Extracción de Información con Algoritmos de Clasificación

Por

ALBERTO TÉLLEZ VALERO

*Tesis sometida como requisito parcial para obtener el grado de **Maestro en Ciencias** en la especialidad de **Ciencias Computacionales** en el Instituto Nacional de Astrofísica, Óptica y Electrónica.*

Supervisada por:

DR. MANUEL MONTES Y GÓMEZ
DR. LUIS VILLASEÑOR PINEDA
INAOE

Tonantzintla, Pue.

2005

Resumen

La cantidad excesiva de documentos en lenguaje natural disponibles en formato electrónico hace imposible su análisis. Una solución propuesta a este problema son los sistemas de extracción de información, los cuales permiten estructurar datos relevantes a un dominio específico en los documentos. En otras palabras, la extracción de información convierte el problema de analizar una colección de textos en consultar una base de datos, siendo esto último más rápido de realizar además de hacer más factible encontrar una relación entre los datos. Generalmente, en la construcción de estos sistemas se emplean una variedad de recursos lingüísticos, el resultado son sistemas con un alto costo de portabilidad a nuevos dominios e idiomas. En este trabajo se presenta un enfoque diferente al tradicional para construir sistemas de extracción de información. La arquitectura propuesta se puede definir como un conjunto de componentes que en una primera etapa identifican fragmentos del texto con posibilidad de ser extraídos, y posteriormente deciden cuáles de estos fragmentos son relevantes al dominio. La característica principal de la arquitectura es el escaso uso de recursos lingüísticos, los cuales son reemplazados por métodos de aprendizaje supervisado. El propósito de este enfoque es evitar los problemas de portabilidad que actualmente presentan estos sistemas. Finalmente, para mostrar lo útil de la arquitectura se presenta una aplicación real de ésta, donde la tarea es extraer información sobre desastres naturales a partir de noticias en español publicadas en periódicos de Internet. Esta aplicación permitió demostrar lo útil de la arquitectura, esto por comparar los resultados del sistema contra un trabajo de extracción realizado de forma manual, donde se comprobó que la propuesta puede ser útil para incrementar el conocimiento del caso de estudio.

Contenido

Capítulo 1

Introducción

La actualmente llamada era de la información se caracteriza por una extraordinaria expansión de datos que son generados y acumulados acerca de todo tipo de disciplina humana. Una creciente proporción de éstos se almacenan en forma de bases de datos automatizadas, lo que permite su fácil acceso mediante tecnología informática. Sin embargo, no toda la información almacenada de forma electrónica tiene una estructura definida, tal es el caso de la información textual donde encontramos principalmente secuencias de palabras. Si tomamos en cuenta que grandes volúmenes de esta información considerada "no estructurada" son fácilmente accesibles, e.g. las páginas Web que componen Internet, además de que se estima que el 80% del total de nuestra información es textual [50]. Entonces, herramientas que permitan administrar el potencial de datos contenidos en los textos son necesarias.

Dentro de los esfuerzos surgidos para gestionar la información textual se encuentran principalmente los del área conocida como recuperación de información (referirse a [5] para un estudio). Sin embargo, los resultados hasta ahora obtenidos no son del todo favorables, teniendo que lidiar con dos inconvenientes principales. Primero, no existe un buen método para encontrar toda la información y sólo la información que estamos buscando cuando trabajamos con grandes colecciones de documentos. Segundo, aún si podemos encontrar una fracción de documentos relevantes, frecuentemente éstos son demasiados para ser analizados o incluso leídos, provocando tomar decisiones basándose en información parcial y fragmentada. Una de las soluciones propuestas a este último inconveniente que está teniendo auge y que es el punto central de esta tesis, consiste en transformar la información textual a una

forma estructurada como en el caso de una base de datos, y así obtener las ventajas que proporciona este tipo de sistemas.

Entre las propuestas para lograr la estructuración automática de la información contenida en los textos, se encuentran principalmente las del área de procesamiento del lenguaje natural, quienes han bautizado a la tarea como extracción de información. En esta área, el enfoque que tradicionalmente se sigue para construir los sistemas de extracción de información se fundamenta en tratar de comprender los documentos, ya sea de forma parcial o total, para así lograr contestar a preguntas acerca de su contenido. Por consecuencia, herramientas que lleven a cabo un análisis lingüístico son necesarias, además de que mucho del conocimiento indispensable en el desarrollo se genera manualmente por expertos en la tarea. El resultado son sistemas altamente dependientes de un dominio específico, donde el principal inconveniente de la portabilidad es que mucha de la experiencia adquirida por los expertos no puede ser aplicada directamente a nuevos escenarios.

Las investigaciones en lingüística computacional más recientes indican que métodos empíricos o basados en corpus son los más alentadores para desarrollar sistemas de procesamiento de lenguaje natural robustos y eficientes. Estos métodos automatizan la adquisición de mucho del conocimiento complejo requerido por medio de entrenar sobre una adecuada colección de datos o textos previamente etiquetados (referirse a [30] para un estudio). Dado el éxito de los métodos empíricos, muchas de las investigaciones actuales en la extracción de información siguen el mismo rumbo. Es decir, utilizan métodos empíricos para generar conocimiento que anteriormente los expertos en el dominio producían. Entre los métodos más utilizados se destacan formas de aprendizaje automático, lo que ha resultado útil en la creación de sistemas con una portabilidad entre dominios más eficiente.

Sin embargo, en la tarea de adecuar los textos para ser utilizados por los métodos empíricos, muchos sistemas de extracción de información actuales siguen dependiendo de formas sofisticadas de análisis lingüístico. El resultado es una pobre portabilidad entre idiomas, provocada entre otras cosas porque no todos los lenguajes cuentan con los mismos recursos para realizar un análisis lingüístico de manera

automática, o bien no se encuentran a un mismo nivel de exactitud, siendo el inglés el que mayores resultados presenta. Además, si tomamos en cuenta que para cualquier idioma un completo análisis lingüístico realizado de forma automática es aún imposible, resulta relevante hacer un estudio de cómo crear sistemas de extracción de información con métodos empíricos, pero evitando emplear recursos lingüísticos complejos, i.e., limitarse a utilizar un análisis a nivel de palabras sin tomar en cuenta aspectos sintácticos, semánticos o del discurso. El propósito es obtener sistemas con una mayor portabilidad no sólo entre dominios sino también entre idiomas, además de alcanzar una mayor eficiencia al no utilizar un costoso análisis lingüístico. De lo anterior, las preguntas generales que surgen y que buscamos contestar a lo largo del documento de tesis son*: ¿cómo construir sistemas de extracción de información con métodos empíricos y sin un análisis lingüístico sofisticado?* y *¿qué tan eficaz puede ser un sistema de estas características?*

En la siguiente sección introducimos formalmente la descripción del problema, además de continuar exponiendo la motivación del trabajo. Posteriormente, en la sección 1.2 presentamos la organización de la tesis.

1.1 Descripción del Problema

La cantidad excesiva de documentos en lenguaje natural disponibles en formato electrónico hace imposible su análisis, los sistemas de extracción de información permiten estructurar esa información para un dominio especifico, lo que convierte el problema de analizar una colección de documentos en consultar una base de datos, siendo esto último más rápido de realizar además de hacer más factible encontrar una relación entre los datos. Sin embargo, la construcción de estos sistemas es una tarea costosa en tiempo y recursos tanto computacionales como humanos. Además, si tomamos en cuenta que actualmente se emplean una variedad de recursos lingüísticos para su construcción, el resultado son sistemas con un alto costo de portabilidad a nuevos dominios e idiomas.

Parece ser que la hipótesis actual en el desarrollo de sistemas de extracción de información es que se necesita entender el texto en el mayor grado posible para lograr extraer piezas de información específicas dentro del mismo. En contraste, la pregunta de investigación de la que parte esta tesis es: *¿cómo construir sistemas de extracción de información sin utilizar un análisis lingüístico sofisticado?* En otras palabras, *¿cómo extraer información sin tratar de entender el texto?* La hipótesis que tenemos y que pretendemos probar en el contenido del documento es que para ciertos fragmentos del texto que representan información factual de forma explicita, conocidos como *entidades*, es posible realizar su extracción sin tratar de entender el contenido del documento. Esto es, partimos de la idea que las palabras que rodean a las diferentes entidades (i.e. su *contexto*) presentan patrones léxicos que las caracterizan, llamados *patrones de extracción*, y hacen posible discriminar entre las entidades aquellas que representan información relevante o irrelevante para el estudio en caso. De modo tal que cuando ocurre una entidad no vista anteriormente, por medio de su contexto y los patrones de extracción existentes sea posible decidir si esta entidad representa información que debe ser extraída.

Consecuentemente, al pretender utilizar únicamente un análisis en el ámbito de palabras para extraer la información, dos preguntas necesitan ser contestadas: ¿cómo obtener los patrones de extracción? y ¿cómo detectar las entidades para analizar sus contextos? La respuesta que proponemos a la primera pregunta es utilizar técnicas de aprendizaje automático para así obtener los patrones requeridos, además esto coincide con las tendencias empíricas actuales que permiten al sistema una mayor portabilidad entre dominios. Con respecto a la segunda pregunta, nuestro objetivo es hacer el mínimo uso de recursos lingüísticos y así obtener una mayor portabilidad entre idiomas, por tal motivo la propuesta que tenemos es recurrir a un análisis mediante expresiones regulares para detectar las entidades; de manera que, enfocamos la extracción únicamente a entidades que presentan una forma regular, tal como ocurre con ciertos formatos de fechas, nombres propios y cantidades.

Con el objetivo de mostrar lo útil de la arquitectura planteada, se presenta una aplicación real de la extracción de información para el escenario de noticias en

español que reportan desastres naturales. Su elección se debe a que es un dominio rico en datos como los que deseamos extraer, además del interés de colaborar en la creación de un inventario con información de este tipo para nuestro país. Por lo tanto, una vez establecida la tarea surgen varias preguntas técnicas. En primer lugar las relacionadas a los patrones de extracción, éstas son: ¿cómo adecuar los contextos para ser analizados por los esquemas de aprendizaje automático?, i.e. ¿cuál es la representación y tamaño de contexto que se tienen que utilizar?; además de contestar ¿cuál de los algoritmos de aprendizaje evaluados se adapta mejor a la tarea? Posteriormente, con respecto a las entidades, ¿es suficiente un análisis con expresiones regulares para detectar entidades? Y finalmente, ¿qué posibilidades tiene esta propuesta de contribuir en el conocimiento del caso de estudio?

