para las categorías PER y ORG en un 4%

Ajuste de tamaño de ventana

En este momento lo que se pretende es cambiar el tamaño de la ventana a 7 y observar si los resultados mejoran o no, las pruebas se realizan para los mismos valores de *s* y *p* usados en la prueba anterior.

s = 50%, *p* = 5			
	Precisión	Recuperación	F
LOC	45.30%	32.79%	38.04
MISC	13.42%	4.49%	6.73
ORG	52.64%	25.25%	34.13
PER	64.54%	46.32%	53.93
Overall	52.39%	30.75%	38.75

Tabla 5.3: Mejores resultados de pruebas con ventana de tamaño 7.

Al igual que antes se siguen cumpliendo las mismas condiciones que observamos anteriormente con respecto a *s* y *p*.

El resultado de esta prueba es considerablemente mas bajo (un 10% aprox.) que los resultados obtenidos con una ventana de tamaño 5. Nuevamente; al igual que K-NN, aumentar el tamaño de la ventana empeora el desempeño del clasificador.

5.3 Conclusiones para el clasificador

Se presenta a continuación los resultados de este clasificador sobre el conjunto esp.testB, con los valores de parámetros que dieron mejores resultados sobre esp.testA (s = 60%, p = 5 y ventana de tamaño 5):

s = 60%, p = 5			
	Precisión	Recuperación	F
LOC	60.37%	45.39%	51.82
MISC	31.82%	14.41%	19.84
ORG	58.55%	46.71%	51.97
PER	64.94%	57.96%	61.25
Overall	59.12%	45.55%	51.45

Tabla 5.4: Resultados sobre el conjunto esp.testB con s = 60%, p = 5 y ventana de tamaño 5.

Al no existir en las CoNLL algún sistema cuyo método principal de clasificación sea Árboles de Decisión, no podemos establecer comparación con sistema alguno. De hecho algunos autores intentaron en primer momento usar dicha técnica pero fue descartada debido a los malos resultados que obtuvieron con esta (lo cuál se confirma en este trabajo), lamentablemente estos resultados no fueron publicados para poder comparar con este clasificador.

Comparando con los resultados de las CoNLL, los valores obtenidos para F y recuperación están por encima del sistema base, en el último y penúltimo lugar respectivamente. Sin embargo la respuesta se encuentra por debajo del mínimo (sistema base), ver tabla 5.5.

Queda pendiente investigar más sobre la relación entre s y p y cómo el proyecto Weka maneja ésta; se considera que conocerla mas en profundidad puede haber influido el cómo se hicieron los ajustes de dichos parámetros.

Tampoco se evaluaron exhaustivamente los atributos usados, de forma de determinar cuál era útil y cuál no.

Aunque los resultados son malos con respecto a los clasificadores presentados en las CoNLL, son mejores a los dados por el sistema base. Se considera que el desempeño obtenido es suficiente como para que este clasificador sea factor de desempate entre los dos métodos propuestos anteriormente.

Articulo	**Precisión**	**Recuperación**	$F_{\beta=1}$
Carreras	81,38 %	81,40%	81,39
Florian	78,70 %	79,40%	79,05
Cucerzan	78,19%	74,14%	77,15
Wu	75,85 %	77,38%	76,61
Burguer	71.79%	72.70%	72.25
Tjong Kim Sang	76,00%	75,55%	75,78
Patrick	74.32%	73.52%	73.92
Jansche	74,03%	73.76%	73,98
Malouf	73.93%	73.39%	73.66
Tsukamoto	69,04 %	74,12%	71,49
Black	68,78%	66,24%	67,49
McNamee	56,28%	66,51%	60,97
Clasificador presentado	59.12%	45.55%	51.45
Sistema base	26,27%	56,48%	35,86

Tabla 5.5: Resultados de los trabajos presentados en las CoNLL 2002 para el idioma español
Amarillo: Clasificador presentado.

6 Integración de los clasificadores

Durante el estudio de los trabajos presentados en las CoNLL, se encontró que aquellos trabajos que presentaban mejores resultados, por lo general, no se basaban en una única técnica, sino que combinaban varias. Existen diversas formas de combinar técnicas, destacándose en las CoNLL la utilización de dos de estas: *Stacking* y *Votación*. En este caso para la integración se utiliza votación debido a que mostró tener buenos resultados en las CoNLL.

Inicialmente se detallará el algoritmo usado para integrar los clasificadores basados en *Boosting* y *K-NN*. Luego se explicará el método de votación y finalmente se mostrará como se integraron los clasificadores anteriores junto con el basado en *Árboles de Decisión*.

6.1 Integración de las técnicas K-NN y *Boosting*

El objetivo de esta etapa es obtener un mejor resultado que los obtenidos por los clasificadores por separado, a partir de su combinación. La integración de los clasificadores consiste en tomar la salida de cada uno de estos (en forma de archivos) y formar una única salida.

En una primera etapa, se pretendió implementar e integrar solamente los clasificadores basados en *boosting* y K-NN. La forma de integrarlos consiste, en cierta forma, en elegir la "entrada más coherente" de entre los dos clasificadores. En caso de no poder determinar cual es esta entrada, se elige aquella que pertenezca al clasificador elegido por defecto. El algoritmo integra tantas entidades como sea posible, o sea, si para una palabra determinada el clasificador *a* asigna la etiqueta "O" y el clasificador *b* asigna la etiqueta B-PER, se toma la asignación de este último.

A continuación se presenta (a modo de ejemplo) una tabla con un fragmento de las salidas de los clasificadores y el resultado de la integración, para ilustrar el funcionamiento del algoritmo. En este caso, las entidades son reconocidas de igual forma, aunque difieren en su clasificación: se elige el resultado del clasificador por defecto (*boosting*), ver figura 6.1:

Boosting	*K-NN*	*Integración*
el O	el O	el O
presidente O	presidente O	presidente O
de O	de O	de O
Telefónica B-ORG	Telefónica B-MISC	Telefónica B-ORG
do I-ORG	do I-MISC	do I- ORG
Brasil I-ORG	Brasil I-MISC	Brasil I- ORG

Figura 6.1: Ejemplo de funcionamiento del algoritmo de integración con igual reconocimiento pero diferente clasificación.

Ahora se presenta un ejemplo en el que el reconocimiento es diferente en los dos clasificadores:

Boosting	*K-NN*	*Integración*
, O	, O	, O
Fernando B-PER	Fernando O	Fernando B-PER
Xavier I-PER	Xavier B-PER	Xavier I-PER
Ferreira I-PER	Ferreira I-PER	Ferreira I-PER

Figura 6.2: Ejemplo de funcionamiento del algoritmo con diferente reconocimiento.

Se aclara que el algoritmo mantiene una especie de "estado", dado que conserva la clasificación elegida para la palabra anterior. Al tomar la palabra "Fernando" se tiene dos posibles clasificaciones: O y B-PER. Como el algoritmo integra tantas entidades como sea posible se asigna B-PER, quedando esta clasificación guardada en un buffer.

Luego al encontrar la palabra "Xavier" también tenemos dos clasificaciones posibles, I-PER o B-PER. Como el buffer contiene la etiqueta B-PER, lo más coherente es optar por la clasificación I-PER para esta palabra, quedando ahora en el buffer la etiqueta I-PER. Se presenta un último ejemplo de cómo funciona el algoritmo:

Boosting	*K-NN*	*Integración*
de O	de O	de O
Telefónica B-ORG	Telefónica I-ORG	Telefónica B-ORG
do I-ORG	do I-ORG	do I-ORG
Brasil I-ORG	Brasil I-ORG	Brasil I-ORG

Figura 6.3: Ejemplo de funcionamiento del algoritmo con clasificación errónea en una de las entradas.

En este caso para *K-NN* la entidad comienza con la etiqueta I-ORG, por lo tanto se opta por la salida del otro clasificador (*Boosting*).

En la figura 6.4 se plantea una de las primeras contradicciones de este algoritmo, debido a que para el caso que se presenta el algoritmo opta por la etiqueta que comienza con B- pero ¿que garantía tenemos de que la etiqueta que comienza con I- no era la correcta?:

Boosting	*K-NN*	*Integración*
, O	, O	, O
Juan O	Juan O	Juan O
Carlos B-PER	Carlos I-PER	Carlos B-PER
Rodríguez I-PER	Rodríguez I-PER	Rodríguez I-PER

Figura 6.4: Ejemplo de funcionamiento del algoritmo con clasificación errónea en las dos entradas.

En este caso se opta por la etiqueta B-PER para la palabra "Carlos", pero en realidad la correcta es I-PER, ya que la palabra "Juan" está mal clasificada y es la que debería llevar la etiqueta B-PER.

Se realizan pruebas de integración variando el clasificador elegido por defecto (K-NN o *Boosting*); los resultados obtenidos son los siguientes:

esp.testA			
	Precisión	Recuperación	F
LOC	53.36%	75.74%	62.61
MISC	31.85%	32.13%	31.99
ORG	53.91%	62.47%	57.87
PER	63.70%	70.38%	66.87
Overall	54.40%	62.66%	58.24

esp.testB			
	Precisión	Recuperación	F
LOC	67.36%	74.63%	70.81
MISC	28.96%	31.18%	30.03
ORG	59.85%	69.54%	64.34
PER	62.36%	80.68%	70.34
Overall	59.87%	67.29%	63.37

Tabla 6.1: Resultados con K-NN elegido como resultado por defecto

esp.testA			
	Precisión	Recuperación	F
LOC	59.34%	68.02%	63.39
MISC	38.36%	33.71%	35.89
ORG	59.95%	60.24%	60.09
PER	61.34%	86.09%	71.64
Overall	58.59%	64.56%	61.43

esp.testB			
	Precisión	Recuperación	F
LOC	70.98%	60.70%	65.44
MISC	33.53%	32.65%	33.08
ORG	64.23%	70.83%	67.37
PER	55.28%	88.30%	67.99
Overall	60.61%	65.34%	62.88

Tabla 6.2: Resultados con Boosting elegido como resultado por defecto

Aunque los resultados difieren según el clasificador elegido por defecto, estos son inferiores al mejor de los obtenidos por los clasificadores individualmente (*Boosting)*. Incluso para el conjunto esp.testB, el resultado está por debajo de cualquiera de los dos clasificadores. Una causa de esto ha de ser la contradicción indicada en la explicación del algoritmo.

Se hace necesario buscar otro algoritmo que integre de mejor forma estos resultados, o incluir un tercer clasificador en la solución mediante el método de *votación*, por esto es que se desarrolla e integra el clasificador basado en Árboles de Decisión explicado en el capítulo 5.

6.2 Método de Votación

El método de votación es el más usado en las CoNLL como forma de integración, aunque también existen otras técnicas como por ejemplo Stacking: En esta técnica la salida de un clasificador es la entrada de otro. El método de votación es similar al usado por la técnica de K-NN para definir el valor de una instancia basándose en las k instancias más cercanas. Una similitud es que los votos se pueden ponderar, con la salvedad de que no se puede usar la distancia como parámetro que determina *w*. Una forma de determinar *w* (una de las mas usadas) para este tipo de votación, es que se calcule en base a los resultados obtenidos por cada clasificador. Por ejemplo se puede determinar que si *Boosting* presenta mejores resultados que K-NN entonces $w_{Boosting} > w_{K\text{-}NN}$. A continuación se presenta un ejemplo que ilustra lo anterior.

Sean los clasificadores *A*, *B* y *C*, cuyos valores de recuperación *R* son 65%, 38% y 25% respectivamente y cuyas predicciones para la palabra *WD* son B-PER, B-LOC y B-LOC respectivamente. Tomando como valor de ponderación $w = R/100$ se tiene que el peso del voto para la categoría B-PER esta dado por el clasificador *A*:

$$w_A = R_A/100 = 65/100 = 6.5$$

Y el peso del voto para la categoría B-LOC esta dado por la suma de *B* y *C*:

$$w_B + w_C = R_B/100 + R_C/100 = 38/100 + 25/100 = 3.8 + 2.5 = 6.3$$

Por lo tanto, la clasificación otorgada a *WD* será B-PER por tener mayor peso.

Al igual que antes, se usará la notación $w = 1$ para el caso en que la votación no es ponderada.

6.3 Integración de las técnicas mediante votación

Al incluir un tercer clasificador se cuenta con dos ventajas: la primera es que se tiene una tercera entrada, lo cuál enriquece la clasificación de entidades; la segunda es que el número de clasificadores es impar por lo cuál minimizamos el número de empates en la votación. La razón por lo que esto no se hizo en el primer intento de integración es que se contaba con dos clasificadores, entonces ¿qué sentido tiene que voten para determinar el resultado más probable?. En una eventual votación de estos dos clasificadores solo habrían dos resultados posibles: o los dos clasificadores coincidirían en su voto y se le asigna esta clase a la palabra, o los dos diferirían. En este caso habría un empate y no se sabría que clase asignar a la palabra.

Este nuevo algoritmo, si bien varía en el funcionamiento global con respecto al anterior, conserva algunas características: se le puede ingresar un clasificador por defecto (para $w = 1$ por ejemplo), ya que si los resultados de los tres clasificadores difieren (empate) es necesario optar por uno de ellos.