En el contenido del documento se tratará de dar respuesta mediante nuestros experimentos a las preguntas antes planteadas, además de responder a la otra pregunta general de *¿cómo se comporta esta propuesta en la extracción de información?*, i.e., *¿cuáles son sus alcances y limitaciones?*.

1.2 Organización de la Tesis

El documento está organizado como sigue: en el siguiente capítulo se presentan conceptos básicos para entender el contenido de la tesis, los cuales incluyen principalmente conocimientos de aprendizaje automático relacionados a nuestro problema de investigación y la descripción de los algoritmos de aprendizaje utilizados en el trabajo. Además, se incluye una sección de clasificación automática de textos en la que se abordan conceptos claves de la arquitectura propuesta para la extracción de información. El lector familiarizado con estos conceptos puede omitir este capítulo.

En el capítulo 3 se presenta una revisión del trabajo más relevante y reciente en el área de extracción de información. Se analiza la arquitectura general para el desarrollo de este tipo de sistemas y se hace una revisión de arquitecturas que utilizan alguna forma de aprendizaje automático. El objetivo fue identificar aspectos positivos de diseño, así como los negativos para evitar tales obstáculos.

El capítulo 4 y 5 son la parte más importante de este trabajo de tesis, debido a que en éstos se resume nuestra contribución al conocimiento. El primero de los capítulos exhibe tanto la idea como la arquitectura propuesta para crear sistemas de extracción de información, y se discuten ventajas y desventajas de la misma. En el capítulo 5 se presenta una aplicación real de la arquitectura y se detallan las decisiones de implementación tomadas, además se presentan y analizan los resultados de evaluación obtenidos. En este último capítulo se discute la posibilidad que tiene el sistema de extracción de información resultante, denominado "Topo", de contribuir en el conocimiento del caso de estudio.

Finalmente, en el capítulo 7 resumimos las principales aportaciones de la investigación así como las conclusiones, y discutimos posibles direcciones para trabajo futuro.

Capítulo 2

Conceptos Básicos

En este capítulo introducimos la teoría que fundamenta la propuesta aquí planteada. Se presentan las definiciones formales usadas a través del documento, las cuales pretenden en primer lugar familiarizar al lector con el área de aprendizaje automático y los algoritmos de aprendizaje empleados en la tesis, secciones 2.1 y 2.2 respectivamente. Posteriormente, en la sección 2.3 se describe la clasificación automática de textos, punto clave de la idea que fundamenta la tesis y que es expuesta en el capítulo 4.

2.1 Aprendizaje Automático

El aprendizaje automático es la disciplina que estudia cómo construir sistemas computacionales que mejoren automáticamente mediante la experiencia. En otras palabras, se dice que un programa ha "aprendido" a desarrollar la tarea T si después de proporcionarle la experiencia E el sistema es capaz de desempeñarse razonablemente bien cuando nuevas situaciones de la tarea se presentan. Aquí E es generalmente un conjunto de ejemplos de T, y el desempeño es medido usando una métrica de calidad P. Por lo tanto, un problema de aprendizaje bien definido requiere que T, E y P estén bien especificados (referirse a [31] para una descripción más detallada).

A pesar de que se desconoce cómo lograr que las computadoras aprendan tan bien como las personas, ciertos algoritmos propuestos en el campo han resultado efectivos en varias tareas de aprendizaje. Por ejemplo en la clasificación, parte

fundamental del presente proyecto, de la cual se expone a continuación una definición así como la forma habitual de medir su desempeño.

2.1.1 Clasificación

La *clasificación* puede ser formalizada como la tarea de aproximar una *función objetivo* desconocida $\Phi : I \times C \to \{T, F\}$ (que describe cómo instancias del problema deben ser clasificadas de acuerdo a un experto en el dominio) por medio de una función $\Theta : I \times C \to \{T, F\}$ llamada el *clasificador*, donde $C = \{c_1, \ldots, c_{|C|}\}$ es un conjunto de categorías predefinido, e I es un conjunto de instancias del problema. Comúnmente cada instancia $i_j \in I$ es representada como una lista $A = \langle a_1, a_2, \ldots, a_{|A|} \rangle$ de valores característicos, conocidos como *atributos*, i.e. $i_j = \langle a_{1j}, a_{2j}, \ldots, a_{|A|j} \rangle$. Si $\Phi : i_j \times c_i \to T$, entonces i_j es llamado un *ejemplo positivo* de c_i, mientras si $\Phi : i_j \times c_i \to F$ éste es llamado un *ejemplo negativo* de c_i.

Para generar automáticamente el clasificador de c_i es necesario un proceso inductivo, llamado el *aprendiz*, el cual por observar los atributos de un conjunto de instancias preclasificadas bajo c_i o $\overline{c_i}$, adquiere los atributos que una instancia no vista debe tener para pertenecer a la categoría. Por tal motivo, en la construcción del clasificador se requiere la disponibilidad inicial de una colección Ω de ejemplos tales que el valor de $\Phi(i_j, c_i)$ es conocido para cada $\langle i_j, c_i \rangle \in \Omega \times C$. A la colección usualmente se le llama *conjunto de entrenamiento* (Tr). En resumen, al proceso anterior se le identifica como *aprendizaje supervisado* debido a la dependencia de Tr.

2.1.2 Evaluación de la Clasificación

En la investigación, la efectividad es considerada usualmente el criterio de evaluación más importante gracias a su confiabilidad para comparar diferentes metodologías. A diferencia de la eficiencia, que se deja en segundo término debido a su dependencia

de parámetros volátiles, e.g. diferentes plataformas software y/o hardware. La forma usual de medir la efectividad de un clasificador es por medio de su *exactitud* (α), i.e. el porcentaje de decisiones correctas. Un cálculo para la exactitud sobre la clase c_i puede darse en términos de su tabla de contingencia de la siguiente manera:

$$\alpha_i = \frac{TP_i + TN_i}{TP_i + FP_i + TN_i + FN_i} \tag{2.1}$$

donde FP_i (*falsos positivos*) es el número de instancias de prueba incorrectamente clasificados bajo c_i, TN_i (*verdaderos negativos*) es el número de instancias de prueba correctamente clasificados bajo $\overline{c_i}$, TP_i (*verdaderos positivos*) y FN_i (*falsos negativos*) son definidos consecuentemente (Ver tabla 2.1).

Categoría C_i		**Juicio del experto**	
		SI	NO
Juicio del clasificador	SI	TP_i	FP_i
	NO	FN_i	TN_i

Tabla 2.1: Tabla de contingencia para la clase c_i

Durante la experimentación es usual dividir Ω en los tres conjuntos disjuntos Tr (el *conjunto de entrenamiento*), Va (el *conjunto de validación*) y Te (el *conjunto de prueba*). El conjunto de entrenamiento es el conjunto de instancias observadas cuando el aprendiz construye el clasificador. El conjunto de validación es el conjunto de instancias utilizadas para optimizar parámetros en los clasificadores, o para seleccionar uno en particular. Entonces, el conjunto de prueba es el conjunto de instancias sobre las cuales se evalúa finalmente la efectividad del clasificador. Sin embargo, la partición anterior no es adecuada cuando la cantidad de datos en Ω es limitada; en tal caso, el camino estándar para calcular la exactitud de una técnica de aprendizaje es usar una *validación cruzada con 10 pliegues* (*10FCV*, por sus siglas en inglés) [52]. Esta técnica consiste en dividir aleatoriamente Ω en diez partes,

conservando en cada partición la proporción original de las clases, posteriormente cada parte es mantenida una vez y el esquema de aprendizaje entrena sobre las nueve partes restantes, entonces la exactitud es calculada sobre la parte conservada fuera del proceso de entrenamiento. Así, el proceso es ejecutado un total de diez veces sobre diferentes conjuntos de entrenamiento. Finalmente, los diez estimados de exactitud son promediados para producir una completa estimación de la misma.

Para finalizar la sección, cabe destacar que existen otras medidas para evaluar la exactitud del clasificador, las cuales dependen en gran medida de una aplicación en particular, e.g. la evaluación en la clasificación automática de textos que se describe en la sección 2.3.2.

2.2 Algoritmos de Aprendizaje

Diferentes tipos de aprendices, i.e. el proceso inductivo necesario para generar automáticamente el clasificador, han sido utilizados en la literatura de aprendizaje supervisado, en este trabajo nos enfocamos en cuatro de los que se reportan resultados destacados para trabajar con valores que caracterizan texto, los cuales se describen a continuación.

2.2.1 Naive Bayes

El clasificador *naive Bayes* (*NB*) se considera como parte de los clasificadores probabilísticos, los cuales se basan en la suposición que las cantidades de interés se rigen por distribuciones de probabilidad, y que la decisión óptima puede tomarse por medio de razonar acerca de esas probabilidades junto con los datos observados [31]. En tareas como la clasificación de textos este algoritmo se encuentra entre los más utilizados (ver [20, 28]). En [29] se presenta una guía básica de las diferentes direcciones que investigaciones sobre naive Bayes han tomado, las cuales se caracterizan por modificaciones realizadas al algoritmo. En este trabajo empleamos el naive Bayes tradicional, el cual se describe a continuación.

En este esquema el clasificador es construido usando Tr para estimar la probabilidad de cada clase. Entonces, cuando una nueva instancia i_j es presentada, el clasificador le asigna la categoría $c \in C$ más probable por aplicar la regla:

$$c = \arg\max_{c_i \in C} P\left(c_i \middle| i_j\right) \tag{2.2}$$

utilizando el teorema de Bayes para estimar la probabilidad tenemos

$$c = \arg\max_{c_i \in C} \frac{P\left(i_j \middle| c_i\right) P\left(c_i\right)}{P\left(i_j\right)} \tag{2.3}$$

el denominador en la ecuación anterior no difiere entre categorías y puede omitirse

$$c = \arg\max_{c_i \in C} P\left(i_j \middle| c_i\right) P\left(c_i\right) \tag{2.4}$$

tomando en cuenta que el esquema es llamado "naive" debido al supuesto de independencia entre atributos, i.e. se asume que las características son condicionalmente independientes dadas las clases. Esto simplifica los cálculos produciendo

$$c = \arg\max_{c_i \in C} P\left(c_i\right) \prod_{k=1}^{n} P\left(a_{kj} \middle| c_i\right) \tag{2.5}$$

donde $P(c_i)$ es la fracción de ejemplos en Tr que pertenecen a la clase c_i, y $P(a_{kj}|c_i)$ se calcula de acuerdo al teorema de Bayes.