Las siguientes pruebas se hicieron sobre el conjunto esp.testA, variando w entre los siguientes valores:

- w = F, el voto de cada clasificador es ponderado por el valor de F obtenido individualmente para la clasificación predicha
- w = Recuperación, el voto de cada clasificador es ponderado por el valor de Recuperación obtenido individualmente para la clasificación predicha
- w = Precisión, el voto de cada clasificador es ponderado por el valor de Precisión obtenido individualmente para la clasificación predicha
- $w = 1$, la clasificación que domine la votación es la asignada (votación por mayoría)

La siguiente figura ilustra la técnica de votación para $w = 1$:

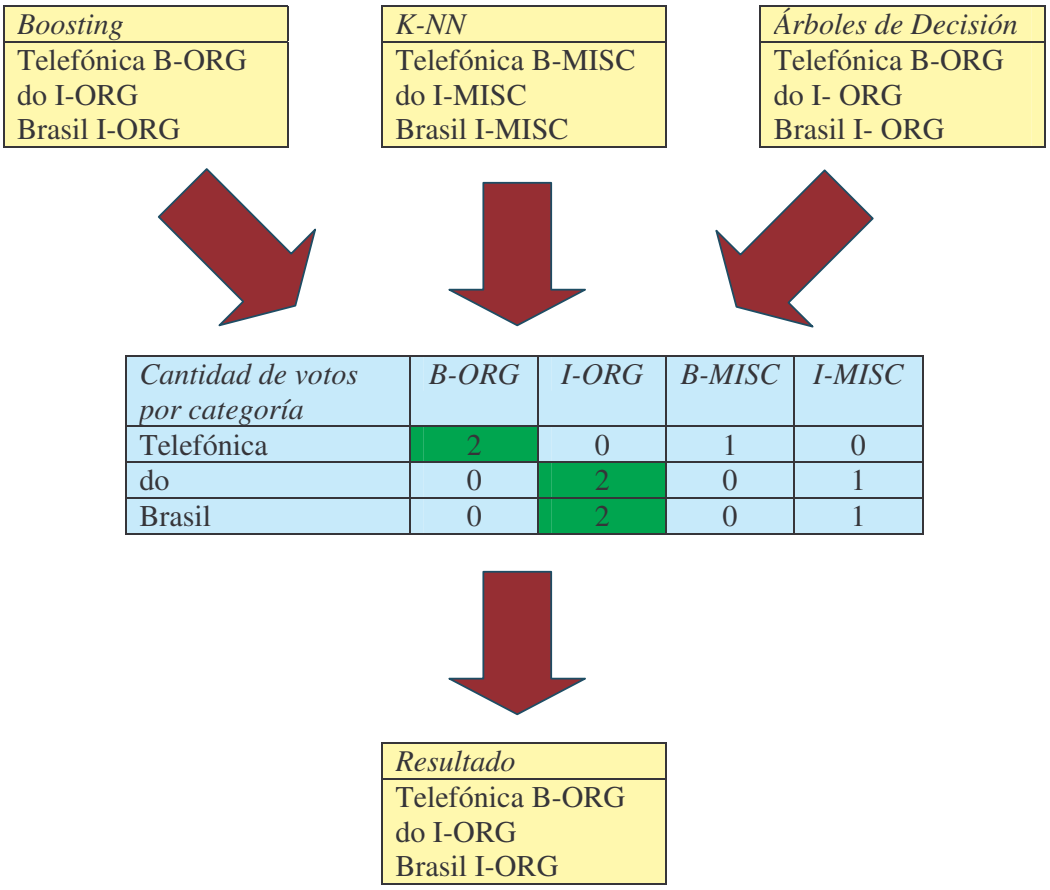

Figura 6.5: Ejemplo de votación para la entidad "Telefónica do Brasil" entre los tres clasificadores implementados

Los valores de F, Recuperación y Precisión que se usaron para establecer *w*, provienen de los resultados individuales de los clasificadores sobre el conjunto esp.testA.

w = F

	Precisión	Recuperación	F
LOC	67.17%	72.28%	69.63
MISC	55.34%	25.62%	35.02
ORG	68.08%	55.86%	61.36
PER	68.84%	79.44%	73.76
Overall	67.46%	63.10%	65.21

w = Recuperación

	Precisión	Recuperación	F
LOC	67.17%	72.28%	69.63
MISC	55.34%	25.62%	35.02
ORG	68.08%	55.86%	61.36
PER	68.84%	79.44%	73.76
Overall	67.46%	63.10%	65.21

w = Precisión

	Precisión	Recuperación	F
LOC	67.45%	72.99%	70.11
MISC	55.34%	25.62%	35.02
ORG	68.08%	55.86%	61.36
PER	69.07%	79.36%	73.86
Overall	67.61%	63.24%	65.35

w = 1

	Precisión	Recuperación	F
LOC	68.15%	71.68%	69.87
MISC	55.30%	26.97%	36.25
ORG	68.19%	56.39%	61.73
PER	69.47%	79.20%	74.01
Overall	67.93%	63.24%	65.50

Tabla 6.3: Resultados de la integración variando *w* sobre el conjunto esp.testA

Como se observa los resultados mejoraron en 2 puntos con respecto a los obtenidos en el primer intento de integración. También se señala que los resultados son similares entre sí, notándose que los dos primeros casos presentan resultados idénticos, lo cuál da entender que pueden estar sucediendo dos cosas. En primer lugar, al observar los valores de F, recuperación y precisión de cada clasificador, se llega a la conclusión de que si dos de ellos coinciden en la clasificación, la suma de sus votos ponderados siempre será mayor al peso del restante. Por lo tanto uno, de los motivos por lo que quizás los dos primeros resultados coinciden es que prácticamente no hay empates, o sea, para casi todos los casos al menos dos clasificadores coinciden en su vaticinio. En segundo lugar las mayorías de empates se dan para la categoría B-PER, ya que es la única categoría para la cuál el mismo clasificador (*Boosting*) domina respecto a F y a recuperación (que son las variables que determinan el peso para los dos primeros casos) al mismo tiempo. Si se observa el resto de las categorías, *Boosting* domina con respecto a F, pero K-NN domina con respecto a recuperación.

Con respecto al tercer resultado, su mejora se puede explicar debido a que el peso es determinado por la precisión. Una de las categorías que presento mejor resultado es LOC, para la cuál K-NN presenta mejor precisión que *Boosting*. En este caso, en un eventual empate, el peso dominante estará dado por la primera técnica.

Sin embargo, el mejor resultado fue obtenido para $w = 1$, o sea, para el método de votación simple eligiendo *Boosting* como clasificador por defecto, debido a que fue el que presentó mejores resultados en general.

A continuación mostramos los resultados obtenidos luego de ejecutar la integración sobre el conjunto esp.testB.

esp.testB			
	Precisión	Recuperación	F
LOC	79.50%	64.02%	70.92
MISC	55.43%	30.18%	39.08
ORG	71.17%	65.79%	68.37
PER	64.16%	83.54%	72.58
Overall	70.47%	65.53%	67.91

Tabla 6.4: Resultados de la integración para w=1

6.4 Conclusiones para el clasificador

Los resultados obtenidos por el clasificador combinado son mejores que los obtenidos por los clasificadores individuales, lo que demuestra la utilidad de combinar varias técnicas.

Mejoraron los resultados para casi todas las categorías, salvo para PER que los resultados son inferiores (aunque la recuperación mejoró). Sin embargo, los resultados generales mejoraron notablemente. El mejor resultado obtenido por los clasificadores simples era de F = 64.55 y el obtenido en el clasificador combinado es de F = 67.91, lo cuál puntúa mejor si se compara con los resultados presentados en las CoNLL 2002 para el idioma español.

Queda pendiente intentar integrar la integración de los clasificadores mediante otro método diferente al de votación: como por ejemplo, Stacking.

Articulo	**Precisión**	**Recuperación**	$F_{\beta=1}$
Carreras	81,38 %	81,40%	81,39
Florian	78,70 %	79,40%	79,05
Cucerzan	78,19%	74,14%	77,15
Wu	75,85 %	77,38%	76,61
Burguer	71.79%	72.70%	72.25
Tjong Kim Sang	76,00%	75,55%	75,78
Patrick	74.32%	73.52%	73.92
Jansche	74,03%	73.76%	73,98
Malouf	73.93%	73.39%	73.66
Tsukamoto	69,04 %	74,12%	71,49
Clasificador integrado	70.47%	65.53%	67.91
Black	68,78%	66,24%	67,49
McNamee	56,28%	66,51%	60,97
Sistema base	26,27%	56,48%	35,86

Tabla 6.5: Resultados de los trabajos presentados en las CoNLL 2002 para el idioma español
Amarillo: Clasificador integrado.

7 Evaluación del clasificador

Con la finalidad de detectar que tan sensible es el clasificador implementado a un cambio la fuente de los textos, se lo empleó para reconocer entidades con nombre en artículos provenientes de un sitio web. También se compararon los resultados con los dados por dos clasificadores basados en reglas.

7.1 Pruebas realizadas sobre un Sitio web

Hasta el momento se entrenó, ajustó y evaluó con los conjuntos de datos provistos por los organizadores de las CoNLL 2002. Ahora bien, se presenta la siguiente duda: ¿qué tan bueno será el clasificador reconociendo artículos de otras fuentes?

El corpus provisto por las CoNLL fue construido basándose en artículos de Reuters [6]. Se analizaran los resultados del clasificador al reconocer entidades con nombre en artículos de un sitio web (el sitio web de Choike[6]). Se conformo un corpus a partir de artículos de éste, teniéndose como requisito que estos fueran de largo mayor a una carilla.

Los artículos de Choike empleados se pasaron al formato de los datos provistos por las CoNLL, se anotaron con pos tags mediante Freeling[8] y se anotaron manualmente con etiquetas de entidad con nombre. El corpus consta de 36 artículos (110000 palabras aproximadamente)

Se evaluaron sobre el corpus de Choike no solo los resultados del clasificador combinado, sino que también se lo hizo con los obtenidos por los distintos clasificadores que lo componen (*boosting*, K-NN y árboles de decisión). Estos fueron evaluados mediante el *script* provisto por los organizadores de las CoNLL.

Los resultados obtenidos fueron más bajos de los esperados, siendo 27 puntos menores a los obtenidos para los archivos de las CoNLL (ver tabla 7.1).

	Precisión	Recuperación	F
LOC	39.79%	72.63%	51.42
MISC	18.21%	10.33%	13.18
ORG	43.49%	52.85%	47.72
PER	17.73%	69.30%	28.24
Overall	33.84%	51.18%	40.74

Tabla 7.1: Resultados de la aplicación del clasificador combinado sobre Choike

[6] www.choike.org

Llama la atención es el bajo rendimiento en la categoría PER. Puede observarse que el problema no es la respuesta (70%), sino la precisión, sugiriendo que hay una considerable cantidad de frases clasificadas como personas que en realidad no lo son (ver figura 7.1)

Palabra	*Etiqueta verdadera*	*Etiqueta predicha*
preparatoria	O	O
de	O	O
El	*B-ORG*	*B-PER*
Cairo	*I-ORG*	*I-PER*
10	O	O
,	O	O
la	O	O
administración	O	O
Bush	B-PER	B-PER
fue	O	O

Figura 7.1: Ejemplo de error de clasificación

Para el resto de las categorías el clasificador se comporta de igual forma que antes (no existe variación en cuál categoría se comporta mejor que otra), logrando resultados para ORG y LOC significativamente más altos que para la categoría MISC.

Se observa que los resultados del clasificador combinado son inferiores al basado en *boosting* (tabla 7.2). Esto en gran medida puede estar dado pues el clasificador basado en K-NN y el basado en árboles estén clasificando incorrectamente las mismas palabras, las cuales son clasificadas correctamente por el clasificador basado en *boosting*, resultando en la instancia de votación el resultado tomado como correcto el dado por la mayoría de clasificadores (o sea el de K-NN y árboles de decisión).

Llama la atención que K-NN obtenga un resultado considerablemente inferior a *boosting*, dado que con los archivos provistos por las CoNLL estos fueron similares. Se piensa que esto se debe a que K-NN es un método basado en memoria: las instancias que conforman este clasificador fueron construidas en base a los textos de las CoNLL y se están clasificando instancias construidas a partir de textos de Choike, pudiendo ser éstas considerablemente distintas. Por lo tanto, en una supuesta clasificación de una instancia *i*, las instancias *j* (cuya clase se conoce) que este método encuentra como cercanas *i* son de una naturaleza considerablemente diferente a esta última. Por otro lado *boosting* trabaja distinto: construye un modelo a partir del conjunto de entrenamiento y luego lo utilizada para clasificar nuevos textos, pudiendo esto ser más flexible y adaptable a nuevos contextos.

Clasificador	Precisión	Recuperación	F
Basado en *boosting*	35.87%	54.67%	43.32
Clasificador combinado	33.84%	51.18%	40.74
Basado en K-NN	32.43%	35.64%	33.96
Basado en Árboles de decisión	5.13%	12.77%	7.32

Tabla 7.2: Resultado obtenidos al clasificar textos de Choike.