En resumen, la tarea de aprendizaje en el clasificador naive Bayes consiste en construir una hipótesis por medio de estimar las diferentes probabilidades $P(c_i)$ y $P(a_{kj}|c_i)$ en términos de sus frecuencias sobre Tr. En [1] se presenta una descripción detallada de los cálculos.

2.2.2 C4.5

El esquema *C4.5* fue diseñado como una extensión del algoritmo *ID3* [37], este último forma parte de los clasificadores conocidos como árboles de decisión, los

cuales son árboles donde sus nodos internos son etiquetados como atributos, las ramas salientes de cada nodo representan pruebas para los valores del atributo, y las hojas del árbol identifican a las categorías. Estos algoritmos proporcionan un método práctico para aproximar conceptos y funciones con valores discretos, e.g. en clasificación de textos [9, 10, 20]. A continuación se presenta la descripción del algoritmo ID3 con el objetivo de facilitar la posterior descripción de C4.5 (para más detalles refiérase a [38]).

Para construir el árbol, ID3 usa una aproximación descendente que da preferencia a los árboles pequeños sobre los grandes. El nodo raíz es seleccionado por encontrar el atributo más valioso en el conjunto de entrenamiento, i.e. el que mejor clasifica las instancias; la búsqueda se realiza por medio de una prueba estadística que mide qué tan bien un atributo separa el conjunto de entrenamiento de acuerdo a las clases. Una vez que la raíz es seleccionada, se agrega una rama desde la raíz para cada posible valor del atributo correspondiente, y el conjunto de entrenamiento es ordenado en los nodos apropiados, i.e. cada nodo contiene los ejemplos que cumplen la restricción de la rama anterior. Para seleccionar el atributo más valioso en cada punto del árbol, se repite el proceso entero usando el conjunto de entrenamiento asociado con el nodo. De manera que cuando una nueva instancia necesita ser clasificada, los atributos especificados por los nodos son evaluados iniciando por el nodo raíz, entonces de manera descendente se recorren las ramas del árbol que corresponden a los valores de los atributos en la instancia dada, el proceso se repite hasta que una hoja es alcanzada, y es en este punto donde la etiqueta asociada a la hoja es asignada a la nueva instancia como su categoría.

En la tabla 2.1 se resume el algoritmo ID3, donde la medida tradicional para encontrar el atributo más valioso es la *ganancia en la información* (GI), que mide qué tan bien un atributo dado separa el conjunto de entrenamiento conforme a las clases. En la ecuación 2.6 se presenta la forma de calcular la ganancia en la información del atributo a respecto a Tr .

ID3(Tr , c_i , A)

- Crear un nodo *raíz* para el árbol
- Si todos los ejemplos en Tr son positivos, regresar el árbol con el único nodo *raíz* etiquetado como c_i
- Si todas los ejemplos en Tr son negativos, regresar el árbol con el único nodo *raíz* etiquetado como $\overline{c_i}$
- Si la lista A está vacía, regresar el árbol con el único nodo *raíz* etiquetado como el valor de c_i más frecuente en Tr
- En otro caso comenzar
  - Sea a el atributo en A que mejor clasifica a Tr
  - Etiquetar a la *raíz* como a
  - Para cada posible valor v_i de a (i.e., $v_i \in Valores(a)$)
    - Agregar una rama bajo el nodo *raíz* correspondiente a la prueba $a = v_i$
    - Sea Tr_{v_i} el subconjunto de ejemplos para los que $a = v_i$
    - Si Tr_{v_i} está vació
      - Agregar debajo de la rama un nodo con el valor de c_i más frecuente en Tr como etiqueta
    - De lo contrario
      - ID3(Tr_{v_i} , c_i , $A - \{a\}$)
- Terminar
- Regresar *raíz*

Tabla 2.2: Algoritmo ID3

$$GI(Tr,a) \equiv Entropía(Tr) - \sum_{v_i \in Valores(a)} \frac{\left|Tr_{v_i}\right|}{|Tr|} Entropía\left(Tr_{v_i}\right) \qquad (2.6)$$

donde $Entropía(S)$ se calcula de la siguiente manera:

$$Entropía(S) \equiv \sum_{i=1}^{|C|} -P(c_i)\log_2 P(c_i) \qquad (2.7)$$

Finalmente, una vez introducido ID3 los pasos a seguir en C4.5 son:

1. Separar Ω en conjunto de entrenamiento y conjunto de validación.
2. Construir el árbol de decisión para el *conjunto de entrenamiento* (aplicar ID3).
3. Convertir el árbol en un conjunto de reglas equivalente, donde el número de reglas es igual al número de posibles rutas desde la raíz a los nodos hoja.
4. Podar cada regla eliminando precondiciones que resulten en mejorar la exactitud en el *conjunto de validación*.
5. Ordenar las reglas descendentemente de acuerdo a su exactitud, y usarlas en ese orden para clasificar futuros ejemplos.

2.2.3 k-Vecinos Más Cercanos

k–Vecinos más cercanos (*k-NN*, por sus siglas en inglés) es uno de los métodos de aprendizaje basados en instancias más básicos, pero con resultados aceptables en tareas que involucran el análisis de texto (ver [20, 53, 54]). En resumen, este algoritmo no tiene una fase de entrenamiento fuera de línea, por lo tanto, el principal cálculo se da en línea cuando se localizan los k vecinos más cercanos. La idea en el algoritmo es almacenar el conjunto de entrenamiento, de modo tal que para clasificar una nueva instancia, se busca en los ejemplos almacenados casos similares y se asigna la clase más probable en éstos.

En la tabla 2.3 se resume el algoritmo, aquí una manera común de encontrar los k ejemplos más cercanos a la instancia i_q es por medio de la *distancia Euclidiana*, donde la distancia entre las instancias i_j e i_q es definida por la siguiente ecuación:

$$d\left(i_j, i_q\right) \equiv \sqrt{\sum_{k=1}^{|A|}\left(a_{kj} - a_{kq}\right)^2} \tag{2.8}$$

Entrenamiento:

- Para cada ejemplo en Tr , agregar el ejemplo a la lista *ejemplos_entrenamiento*

Clasificación:

- Dada una instancia de prueba i_q a ser clasificada,
- Sean $i_1,\ldots,i_k$ los k ejemplos de la *lista_entrenamiento* que son más cercanos a i_q
- Regresar

$$c = \arg\max_{c_i \in C} \sum_{j=1}^{k} \delta\left(c_i, c_{i_j}\right)$$

donde $\delta(a,b)=1$ si $a=b$ y $\delta(a,b)=0$ en otro caso.

Tabla 2.3: Algoritmo k-Vecinos más cercanos

2.2.4 Máquinas de Vectores de Soporte

Las *máquinas de vectores de soporte* (*SVM*, por sus siglas en inglés) han mostrado conseguir buen desempeño de generalización sobre una amplia variedad de problemas de clasificación, destacando recientemente en problemas de clasificación de textos (ver [12, 20, 21]), donde se aprecia que SVM tiende a minimizar el error de generalización, i.e. los errores del clasificador sobre nuevas instancias. En términos geométricos, SVM puede ser visto como el intento de encontrar una superficie (σ_i) que separe a los ejemplos positivos de los negativos por el margen más amplio posible. (ver [18] para una descripción detallada del algoritmo).

La búsqueda de σ_i que cumple que la distancia mínima entre él y un ejemplo de entrenamiento sea máxima, se realiza a través de todas las superficies $\sigma_1, \sigma_2, \ldots$ en el espacio $|A|$-dimensional que separan a los ejemplos positivos de los negativos en el conjunto de entrenamiento (conocidas como *superficies de decisión*). Para entender mejor la idea detrás del algoritmo SVM tomaremos el caso en el que los ejemplos positivos y negativos son linealmente separables, por lo tanto las superficies de

decisión son $(|A|-1)$-hiperplanos. Por ejemplo, en el caso de dos dimensiones varias líneas pueden ser tomadas como superficies de decisión (ver figura 2.1), entonces el método SVM elige el elemento medio del conjunto más ancho de líneas paralelas, i.e., desde el conjunto en el que la distancia máxima entre dos de sus elementos es la mayor. Cabe resaltar que la mejor superficie de decisión es determinada únicamente por un conjunto pequeño de ejemplos de entrenamiento, llamados *vectores de soporte* (en la figura 2.1 los cuadros distinguen los vectores de soporte).

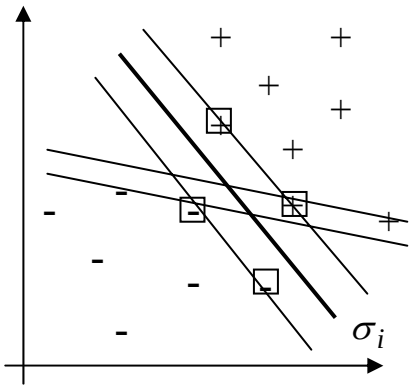


Figura 2.1: Problema de clasificación linealmente separable

Una ventaja importante de SVM es que permite construir clasificadores no lineales, i.e., el algoritmo representa datos de entrenamiento no lineales en un espacio de alta dimensionalidad (llamado el *espacio de características*), y construye el hiperplano que tiene el margen máximo (Ver figura 2.2). Además, debido al uso de una función *kernel* para realizar el mapeo, es posible calcular el hiperplano sin representar explícitamente el espacio de características.