7.2 Sistemas basados en reglas

Otra técnica muy utilizada para el reconocimiento de entidades con nombre, son los clasificadores basados en reglas. Éstos consisten en analizar el idioma para el cual se desean reconocer entidades con nombre y posteriormente escribir reglas para determinar que palabras lo son y a qué categoría pertenecen. Estos pueden, además, ayudarse mediante la búsqueda de palabras en diccionarios de lugares, nombres propios, organizaciones, etc.

La principal desventaja de estos sistemas es que son totalmente dependientes del lenguaje, precisándose conocimientos expertos de éste para poder escribir reglas que permitan lograr buenos resultados. Una desventaja adicional es que las reglas escritas para un idioma no tienen porque funcionar bien para otro. Por lo general es necesario en el caso de querer reconocer más de un idioma, rescribir las reglas, lo cual es tedioso y costoso en tiempo.

Otra desventaja adicional, es que la forma de mejorar sus resultados es mediante la escritura de nuevas reglas o aumentando el tamaño de los diccionarios empleados. Por el contrario, en el caso de los clasificadores basados en técnicas de aprendizaje automático, aumentar el tamaño del conjunto entrenamiento puede mejorar los resultados. Otra posibilidad también es agregar nuevas características a la función de aprendizaje.

7.2.1 Comparación con el proyecto 'Identificación y clasificación de nombres propios para el español'

Se comparo el clasificador construido con un proyecto de grado de la Facultad de Ingeniería del año 2001, 'Identificación y clasificación de nombres propios para el español' [10].

Este clasificador trabaja en dos etapas: en la primera se reconocen las entidades con nombre y en la segunda se clasifican las entidades reconocidas en el paso previo. La clasificación se hace en las siguientes categorías: lugares, personas, organizaciones, siglas y otros. Se cuenta con reglas para clasificar palabras en las cuatro primeras, quedando en la categoría 'otros' las entidades que según estas reglas no corresponden a ninguna de las anteriores.

Notar que este clasificador cuenta con la categoría siglas (con la cual no cuenta el clasificador), pudiendo éstas ser personas (J.F.K.), lugares (EEUU) u organizaciones (ONU). Dado este problema se decidió, por un lado comparar los resultados de reconocimiento (tarea para la cual los clasificadores son perfectamente comparables), y por otro lado comprar los resultados de clasificación, sin tener en cuenta las palabras clasificadas como siglas por el clasificador basado en reglas.

Para comparar los resultados, se clasificó el juego de datos esp.testB con el clasificador basado en reglas. Para esta tarea se encontraron varias dificultades. La primera fue que este clasificador procesaba textos de menos de 6kb, siendo el tamaño del texto a clasificar de 609kb. Fue necesario entonces partir cuidadosamente el archivo esp.testB y luego unirlo. Otra dificultad, fue que tanto el formato de entrada como de

salida de este clasificador era muy distinto al de las CoNLL. Para poder emplear el script de evaluación de resultados provisto por ellos, era fundamental transformar los resultados del clasificador basado en reglas al formato de salida de éstas. Se debió entonces realizar un procesamiento manual del texto salida del clasificador basado en reglas. También fue necesaria la remoción de todas las entidades con nombre que correspondían a la categoría siglas.

Comparación de reconocimiento

Los resultados se encuentran en la tabla 7.3. El resultado general obtenido es más alto, difiriendo el resultado en 1,5 puntos. Se observa una mayor precisión y una menor recuperación por parte del clasificador combinado. Se observa además, que el clasificador basado en *boosting* también obtiene resultados apenas más altos que el basado en reglas.

Otras cosas que no deben ser pasadas por alto son: primero, el clasificador basado en *boosting* obtuvo mejores resultados para la etapa de reconocimiento que el sistema combinado; segundo, los resultados obtenidos para el reconocimiento por *boosting* sobre esp.testB sin siglas son mejores que los obtenidos al reconocer el conjunto esp.testB completo. Esto estaría indicando que una de las fallas de este clasificador es al reconocer palabras en mayúsculas y palabras en mayúsculas conteniendo puntos (a grandes rasgos, palabras con estos patrones es lo que el clasificador basado en reglas clasifica como siglas)

Clasificador	Precisión	Recuperación	F
Basado en *boosting*	83.41%	84.87%	84.13
Clasificador combinado	85.58%	80.37%	82.89
Basado en reglas	79.28%	83.65%	81.41
Basado en K-NN	83.98%	78.94%	81.38
Basado en Árboles de decisión	67.53%	51.60%	58.50

Tabla 7.3: Resultado obtenidos para el reconocimiento sobre esp.testB

Comparación de clasificación

Los resultados generales son abrumadoramente mejores para el clasificador combinado (tabla 7.4). Se observa una diferencia de casi 29 puntos en el resultado final. Inclusive para el clasificador basado en árboles de decisión (para el cual los resultados obtenidos son considerados bajos) los resultados obtenidos son mucho mejores que los del basado en reglas en casi 11 puntos.

Los resultados generales para todas las categorías son mejores (tabla 7.5), siendo la menor diferencia en la categoría organización con 17 puntos y difiriendo del resto más de 20 puntos.

Comparando más en detalle los resultados, se observa en cuando a la precisión que el clasificador basado en reglas obtiene mejores resultados al reconocer lugares, organizaciones y personas, dándose la mayor diferencia (26 puntos) al reconocer personas. Al reconocer misceláneos el resultado se revierte en favor del clasificador combinado, existiendo una diferencia de casi 45 puntos. Estas diferencias en los

resultados de precisión vienen dados por la forma en que trabaja el clasificador basado en reglas. Éste es muy acertado al clasificar una entidad como lugar, persona u organización; el problema es que muchas veces se equivoca al no clasificar entidades de esas categorías, dejándolas como misceláneas.

En cuando a la recuperación, los resultados obtenidos por el clasificador basado en reglas son mucho más bajos (inclusive la mitad) para todas las categorías menos para misceláneos. Esto se debe a que no confunde tantas entidades misceláneas con entidades pertenecientes a otras categorías.

Cabe destacar que los resultados obtenidos por los clasificadores individuales y el combinado para la clasificación del conjunto de datos esp.testB sin siglas son mejores que para la clasificación del juego de datos completo.

Clasificador	Precisión	Recuperación	F
Clasificador combinado	69.47%	65.62%	67.49
Basado en *boosting*	64.46%	65.76%	65.10
Basado en K-NN	63.69%	61.84%	62.75
Basado en Árboles de decisión	56.71%	43.33%	49.12
Basado en reglas	38.87%	38.16%	38.51

Tabla 7.4: Resultados obtenidos para la clasificación sobre esp.testB donde no son consideradas las entidades clasificadas como siglas por el sistema basado en reglas.

Clasificador basado en reglas			
	Precisión	Recuperación	F
LOC	81.72%	38.03%	51.90
MISC	9.94%	56.67%	16.92
ORG	76.43%	32.63%	45.73
PER	90.23%	37.89%	53.37
Overall	38.87%	38.16%	38.51

Clasificador implementado			
	Precisión	Recuperación	F
LOC	79.62%	69.46%	74.19
MISC	54.88%	32.26%	40.63
ORG	67.84%	57.20%	62.07
PER	64.45%	84.39%	73.09
Overall	69.47%	65.62%	67.49

Tabla 7.5: Resultados obtenidos para la clasificación sobre esp.testB donde no son consideradas las entidades clasificadas como siglas por el sistema basado en reglas separado por categoría de clasificación

7.2.2 Comparación con el proyecto de grado 'Utilización de descripciones de Web semántica y ontologías en consultas sobre una red de ONG'

Este clasificador fue implementado como parte del proyecto de grado 'Utilización de descripciones de Web semántica y ontologías en consultas sobre una red de ONG' [3] en el cual se requirió reconocer entidades con nombre para la tarea de categorización de información del sitio web de Choike. Este clasificador no solo clasifica entidades con nombre, sino que identifica otro tipo palabras. Las clasificaciones con las que trabaja son: personas, organizaciones, documentos, eventos, cargos, siglas, fechas, ciudades, países y regiones. De éstas categorías, las que importan a la hora de compararlo con este trabajo son las que coinciden con entidades con nombre. Por esto, para la comparación de los clasificadores, se van a tener en cuenta las categorías ciudades, países y regiones que se corresponden a la categoría LOC; la categoría personas a la categoría PER; y organizaciones a la categoría ORG. Las otras clasificaciones no son tenidas en cuenta y tampoco es tenida en cuenta la clasificación MISC. Nuevamente se

observa la existencia de la categoría sigla coincidente con personas, lugares y organizaciones.

Dadas las características del clasificador basado en reglas, lo que se va a comparar son los resultados al reconocer y clasificar entidades con nombre que denotan lugares, personas y organizaciones. No se tendrán en cuenta las entidades que fueron clasificadas como siglas por el clasificador basado en reglas, ni las correspondientes a la categoría misceláneos. Esto llevó, al igual que en el caso anterior, a un tedioso pre procesamiento de los textos.

El clasificador basado en técnicas de aprendizaje automático fue entrenado con los datos de entrenamiento para el idioma español provistos por los organizadores de las CoNLL 2002. El corpus contra el que se corrieron ambos clasificadores son artículos provistos por el grupo de proyecto de grado, este corpus fue armado en base a paginas del sitio web de Choike (aproximadamente 6500 palabras).

Tanto para la tarea de reconocimiento como para la de clasificación, los resultados del clasificador basado en reglas son mejores (tablas 7.6 y 7.7). Esto se explica pues las reglas escritas para el clasificador basado en reglas fueron específicamente hechas para reconocer organizaciones, personas y lugares de los artículos de las páginas web del sitio de Choike. Queda como tarea a futuro utilizar el clasificador basado en reglas sobre los artículos provistos por los organizadores de las CoNLL (esp.testA y esp.testB) o entrenar el clasificador basado en aprendizaje automático con textos provenientes del sitio de Choike.

Comparando la clasificación por categoría (tabla 7.8), el clasificador basado en reglas obtiene mejores resultados para todas, obteniendo valores de precisión mucho más altos y de recuperación también más altos (salvo en la categoría personas)

Al igual que fue observado al reconocer entidades con nombre en artículos del sitio web de Choike (capitulo 7 sección 1), tanto para el reconocimiento como para la clasificación, los resultados obtenidos por *boosting* son mejores que los obtenidos por el clasificador combinado.

Clasificador	Precisión	Recuperación	F
Basado en reglas	90.91%	40.82%	56.34
Clasificador combinado	46.73%	64.13%	54.06
Basado en *boosting*	50.46%	71.30%	59.10
Basado en K-NN	50.63%	54.05%	52.29
Basado en Árboles de decisión	19.91%	41.18%	26.84

Tabla 7.6: Resultado obtenidos para el reconocimiento de organizaciones, lugares y personas que no son consideradas como siglas por el clasificador basado en reglas sobre artículos de Choike

Clasificador	Precisión	Recuperación	F
Basado en reglas	90.15%	39.80%	55.22
Clasificador combinado	34.91%	49.78%	41.04
Basado en *boosting*	41.69%	57.83%	48.45
Basado en K-NN	33.33%	37.39%	35.24
Basado en Árboles de decisión	6.44%	22.06%	9.97

Tabla 7.7: Resultados obtenidos para la clasificación sobre artículos de Choike donde no son consideradas las entidades clasificadas como siglas por el clasificador basado en reglas ni las entidades que corresponden a misceláneos.

Clasificador basado en reglas			
	Precisión	Recuperación	F
LOC	89.09%	72.06%	79.67
ORG	89.23%	48.33%	62.70
PER	100.00%	35.29%	52.17
Overall	90.15%	39.80%	55.22

Clasificador combinado			
	Precisión	Recuperación	F
LOC	48.35%	64.71%	55.35
ORG	45.45%	41.67%	43.48
PER	14.53%	50.00%	22.52
Overall	34.91%	49.78%	41.04

Tabla 7.8: Resultados obtenidos para la clasificación sobre artículos de Choike donde no son consideradas las entidades clasificadas como siglas por el clasificador basado en reglas ni las entidades que corresponden a misceláneos. separado por categoría de clasificación

7.3 Conclusiones

En cuanto la clasificación de textos de otras fuentes, se vio que el clasificador implementado es muy sensible a la forma en la que están escritos los textos. Especialmente se noto que la performance del clasificador más afectado fue la del basado en K-NN. Debiéndose esto a las características inherentes de este clasificador (es un método basado en memoria).

Se sospecha que debido a lo explicado en el párrafo anterior es que el clasificador propuesto por el proyecto 'Utilización de descripciones de Web semántica y ontologías en consultas sobre una red de ONG' obtenga mejores resultados que el implementado.