En el presente trabajo se utiliza el método de optimización mínima secuencial (SMO, por sus siglas en inglés) para entrenar el algoritmo SVM. En general, SMO divide la gran cantidad de problemas de programación cuadrática (QP, por sus siglas en inglés) que necesitan ser resueltos en el algoritmo SVM por una serie de problemas QP más pequeños. El lector puede consultar [36] para una descripción detallada del método, la cual se omitió porque no representa el objetivo de la tesis.

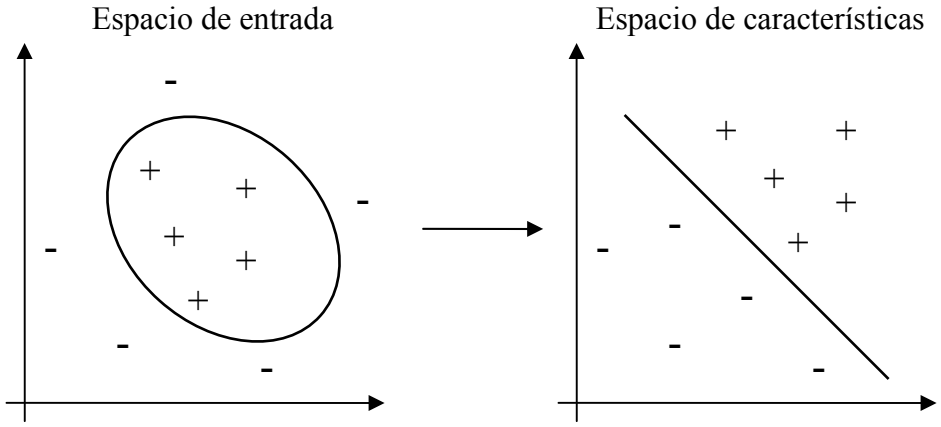


Figura 2.2: Mapeo de datos no lineales a un espacio de mayor dimensionalidad

2.3 Clasificación Automática de Textos

La *clasificación de texto* (también conocida como categorización de texto o ubicación del tema) es la tarea de clasificar automáticamente un conjunto de documentos en categorías (o temas) dentro de un conjunto predefinido [42]. Actualmente la exactitud de muchos de los sistemas de clasificación de texto compite con la de profesionales humanos especializados. Tal avance se debe principalmente a la combinación de tecnologías como son la recuperación de información y el aprendizaje automático, lo cual desde principios de los 90's ha ganado popularidad y eventualmente se ha convertido en el enfoque dominante para construir sistemas de clasificación de textos. La idea básica de este enfoque es que un proceso inductivo automáticamente construya un clasificador por observar las características de un conjunto de documentos previamente clasificados. De tal forma que el problema de clasificación de textos se convierte en una actividad de aprendizaje supervisado como se presentó en la sección 2.1.1.

Sin embargo, un inconveniente inicial es que los textos no pueden ser directamente interpretados en primer lugar por el aprendiz, y posteriormente por el clasificador una vez que éste ha sido construido. Por lo tanto, antes de aplicar el método inductivo un proceso conocido como indexado se aplica para representar los

textos (en la sección 2.3.1 se describe a más detalle este proceso), el cual necesita ser uniformemente aplicado al conjunto de entrenamiento, así como a los nuevos documentos a ser catalogados. En la figura 2.3 se muestra una visión general de los elementos y procesos necesarios dentro de la clasificación automática de textos[1].

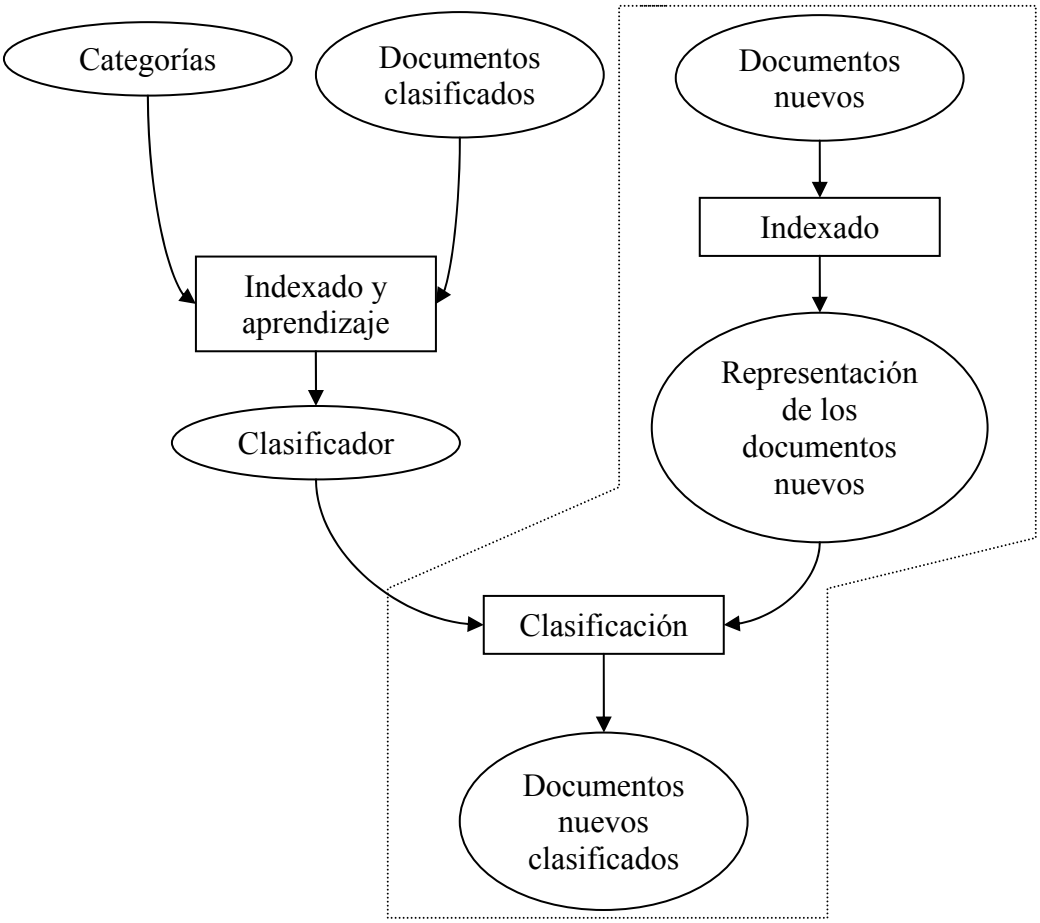


Figura 2.3: Visión general de los procesos en la clasificación de textos

En la evaluación de efectividad para la clasificación de textos, las medidas generalmente usadas son precisión y cobertura [27], que miden lo correcto y completo del método, respectivamente. Además se tiene la medida F, que es una combinación lineal de la precisión y la cobertura (en la sección 2.3.2 se describen a

[1] En todas las figuras del documento los elementos se representan con óvalos y los procesos con cuadros.

detalle estas medidas). En la práctica se tienen colecciones de textos que son ampliamente usadas para comparar metodologías usadas en la clasificación de textos, entre las más populares se tiene la colección Reuters-21578[2], que consiste en una compilación de artículos de revistas en línea clasificados bajo categorías relacionadas a economía. En la tabla 2.4 se presenta un resumen de algunos de los resultados reportados sobre esta colección, los valores se indican como el micro-promedio de F_1 (ver tabla 2.5 y ecuación 2.11 para una descripción).

REFERENCIA	NB	C4.5	K-NN	SVM
[12]	75.2			87.0
[20]	72.0	79.4	82.3	86.4
[54]	79.5		85.6	85.9
[49]	73.4	78.9	86.3	86.3

Tabla 2.4: Resultados de diferentes clasificadores sobre la colección Reuters.
La primera columna indica la referencia bibliográfica donde aparecen los resultados

En los resultados listados en la tabla anterior, los autores usaron un conjunto de prueba y entrenamiento de 9,603 y 3,299 documentos, respectivamente; conteniendo un total de 90 categorías. Cabe destacar que en el proceso de construcción de los diferentes clasificadores se utilizaron varias formas para el indexado, entre las que destacan las utilizadas en este trabajo. También algunos de los autores evaluaron más métodos de aprendizaje que los mencionados en la tabla; sin embargo, decidimos comparar sólo aquellos que fueron descritos en la sección 2.2, los cuales se encuentra entre los mejor evaluados.

2.3.1 Representación de los Documentos

El *indexado* denota la actividad de hacer el mapeo de un documento d_j en una forma compacta de su contenido. La representación más comúnmente usada para representar

[2] http://www.research.att.com/~lewis/reuters21578.html

cada documento es un vector con términos ponderados como entradas, concepto tomado del modelo de espacio vectorial usado en recuperación de información [5]. Es decir, un texto d_j es representado como el vector $\vec{d}_j = \langle w_{ij}, \ldots, w_{|\tau| j} \rangle$, donde τ es el *diccionario*, i.e. el conjunto de *términos* que ocurren al menos una vez en algún documento de Tr, y w_{kj} representa cuanto el término t_k refleja el contenido del documento d_j. En ocasiones τ es el resultado de filtrar las palabras del vocabulario con respecto a una lista de *palabras vacías* (i.e. palabras frecuentes que no transmiten información, e.g. artículos y preposiciones) y un *lematizador* (i.e. obtener sus raíces morfológicas por aplicar un algoritmo de eliminación de afijos en una palabra de tal modo que aparezca sólo su raíz léxica, e.g. el algoritmo de Porter[3] que elimina sufijos de términos en inglés). Con respecto al peso w_{kj} se tienen diferentes formas de calcularlo, entre las más usadas en la comunidad científica se tienen el ponderado Booleano y ponderado por frecuencia de término, las cuales son también empleadas en el presente proyecto, por lo tanto a continuación se da una breve descripción.