Ahora bien, comparando los resultados con los del clasificador del proyecto 'Identificación y clasificación de nombres propios para el español' (donde se evaluaron sus resultados con textos de las CoNLL). El clasificador presentado en este trabajo, mostró mejores resultados tanto al reconocer como al clasificar entidades con nombre para el idioma español, destacándose en la última de estas tareas. Se debe tener en cuenta, además, que la mayoría de los clasificadores presentados en las CoNLL 2002 obtuvieron por lejos mejores resultados. Por lo que se concluye que, al menos para la tarea de reconocer y clasificar entidades con nombre en el idioma español, un clasificador basado en aprendizaje automático da mejores resultados que uno basado en reglas.

Dadas las características de un clasificador basado en reglas y de uno basado en técnicas de aprendizaje automático, se puede esperar que para el reconocimiento de

entidades con nombre en otro idioma, el clasificador implementado obtendría mejores resultados que el clasificador basado en reglas con el que fue comparado[7].

.

[7] El clasificador basado en técnicas de aprendizaje debe ser entrenado con un conjunto de datos en el nuevo idioma en el que se deseen reconocer las entidades con nombre.

8 Conclusiones

8.1 Análisis de los resultados

En este trabajo se presentó un clasificador de entidades con nombre basado en técnicas de aprendizaje automático. En su construcción se combinaron tres clasificadores, cada uno de ellos implementado con una técnica distinta; *boosting*, KNN y Árboles de Decisión Además, se probaron distintas formas mediante las cuales de combinarlos para obtener en un único el clasificador final.

Con respecto a los clasificadores individuales, el que obtuvo mejores resultados fue el basado en *boosting*, seguido por el basado en KNN, siendo el basado en Árboles de Decisión el que dio resultados significativamente inferiores a los anteriores. Si se observan los resultados desglosados por categoría, tanto el clasificador basado en *boosting* como el que empleo K-NN tienen resultados muy similares, difiriendo en sus resultados, categoría a categoría, en 2 puntos aproximadamente.

En cuanto al clasificador integrado los mejores resultados se obtienen mediante un esquema de votación por mayoría. En el caso de que los tres clasificadores voten distinto, se toma el resultado del clasificador que haya obtenido mejores resultados individualmente (en este caso *boosting*). La combinación de clasificadores demostró ser una estrategia útil, dando mejores resultados que los clasificadores por separado. Sin embargo esta mejora fue tan sólo de 3 puntos con respecto al clasificador que había dado los mejores resultados. Esto se atribuye a que los distintos clasificadores cometen distintos errores al clasificar las mismas palabras. Lamentablemente, el resultado obtenido es de F=67.91, el cual es inferior al de la mayoría de los trabajos presentados en las CoNLL 2002, tan solo por encima de otros dos trabajos.

A modo de ver que tan bueno es aplicar técnicas de aprendizaje automático para el reconocimiento de entidades con nombre, se compararon los resultados obtenidos con dos clasificadores basados en reglas. Los resultados fueron dispares: el clasificador tuvo una actuación 14% inferior a uno, mientras que los resultados fueron 30% mejores a otro.

Algo que no debe ser pasado por alto, es que, en el primer caso, las reglas del clasificador fueron escritas para reconocer entidades con nombre en artículos del sitio web de Choike, mientras que el clasificador basado en técnicas de aprendizaje fue entrenado con los archivos provistos por las CoNLL. La evaluación se hizo sobre textos extraídos de las páginas web de Choike, con lo cual el clasificador basado en reglas contaba con ventaja. Es comprensible, entonces, la diferencia de 14% en contra de este último.

En el segundo caso, la evaluación fue más justa: ninguno de los clasificadores habían sido creados en base a los textos sobre los cuales se medía su actuación. En estas circunstancias, el clasificador combinado se comportó un 30% mejor que el basado en reglas.

Para observar que tan bueno es el clasificador frente a un cambio en los textos a procesar, se entrenó con los archivos provistos por las CoNLL y se clasificaron artículos

del sitio web de Choike. Si bien en la comparación con uno de los clasificadores basados en reglas se utilizó corpus extraído de Choike, este era pequeño (unas 6000 líneas) como para evaluar el cambio de contexto. Por lo tanto, se construyó un nuevo corpus más grande (unas 110000 líneas), que permitiera una mejor evaluación del clasificador. Los resultados fueron 20 puntos más bajos que los obtenidos sobre los conjuntos de datos de las CoNLL, siendo además muy bajos para la categoría PER. Se concluye que un clasificador basado en técnicas de aprendizaje automático es sensible a los textos con los que es entrenado y al formato en el que éstos son escritos.

Finalmente, se menciona que durante el desarrollo de los clasificadores, las librerías del proyecto Weka demostraron ser de suma utilidad. Proveen una rica variedad de clasificadores y métodos estadísticos. Sin embargo, tiene mucho por mejorar. Por ejemplo, no se procesan de forma adecuada los atributos cuyo rango es discreto y se carece de la capacidad de ponderar de manera diferenciada los atributos de una instancia. Esta última, si bien no fue una restricción, condicionó en cierta forma la implementación del clasificador basado en K-NN. Por otra parte, existen más técnicas de aprendizaje automático que sería interesante integrar a esta librería: Transformation Based Learning, Modelos de Markov, etc.

8.2 Opinión de los autores

Fue de mucho interés profundizar en el campo del aprendizaje automático y en su aplicación en diferentes tareas (en este caso en el reconocimiento de entidades con nombre). Aunque esta área ha alcanzado un nivel de madurez importante, a entender de los autores queda mucho por investigar y desarrollar en lo que respecta a técnicas de aprendizaje. Como se vio en este proyecto, el área de la lingüística ha demostrado ser muy complejo; esto probablemente sea uno de los motivos más fuertes que exigen nuevas técnicas de aprendizaje, o como combinar las ya existentes de manera de obtener resultados óptimos. También se cree que la aplicación de tales técnicas será fundamental en desarrollos futuros, sobre todo en el campo de la inteligencia artificial y en la predicción de comportamiento de usuarios de diversos servicios.

8.3 Posibles mejoras y trabajos a futuro

Se vio que la mayor diferencia existente entre el clasificador basado en *boosting* y el que obtuvo mejores resultados en las CoNLL 2002 (presentado por Carreras, basado en *boosting*). Se observó además que las mayores carencias del primero vienen dadas por la etapa de clasificación y no tanto la de reconocimiento. Una posible mejora a esto podría ser que en lugar de asignar la categoría de la entidad palabra a palabra, tomar la secuencia de palabras que denotan una entidad y clasificarla como que fuera una unidad. Si además se quieren mejorar los resultados de reconocimiento, se podrían mejorar los diccionarios, dado que se sospecha que los mejores resultados de Carreras en esta etapa vienen dados por diccionarios mucho más ricos de los empleados en este trabajo.

Con respecto a K-NN, hubiera sido interesante integrar diccionarios y palabras disparadoras al conjunto de características, lo cuál según el criterio de los autores sería positivo para su desempeño.

Si bien esta establecido que la técnica de árboles de decisión por si sola no es apropiada para el reconocimiento de entidades con nombre, los resultados obtenidos por su aplicación (sobre los archivos provistos por las CoNLL) fueron mejores de lo esperado. Se cree que, si se elige la implementación (IB3, C4.5) y parámetros de ajustes (poda, confianza) correctos este método puede ser un componente útil para asistir a esta tarea. Un ejemplo de esto fue el uso dado en este proyecto, como en el clasificador basado en *boosting* (como algoritmo débil) o desempatando en el caso de la integración de los clasificadores.

Aunque fue interesante investigar y desarrollar una solución para integrar los clasificadores, quedaron por el camino estrategias que merecen ser consideradas. Un ejemplo de esto es la técnica de *Stacking*. También se ve interesante el tener en cuenta la combinación de varias de estas técnicas como método de integración.

Como se observó anteriormente, al momento de evaluar la solución implementada con datos que provenían de fuentes diferentes a las CoNLL los resultados obtenidos fueron bajos. Esto se debe, en parte, al hecho de que estos datos tenían un formato significativamente diferente a los provistos por las CoNLL. Por ejemplo, la existencia de palabras escritas con mayúsculas con el fin de resaltarlas, pero que no indicaban la presencia de una entidad con nombre, llevó a clasificaciones erróneas. Esto deja pendiente el estudio y desarrollo de un componente que desempeñe el pre-procesamiento de datos. Éste puede consistir en tomar datos de diversas fuentes y convertirlos a un formato genérico, cambiar palabras mayúsculas por minúsculas según sea el caso, etc.

REFERENCIAS

[1] X. Carreras, L. Màrquez, y L. Padró, A Simple Named Entity Extractor using AdaBoost. In: Proceedings of CoNLL-2003, Edmonton, Canada, 2003, pp. 152-155.

[2] X. Carreras, I. Chao, L. Padró y M. Padró. FreeLing: An Open-Source Suite of Language Analyzers. Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC'04). Lisbon, Portugal. 2004.

[3] X. Carreras, L. Márques y L. Padró, Named Entity Extraction using AdaBoost In: Proceedings of CoNLL-2002, Taipei, Taiwan, 2002, pp. 167-170.

[4] N. Chiaro y P. Damonte, Utilización de descripciones de Web semántica y ontologías en consultas sobre una red de ONGs. Proyecto de grado Facultad de Ingeniería de la Republica Oriental del Uruguay 2005.

[5] S. Cucerzan y D. Yarowsky, Language independent named entity recognition combining morphological and contextual evidence. In Proceedings of 1999 Joint SIGDAT Conference on EMNLP and VLC, University of Maryland, MD, 1999.

[6] Y. Freund y R. E. Schapire. 1997. A decision.theoretic generalization of on-line learning and an application to boosting. In Journal of Computer and System Sciences, 55(1), pages 119-139

[7] P. McNamee y J. Mayfield, Entity Extraction Without Language-Specific Resources. In: Proceedings of CoNLL-2002, Taipei, Taiwan, 2002, pp. 183-186.

[8] T. M. Mitchel. Machine Learning. Carnegie Mellon University. 1997.

[9] D. D. Palmer y D. S. Day, A Statistical Profile of the Named Entity Task. In Proceedings of Fifth ACL Conference for Applied Natural Language Processing (ANLP-97), Washington D.C., 1997

[10] Reuters online: http://about.reuters.com. Último acceso mayo 2005

[11] A. Rosa y G. Vecino, Identificación y clasificación de nombres propios para el español. Proyecto de grado Facultad de Ingeniería de la Republica Oriental del Uruguay 2001

[12] Sitio web de las conferencias MUC 6:
http://www.cs.nyu.edu/cs/faculty/grishman/muc6.html. Último acceso mayo 2005

[13] E.F Tjong Kim Sang. Introduction to the CoNLL-2002 Shared Task: Language-Independent Named Entity Recognition. CoNLL-2002. 2002.

[14] E.F Tjong Kim Sang y F. De Meulder. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. CoNLL-2003. 2003.

[15] E.F Tjong Kim Sang. Memory-Based Named Entity Recognition. CoNLL-2002. 2002.

[16] K. Tsukamoto, Y. Mitsuishi y M. Sassano, Learning with Multiple Stacking for Named Entity Recognition. In: Proceedings of CoNLL-2002, Taipei, Taiwan, 2002, pp. 191-194.

[17] I. H. Witten y E. Frank, M. Kaufmann. Data Mining: Practical machine learning tools with Java implementations. San Francisco, 2000

[18] D. Wu, G. Ngai, M. Carpuat, J. Larsen y Y. Yang, Boosting for Named Entity Recognition. In: Proceedings of CoNLL-2002, Taipei, Taiwan, 2002, pp. 195-198.

GLOSARIO

Bolsa de palabras: Consiste en las palabras de la ventana no es teniendo en cuenta el orden en que se encuentran.

Categoría: Clase asignada a una instancia, en el contexto de este trabajo B-LOC, I-LOC, B-PER, etc.

Clasificador: Solución que utiliza técnica de aprendizaje automático para reconocer y categorizar entidades con nombre.

Clasificador débil: Clasificador fácil y rápido de construir. Sus resultados no son muy buenos.

Corpus: Conjunto de instancias.

Decision stump: Representan una regla que pregunta por un único atributo.

Entidad con nombre: Expresión lingüística que denota personas, lugares, personas, etc.

F: Función utilizada para evaluar el grado de desempeño correcto obtenido por un clasificador.

NE: Entidad con nombre.

NER: Reconocimiento de entidades con nombre.

NEC: Clasificación de entidades con nombre.

Palabra disparadora: Palabra que puede indicar el comienzo de una entidad con nombre. Un ejemplo de una entidad conteniendo una entidad con nombre es 'doctor Pedro Suar', donde 'doctor' es una palabra disparadora. Otros ejemplos de éstas son: 'presidente', 'hospital', 'hotel', 'ingeniero', etc

Palabra funcional: Palabras en minúsculas que ocurren en una entidad con nombre. Estas son 'y', 'en', 'la', 'de', 'las', 'para' y 'los'.

PoS tag: Notación usada para denotar la característica gramatical de una palabra, adverbio, artículo, etc.

Precisión: Porcentaje de entidades con nombre encontradas por el sistema que son correctas.

Recuperación: Porcentaje de entidades con nombre presentes en el corpus que son encontradas por el sistema.

Stacking: Técnica usada para combinar varios clasificadores de forma que la salida de uno sea la entrada del siguiente.