- *Ponderado Booleano*: Consiste en asignar el peso de 1 si la palabra ocurre en el documento y 0 en otro caso

$$W_{kj} = \begin{cases} 1 & si \quad t_k \quad aparece \quad en \quad d_j \\ 0 & en \quad otro \quad caso \end{cases} \tag{2.9}$$

- *Ponderado por frecuencia de término*: Es asignar el número de veces que el término t_k ocurre en el documento d_j

$$W_{kj} = f_{kj} \tag{2.10}$$

En la clasificación de textos la alta dimensionalidad del espacio de términos (un amplio valor de $|\tau|$) puede ocasionar un *sobre-ajuste* en el proceso de aprendizaje, i.e. ocurre el fenómeno por el cual un clasificador es adaptado también a las

[3] una versión para el idioma español del algoritmo se encuentra en http://snowball.tartarus.org/

características contingentes de Tr en lugar de únicamente a las características constitutivas de las categorías, provocando problemas de efectividad debido a que el clasificador tiende a comportarse mejor sobre los datos con los que ha sido entrenado y sin conservar la tendencia en aquellos no vistos, además de que la alta dimensionalidad también se refleja en la eficiencia, haciendo el problema menos tratable para el método de aprendizaje. Para evitar el problema un subconjunto de τ es a menudo seleccionado (i.e. una *selección de características*), este proceso se conoce como *reducción de dimensionalidad.* Su efecto es reducir el tamaño del vector de características de $|\tau|$ a $|\tau'|<<|\tau|$; el conjunto τ' es llamado el *conjunto de términos reducido*. La reducción es hecha calculando una función de calidad para los términos, y seleccionando aquellos con mayor calificación. En el proyecto actual se empleó como función de calidad la ganancia en la información (ver ecuación 2.6), la cual se encuentra entre las más efectivas de acuerdo con Yang y Perdensen [53], quienes en experimentos hechos a diferentes métodos de reducción de dimensionalidad encontraron que con esta función hasta un 99% del conjunto original de términos pueden ser eliminados, incrementando la exactitud del clasificador y obteniendo una importante mejora en la eficiencia.

2.3.2 Evaluación en la Clasificación de Textos

La *precisión* (π), *cobertura* (ρ) y medida F (F_β) son nociones clásicas de recuperación de información adaptadas a la clasificación de textos. Donde π es la probabilidad de que si un documento aleatorio d_x es clasificado bajo c_i, esta decisión sea correcta. Mientras que ρ es la probabilidad de que si un documento aleatorio d_x debe ser clasificado bajo c_i, esta decisión sea tomada. Estas probabilidades pueden ser estimadas como se muestra en la tabla 2.5, donde el micro-promedio es para dar a las categorías una importancia proporcional al número de ejemplos positivos que le corresponden, mientras que en el macro-promedio todas las categorías importan lo

mismo (las ecuaciones están en términos de la tabla de contingencia para la categoría c_i, ver tabla 2.1).

MEDIDA	MICRO-PROMEDIO	MACRO-PROMEDIO
Precisión (π)	$\pi = \frac{\sum_{i=1}^{\|C\|} TP_i}{\sum_{i=1}^{\|C\|} TP_i + FP_i}$	$\pi = \frac{\sum_{i=1}^{\|C\|} \frac{TP_i}{TP_i + FP_i}}{\|C\|}$
Cobertura (ρ)	$\rho = \frac{\sum_{i=1}^{\|C\|} TP_i}{\sum_{i=1}^{\|C\|} TP_i + FN_i}$	$\rho_i = \frac{\sum_{i=1}^{\|C\|} \frac{TP_i}{TP_i + FN_i}}{\|C\|}$

Tabla 2.5: Promedio de precisión y cobertura a través de diferentes categorías

Además se tiene una combinación lineal de π y ρ calculada por la función F_β [39], ésta es:

$$F_\beta = \frac{\left(\beta^2 + 1\right) \times \pi \times \rho}{\beta^2 \times \pi + \rho} \tag{2.11}$$

aquí β puede ser vista como el grado de importancia atribuido a π y ρ. Si $\beta = 0$ entonces F_β coincide con π, mientras que si $\beta = +\infty$ entonces F_β coincide con ρ. Usualmente, un valor de $\beta = 1$ es usado, el cual atribuye igual importancia a π y ρ.

Capítulo 3

Extracción de Información

El presente capítulo tiene como propósito ubicar al lector en el área donde se enfoca la tesis, i.e. en el desarrollo de sistemas para la extracción de información. Durante la revisión se darán a conocer resultados alcanzados en el área, el objetivo es destacar los aspectos relevantes de las metodologías actuales, además de resaltar puntos negativos de las mismas, donde nuestra meta en la tesis es evitar tales obstáculos. El contenido del capítulo inicia con una descripción de lo que significa la tarea de extraer información (sección 3.1); posteriormente, en la sección 3.2, se expone la arquitectura general para desarrollar estos sistemas; y se continua en la sección 3.3 con una descripción de cómo evaluar la efectividad; para terminar con el estudio se presenta un resumen de los métodos de aprendizaje automático empleados en la extracción de información (sección 3.3). Finalmente, se concluye el capítulo en la sección 3.4 con una breve discusión de las técnicas actuales.

3.1 Objetivo de la Extracción de Información

La *extracción de información* se ocupa de estructurar información contenida en textos que son relevantes para el estudio de un dominio (o escenario) particular, llamado *dominio de extracción*. En otras palabras, el objetivo de un sistema de extracción de información es encontrar y enlazar la información relevante mientras ignora la extraña e irrelevante [11]. A manera de ejemplo, se presenta el siguiente fragmento de una noticia relacionada a ataques terroristas (reportada el 3 de abril de 1990 en la cadena de televisión Inravisión de Bogotá, Colombia [26]):

El senador liberal Federico Estrada Vélez fue secuestrado el tres de abril en la esquina de las calles 60 y 48 oeste en Medellín... Horas después, por medio de una llamada anónima a la policía metropolitana y a los medios, los Extraditables se atribuyeron la responsabilidad del secuestro... La semana pasada Federico Estrada Vélez había rechazado pláticas entre el gobierno y traficantes de drogas.

Un sistema de extracción de información debe ser capaz de ubicar como información relevante los siguientes fragmentos: *secuestro* como el tipo de incidente, *los Extraditables* como el grupo autor, *Federico Estrada Vélez* como el objetivo humano, además de *3 de abril* y *Medellín* como la fecha y lugar, respectivamente, del incidente. Entonces, la salida de un sistema de extracción son los registros que sirven para llenar lo que se conoce como *plantilla de extracción*. Además, los registros extraídos pueden ser complementados con información del dominio (e.g. en la tabla 3.1 se muestra la plantilla de extracción para la noticia antes expuesta). Por lo tanto, los elementos que componen la plantilla de extracción deben ser definidos desde las primeras etapas de desarrollo del sistema y tienen dependencia directa con el dominio de extracción.

INFORMACIÓN DEL INCIDENTE:	
CATEGORÍA	Ataque terrorista
TIPO	Secuestro
FECHA	03 de abril (de 1990)
LUGAR	Medellín (Colombia)
GRUPO AUTOR	Los Extraditables
OBJETIVO HUMANO	Federico Estrada Vélez

Tabla 3.1: Plantilla generada por un sistema de extracción de información

Para finalizar la sección, cabe destacar que algunos autores consideran a la extracción de información como una etapa posterior de la recuperación de información [11], donde la principal diferencia entre ambas radica en que la primera proporciona la información que exclusivamente interesa, mientras que la segunda

proporciona los textos en los que aparece dicha información. Algunas de las nuevas tecnologías tratan de superar las diferencias y tomar ventaja de ambas técnicas, e.g. la generación de "wrappers" para Internet (i.e. extracción de información desde documentos HTML), y la búsqueda de respuestas (i.e. contestar automáticamente a preguntas puntuales por medio del contenido de una colección de documentos) [13].

3.2 Arquitectura General

Los inicios de la extracción de información se ubican a mediados de los años 60's (en [51] se puede encontrar una breve reseña sobre su historia). Sin embargo, es a finales de los años 80's cuando esta tecnología comienza a tener auge, el cual se debe principalmente a tres factores. Primero, el poder computacional ya entonces disponible; segundo, el exceso de información textual existente de forma electrónica; y finalmente, la intervención de la Agencia de Defensa de los Estados Unidos (DARPA[4], por sus siglas en inglés), quienes patrocinaron durante los años de 1987 a 1998 las siete conferencias de entendimiento de mensajes (MUC[5], por sus siglas en inglés) y activaron durante los años de 1990 a 1998 el programa TIPSTER[6] (programa de investigación sobre recuperación y extracción de información del gobierno de EE.UU.), donde las MUC's fueron incluidas. En resumen, las conferencias del MUC fomentaron la competición entre diferentes grupos de investigación para que desarrollasen sistemas de extracción de información, el resultado fue una década de experiencia en la definición, diseño y evaluación de esta tarea, y es dentro de las mismas donde Hobbs [19] definió lo que se considera la arquitectura general para construir sistemas de extracción de información. Ésta arquitectura genérica es descrita como "una cascada de módulos que en cada paso agregan estructura al documento, y algunas veces, filtran información relevante por medio de aplicar reglas". En la figura 3.1 se muestra un bosquejo de la misma.

[4] www.darpa.mil/
[5] www.itl.nist.gov/iaui/894.02/related_projects/
[6] www.ldc.upenn.edu/Catalog/Tipster.html

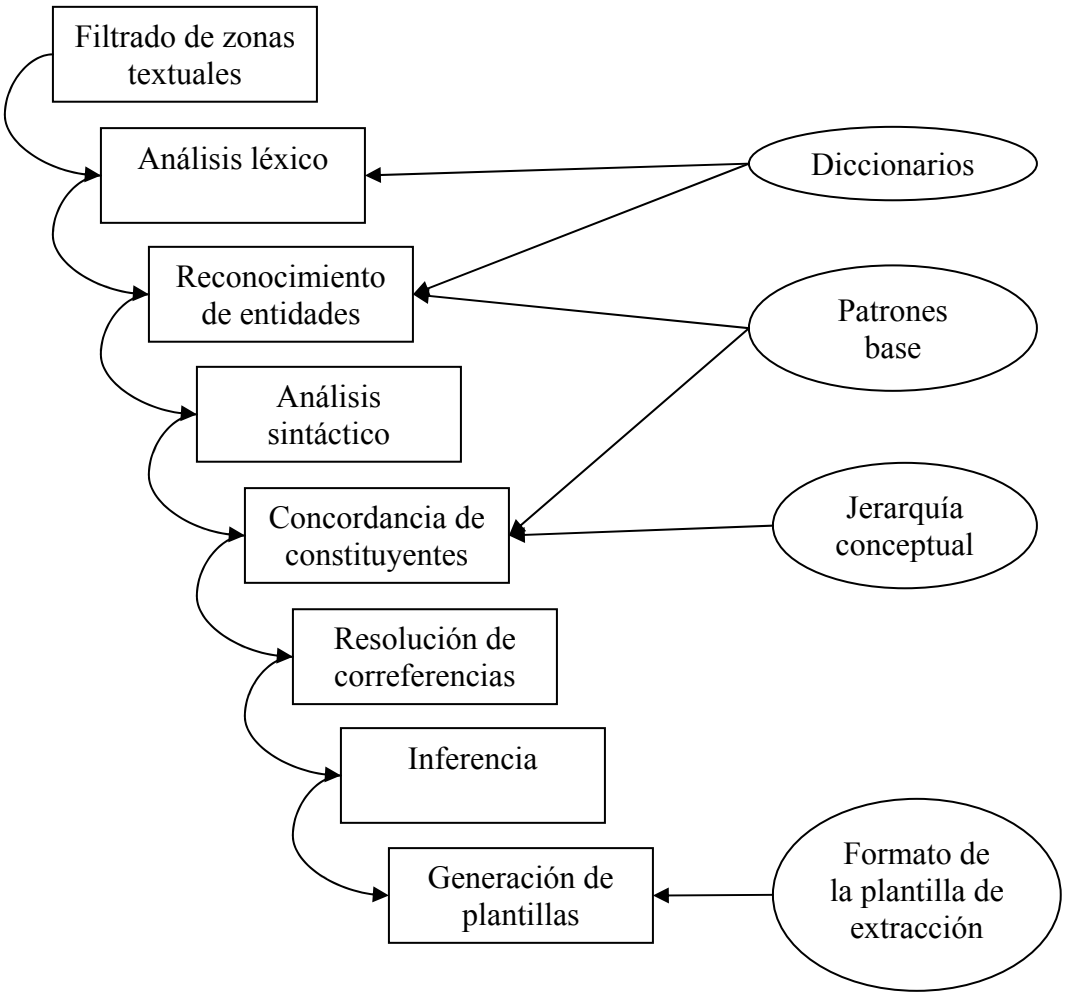


Figura 3.1: Arquitectura genérica de un sistema de extracción de información

Ejemplos de metodologías que siguen fielmente la arquitectura genérica se presentan en [11] y [17]. En resumen, la combinación de módulos que componen la arquitectura permiten en un mayor o menor grado alguna de las cuatro funciones siguientes:

1. El **pre-procesamiento de los documentos**, que es un proceso que puede ser logrado por aplicar una variedad de módulos, por ejemplo:

- La *división en zonas textuales*, que consiste en segmentar un texto en zonas, por ejemplo en párrafos.
- La *segmentación del texto*, que transforma las zonas en segmentos apropiados, usualmente en oraciones.
- El *filtrado de texto*, encargado de seleccionar los segmentos relevantes y de eliminar información irrelevante como ciertas etiquetas o marcas de formato.
- La *tokenización*, que se ocupa de obtener las unidades léxicas (i.e. los tokens) en los segmentos, generalmente son palabras.
- El *analizador léxico*, se ocupa principalmente del análisis morfológico de los tokens (i.e. la forma de las palabras, por ejemplo: género y número), así como del reconocimiento y clasificación de entidades, e.g. nombres propios.
- La *desambiguación*, que tienen como objetivo tratar con palabras que desempeñan diferentes categorías sintácticas y palabras polisémicas que en función del contexto pueden tener un sentido u otro, para esto se hace uso del *etiquetado de las partes de la oración* y de la *desambiguación del sentido de las palabras*.
- La *obtención del lema* y *truncado*, que por conocer las reglas de formación de las palabras permiten proporcionar el lema o raíz de una palabra.

2. El **análisis sintáctico e interpretación semántica**, que tratan en primer término de identificar la forma en que las palabras se combinan para formar constituyentes a nivel sintáctico superior (i.e. los sintagmas); y posteriormente, generar bien una forma lógica o una plantilla parcial desde las sentencias analizadas de forma sintáctica. Ejemplos de módulos útiles para este propósito son:

    - El *análisis sintáctico completo*, que se encarga de decidir si una oración es gramaticalmente correcta, por lo tanto depende de utilizar una gramática muy extensa formada por todas las reglas del lenguaje.

- El *análisis sintáctico parcial*, que tiene como objetivo recuperar información sintáctica de forma eficiente y fiable, desde texto no restringido. Este tipo de análisis sacrifica la completes y profundidad del análisis completo; sin embargo, durante la MUC-3 los mejores resultados fueron logrados por una aproximación de este tipo [25], hecho que fue reafirmado durante la MUC-4 [2]. La argumentación fue que sólo los conceptos que forman parte del escenario de extracción son relevantes para ser detectados en el documento, por lo tanto el análisis sintáctico y semántico debe simplificarse a un análisis de frases. En [16] como parte de la MUC-6 se expusieron una serie de puntos a favor del uso del análisis parcial, destacando que el análisis completo no es un proceso robusto debido a que no siempre se tiene un árbol sintáctico general.
- La *concordancia de patrones*, una vez que los fragmentos del texto han sido etiquetados de forma sintáctica (llamados *frases constituyentes*), el sistema puede determinar dependencias específicas al dominio de extracción entre los constituyentes (i.e. decidir que información debe ser extraída). Para determinar tales dependencias simplemente se aplica un emparejamiento de patrones, esto es, patrones específicos del dominio (conocidos como *patrones de extracción*) son usados para identificar la información relevante así como dependencias entre los mismos constituyentes. Tradicionalmente la manera de obtener los patrones de extracción era de forma manual por ingenieros del conocimiento y expertos en el dominio, en la actualidad varias técnicas empíricas están siendo empleadas para automatizar el proceso (en la sección 3.4 se profundiza en el tema). Finalmente cabe destacar que la representación de los patrones de extracción no es estándar, i.e. difiere de acuerdo a la metodología usada en el sistema de extracción de información.
- Las *relaciones gramaticales*, al igual que en la concordancia de patrones el objetivo es determinar las dependencias entre los constituyentes; pero, en lugar de emplear patrones de extracción se utiliza un modelo sintáctico más flexible [48]. El modelo consiste en definir un conjunto de relaciones

gramaticales entre las entidades, estas relaciones generalizan aspectos como tiempo, lugar, sujeto, objeto, entre otras. Para representar el modelo se utiliza un grafo donde los nodos representan las partes de los constituyentes y los arcos etiquetados representan las relaciones entre éstos.

3. El **análisis del discurso**, que se ocupa de resolver aspectos semánticos como son: la elipsis (i.e., omitir en la oración una o más palabras) y la anáfora (i.e., asumir el significado de una parte del discurso ya emitida). Para esto, los sistemas de extracción de información generalmente proceden de dos formas:

   - Representar la información extraída como *plantillas llenadas parcialmente* y posteriormente usar algún procedimiento de fusión.
   - Representar la información extraída como *formas lógicas* para usar procedimientos tradicionales de interpretación semántica.

4. La **generación de plantillas de salida**, donde el propósito es enlazar las piezas de información extraídas con el formato de salida deseado. Sin embargo, en esta fase puede requerirse algún tipo de inferencia, la cual es provocada por restricciones específicas del dominio para la estructura de salida, por ejemplo:

   - Registros que toman valores desde un conjunto predefinido.
   - Registros que deben ser normalizados, por ejemplo fechas.
   - Registros que son obligados a ser llenados, en contraste con los registros opcionales que pueden bien no tener piezas de información asignadas.
   - Existencia de diferentes plantillas de extracción para ciertas clases de información.

3.3 Evaluación

Las MUC's además de promover la creación de sistemas de extracción de información, también se encargaron de definir los métodos para su evaluación. Y fue durante la MUC-3 que se establecieron las bases de las métricas actuales, las cuales fueron evolucionando en el transcurso de las competencias siguientes (para un estudio de la evolución ver [26]). En este trabajo utilizamos la métrica empleada en la última competición del MUC [8]. Donde, para obtener la calificación de un sistema de extracción de información se requieren de tres cosas (ver figura 3.2). Primero, la colección de documentos de donde se va a extraer la información (i.e. los *textos*); segundo, el conjunto de registros extraídos por un grupo de expertos (llamados *claves*); y finalmente, el conjunto de registros que el sistema a ser evaluado extrae (llamados *respuestas*). El objetivo de la evaluación es comparar respuestas contra claves por medio de las cantidades descritas en la tabla 3.2, y posteriormente calcular las medidas de evaluación expuestas en la tabla 3.3.

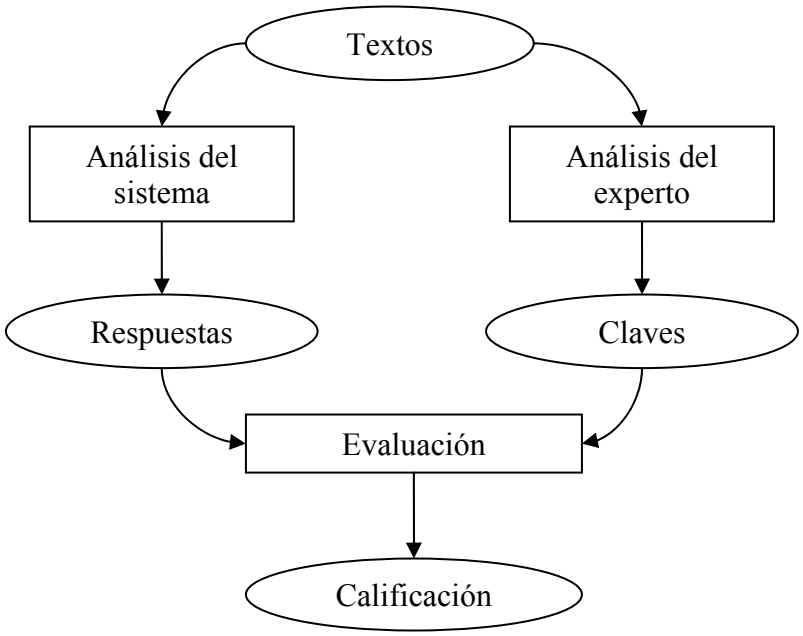


Figura 3.2: Proceso de evaluar un sistema de extracción de información

Número correcto	COR	Ocasiones donde la clave y la respuesta coinciden
Número incorrecto	INC	Ocasiones donde la clave y la repuesta no coinciden
Número perdido	MIS	Ocasiones donde existe una clave pero no una respuesta
Número falso	SPU	Ocasiones donde existe una respuesta pero no una clave
Número evasivo	NON	Ocasiones donde no existen respuesta y clave
Número posible	$POS = COR + INC + MIS$	Número de registros en la clave
Número actual	$ACT = COR + INC + SPU$	Número de registros en la respuesta

Tabla 3.2: Cantidades necesarias para valorar la extracción de información

Cobertura	$REC = \frac{COR}{POS}$
Precisión	$PRE = \frac{COR}{ACT}$
Subgeneración	$UND = \frac{MIS}{POS}$
Sobregeneración	$OVG = \frac{SPU}{ACT}$
Error en respuestas	$ERR = \frac{(INC + SPU + MIS)}{(COR + INC + SPU + MIS)}$

Tabla 3.3: Métricas de evaluación para la extracción de información

Además, a partir de la MUC-4 la medida F_1 fue tomada para mejorar la comparación global de los sistemas (ver ecuación 2.11), y durante las MUC-6 y MUC-7 se definieron nuevas tareas de evaluación (ver tabla 3.4), los objetivos fueron: identificar funciones que fueran en gran medida independientes del dominio (*NE*), alentar a los competidores a crear mecanismos necesarios para profundizar el entendimiento (*COR*) y enfocarse en la portabilidad de los sistemas de extracción (*TE* y *TR*).

Reconocimiento de entidades	*NE*	Tarea de encontrar y clasificar las entidades, e.g., nombres de personas, organizaciones, lugares, expresiones temporales y numéricas.
Resolución de correferencias	*CORR*	Tarea de identificar las expresiones en el texto que hacen referencia al mismo objeto.
Plantillas de elementos	*TE*	Tarea de añadir información descriptiva al resultado de *NE*, i.e., estandarizar conceptos (e.g. *persona* y *organización*)
Relación de plantillas	*TR*	Tarea de identificar las relaciones entre las diferentes *TE*, e.g., *empleado de*, *localizado en* y *producto de*.
Plantillas de escenario	*ST*	Tarea original del MUC de extraer información, i.e., reunir los resultados de *TE* en el escenario específico.

Tabla 3.4: Tareas de evaluación propuestas a través del MUC-6 y MUC-7

Cabe destacar que durante las competencias se subrayó la dificultad de la tarea general de extraer información, siendo la tarea *ST* la más difícil, donde los resultados en la medida *F* estuvieron por debajo del 60% (ver tablas 3.5 y 3.6 para un resumen). Además, las MUC también demostraron que la tarea de extracción de información es

difícil hasta para las personas, donde analistas humanos alcanzaron un grado de coincidencia entre el 60% y 80% [3].

MUC	AÑO	DOMINIO Y NO. DE PARTICIPANTES
1	1987	Noticias sobre operaciones navales (6 participantes)
2	1989	Mismo dominio del MUC-1 (8 participantes)
3	1991	Noticias sobre atentados terroristas en América Latina (15 participantes)
4	1992	Mismo dominio del MUC-3 (17 participantes)
5	1993	Dos dominios: noticias sobre fusiones de empresas y anuncios de productos microelectrónicos (17 participantes)
6	1995	Noticias sobre sucesiones de dirección en eventos financieros (17 participantes)
7	1998	Dos dominios: noticias sobre accidentes de avión y lanzamientos de misiles y artefactos (5 participantes para la tarea *ST*)

Tabla 3.5: Dominios de extracción utilizados en las MUC´s

MUC	NE	COR	TE	TR	ST
1	No hubo criterio de evaluación				
2	El criterio de evaluación no fue adecuado				
3					$REC < 50$ $PRE < 70$
4					$F_1 < 56$
5					$F_1 < 53$
6	$F_1 < 97$	$F_1 < 68$	$F_1 < 80$		$F_1 < 57$
7	$F_1 < 94$	$F_1 < 62$	$F_1 < 87$	$F_1 < 76$	$F_1 < 51$

Tabla 3.6: Mejores resultados reportados en las MUC´s

3.4 Aprendizaje Automático en la Extracción de Información

Una de las principales desventajas de la tecnología de extracción de información es la portabilidad de sistemas existentes a nuevos dominios e idiomas. En general, la portabilidad implica reajustar manualmente el conocimiento lingüístico que depende del dominio, por ejemplo: diccionarios, gramáticas, patrones de extracción, entre otros. Desde finales de los 90's a la fecha las investigaciones se enfocan en el uso de métodos empíricos para automatizar y reducir el alto costo de la portabilidad, los esfuerzos se concentran principalmente en el uso de técnicas de aprendizaje automático para adquirir de forma automática los patrones de extracción útiles para tratar con un lenguaje y dominio particular, y que además es una de las tareas más costosas en el desarrollo del sistema. Las aproximaciones más comúnmente aplicadas son las que utilizan alguna forma de aprendizaje supervisado (ver [35] [33] para un estudio). En la tabla 3.7 se presenta una clasificación de los tipos de patrones de extracción adquiridos por utilizar aprendizaje automático (este tipo de clasificación fue propuesta en [47]), cabe destacar que los sistemas incluidos en la tabla son sólo una parte de los sistemas actuales, pero que resultan útiles para ejemplificar el estado actual de la tecnología. A continuación se describen brevemente cada una de las categorías en la tabla.

3.4.1 Aprendizaje de Reglas

Esta aproximación es la más comúnmente usada, la cual se basa en un proceso de aprendizaje inductivo del tipo simbólico (e.g. programación lógica inductiva [32]). Algunas de estas aproximaciones trabajan en el contexto de un aprendizaje proposicional, mientras que otras lo hacen en el contexto de aprendizaje relacional. A continuación se detalla cada una.

El *aprendizaje proposicional* se basa en representar los ejemplos de un concepto en términos de la lógica de proposiciones, algunos sistemas desarrollados bajo este paradigma son AutoSlog [40] y CRYSTAL [45]. El objetivo ha sido aprender los patrones de extracción solamente de ejemplos positivos.

<table>
<tr><th>SISTEMA</th><th>CLASE</th><th>MODELO</th><th>TEXTOS</th><th>FRAGMENTO EXACTO</th></tr>
<tr><td>AutoSlog [40]</td><td rowspan="5">Aprendizaje de reglas</td><td>Aprendizaje proposicional</td><td>NE</td><td>No</td></tr>
<tr><td>CRYSTAL [45]</td><td rowspan="3">Aprendizaje relacional</td><td rowspan="2">SE</td><td rowspan="10">Si</td></tr>
<tr><td>SRV [14]</td></tr>
<tr><td>RAPIER [7]</td><td>NE, SE, E</td></tr>
<tr><td>WHISK [46]</td><td rowspan="3">Separadores lineales</td><td rowspan="3">Clasificadores</td><td rowspan="2">SE</td></tr>
<tr><td>Textractor [55]</td></tr>
<tr><td>SNoW-IE [41]</td><td>NE, SE</td></tr>
<tr><td>CoA [23]</td><td rowspan="3">Aprendizaje estadístico</td><td rowspan="3">Modelos ocultos de Markov</td><td rowspan="3">SE</td></tr>
<tr><td>[43]</td></tr>
<tr><td>[15]</td></tr>
<tr><td>[23]</td></tr>
</table>

Tabla 3.7: Patrones de extracción adquiridos con aprendizaje automático. La primera columna indica el nombre del sistema y/o la referencia bibliográfica donde se reporta. En la columna textos, NE representa textos no estructurados, SE textos semi estructurados y E textos estructurados

En esta técnica, los ejemplos de un concepto son representados generalmente como un conjunto de atributos. Mientras que los valores a ser extraídos son los núcleos de frases sintácticas que ocurren en los documentos de entrenamiento. Por ejemplo, una de las primeras aproximaciones fue AutoSlog que genera patrones de extracción llamados *nodos de concepto*, donde un nodo es definido como una palabra *disparador* que puede activar el nodo de concepto que realiza la extracción, además un conjunto de restricciones envuelven al disparador y a los valores extraídos (ver figura 3.3). En resumen, sistemas que siguen esta aproximación son útiles para extraer información a partir de textos no estructurados (NE), sin embargo la extracción del fragmento de texto relevante no es exacta (i.e., a pesar de que se detecta la información correcta ésta no se extrae íntegramente).

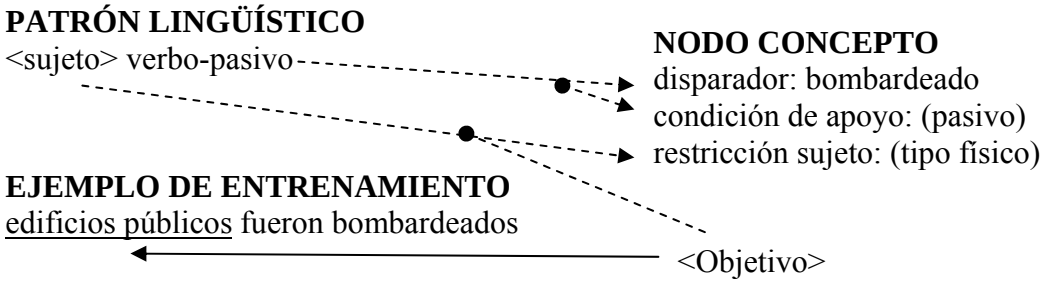


Figura 3.3: Un nodo de concepto inducido por AutoSlog

En el *aprendizaje relacional* los ejemplos de un concepto son representados en términos de lógica de predicados, i.e. los patrones de extracción se representan como atributos y relaciones entre elementos textuales. Ejemplos de esta aproximación son SRV [14], RAPIER [7] y WHISK [46]. Un caso especial para la tesis es SRV, donde transforman el problema de aprender los patrones en una clasificación, i.e., ¿debe un cierto fragmento de texto ser extraído? La entrada de este sistema es un conjunto de características relacionadas a los tokens que representan ejemplos positivos y negativos de los patrones, la salida es un conjunto de reglas que ayudan a contestar la pregunta de clasificación (en la figura 3.4 se presenta un ejemplo de los patrones inducidos por SRV). En general, los sistemas que siguen esta aproximación son útiles para extraer información a partir de textos no estructurados (NE), semi estructurados (SE) y estructurados (E), además de que se tiene una exacta extracción de los fragmentos de texto relevantes.

```
interlocutor:-                                  //  F es un interlocutor si
   cierto(?A,[],token,*desconocido*)            //  F contiene un token (A)
   cada(capitalizar,true)                       //  cada A en F es capitalizado
   longitud(=,2)                                //  F contiene exactamente dos A's
   cierto(?B,[],token,*desconocido*)            //  F contiene otro token (B)
   cierto(?B,[precede],token,":")               //  B es precedido por dos puntos
   cierto(?A,[sucede],doble_char,false)         //  A no es sucedido por 2 caracteres
   cada(cuádruple_char,false)                   //  cada A en F no es de 4 caracteres
   cierto(?B,[2_precede],token,"quién")         //  2 tokens antes de B está la palabra
                                                //  "Quién"
```

Figura 3.4: Regla inducida por SRV

3.4.2 Separadores Lineales

Esta técnica surgió por la necesidad de crear de una forma más rápida los patrones de extracción requeridos. Es decir, en comparación con los sistemas de aprendizaje de reglas donde en algunos casos se requiere la intervención del experto en el proceso de aprendizaje. En esta metodología se emplean algoritmos de aprendizaje automático para tratar de inducir el conocimiento necesario a partir de un conjunto de documentos de entrenamiento, donde cada entidad a ser extraída está etiquetada. Además, en esta técnica también se incluyen entidades que representan ejemplos negativos a la extracción. La hipótesis que se toma es que las entidades relevantes y las irrelevantes son linealmente separables, i.e. el problema de extracción se transforma en un problema de clasificación (ver figura 3.5).

Generalmente, los patrones de extracción son inducidos por generalizar los contextos de las diferentes entidades, por lo tanto los patrones obtenidos están implícitos en el clasificador. Ejemplos de esta aproximación son: Textractor [55], SNoW-IE [41] y CoA [23]. Este último ejemplo es de especial interés para la tesis debido a la filosofía de no utilizar un análisis lingüístico sofisticado para extraer información de documentos no estructurados; sin embargo, el dominio y la plantilla de extracción del sistema resultan ser demasiado simples para obtener conclusiones importantes acerca de la propuesta (i.e., extrae direcciones de correo electrónico desde los correos que llegan a una cuenta determinada con el propósito de mantener actualizada una lista de contactos). En el siguiente capítulo se profundiza más esta idea dado que es fundamental para la arquitectura aquí planteada, la cual se distingue por enfocarse a tareas de extracción de mayor complejidad. En resumen, esta técnica se ha empleado principalmente para tratar con textos semi estructurados (SE) y en algunos casos sencillos de textos no estructurados (NE), además de que se tiene una exacta extracción de los fragmentos de texto relevantes.

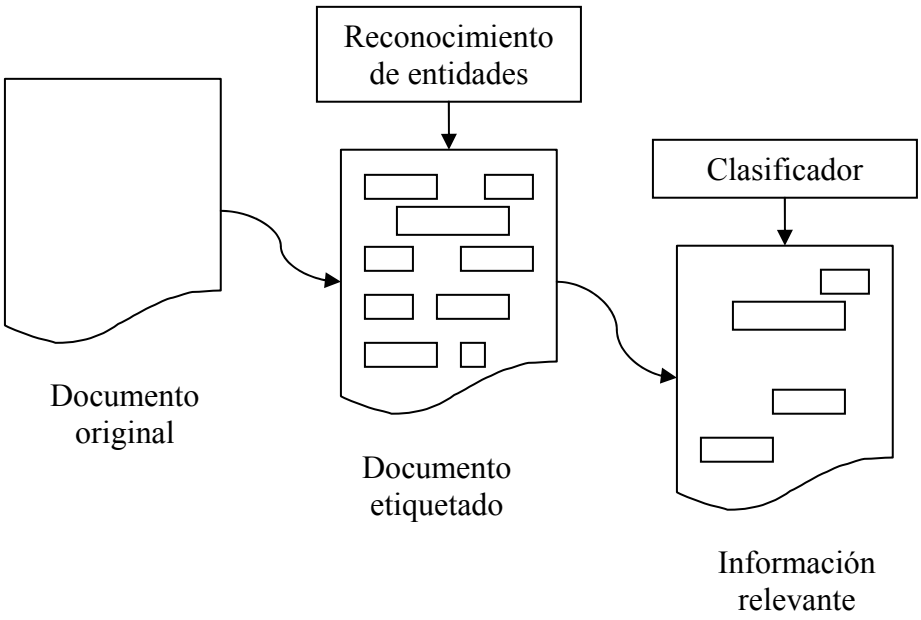


Figura 3.5: Extracción de información como clasificación de entidades

3.4.3 Aprendizaje Estadístico

Los modelos ocultos de Markov (HMMs, por sus siglas en inglés) son la base de esta aproximación, en esta estructura se representa el conocimiento necesario para extraer los fragmentos relevantes de los textos (i.e. los patrones de extracción son representados por HMMs). Ejemplos de esta aproximación son presentados en [43], [15] y [23]. Aquí, generalmente los nodos representan tokens o elementos característicos de éstos, y los enlaces representan sus relaciones, además cada enlace tiene asociada una probabilidad de ocurrencia obtenida de los datos de entrenamiento (ver figura 3.5 para un ejemplo tomado de [23] donde se extrae información desde tarjetas de presentación para llenar un directorio de contactos). En resumen, la relevancia del método es que aprovecha la estructura intrínseca de algunos textos, por lo tanto es adecuado para textos semi estructurados (SE), además de que los fragmentos de texto obtenidos son exactos.

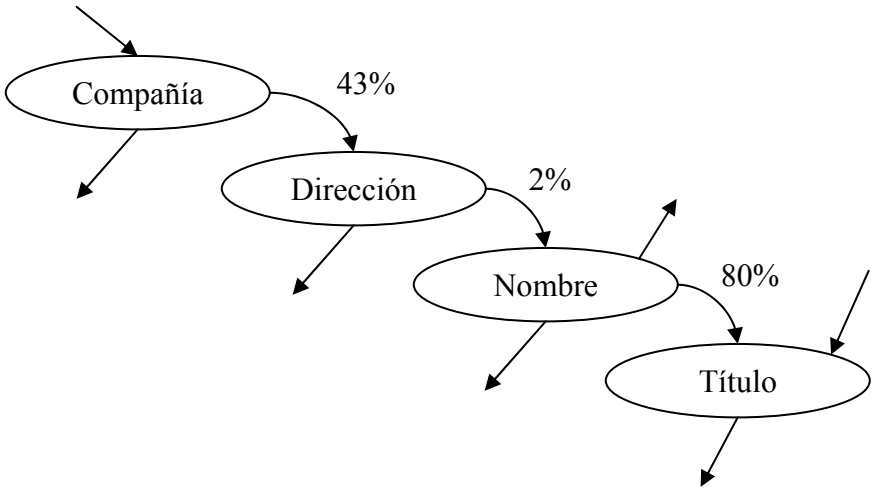


Figura 3.6: Parte de la estructura inducida con HMM's

3.5 Discusión

En general, el objetivo de un sistema de extracción de información está bien definido; además, gracias a la intervención de promotores que impulsaron el desarrollo de estos sistemas, actualmente se tienen avances importantes. Uno de esos avances fue la definición de una arquitectura general; sin embargo, en la descripción de los módulos que componen esta arquitectura, presentada en la sección 3.2, se pudo distinguir la fuerte dependencia de recursos lingüísticos que ésta tiene. Además, si tomamos en cuenta que actualmente tales módulos logran diferentes rangos de exactitud (i.e., no son completos), motivo por el cual se genera un problema de *propagación de error*, esto quiere decir que pequeños errores pueden producir grandes fallas a través del proceso de extracción de información.

Por otro lado, se pretende que los sistemas de extracción de información actuales tengan una mayor portabilidad a nuevos dominios e idiomas, lo que implica generar el conocimiento específico para la nueva tarea. Con el propósito de manejar tal dificultad, las aproximaciones actuales hacen uso de métodos de aprendizaje automático (presentado en la sección 3.4), donde los sistemas basados en aprendizaje de reglas son los más difundidos y además tienen una mejor respuesta. Pero, el error

de propagación en estos sistemas no se ha superado debido a que continúan dependiendo de módulos que tienen una baja exactitud (e.g., análisis sintáctico). También existen las aproximaciones que emplean separadores lineales y aprendizaje estadístico, estos métodos son útiles en el sentido de que pueden trabajar con un mínimo o a veces nulo conjunto de módulos para las etapas de proporcionar estructura a los documentos. Pero, en el aprendizaje estadístico generalmente se requieren textos semi estructurados, mientras que en los separadores lineales también se trabaja con documentos sin ninguna estructura.

Además, cabe destacar que a pesar de los esfuerzos en definir una métrica de evaluación, existen trabajos donde critican la valoración actual (ver [24] para un estudio). Una de las críticas más relevante es la presentada en [22], donde castiga principalmente la falta de alineación en la evaluación con respecto a las respuestas obtenidas por el experto contra las del sistema. Es decir, la valoración no debe ser la misma si el sistema tomó la respuesta de una parte del texto diferente a la que el experto consideró, a pesar de que sea la misma respuesta (e.g., en la noticia de la sección 3.1 no es lo mismo extraer de la primera línea que *Federico Estrada Vélez* es el objetivo humano que tomarlo de la penúltima línea). También, actualmente se trabaja en cómo analizar formalmente la complejidad de un dominio con respecto a una tarea de extracción de información, avances interesantes en este aspecto se muestran en [6], donde además se intenta analizar la tarea de comprensión de lectura. En resumen, se puede concluir que la extracción de información es un campo de investigación ampliamente abierto, y que los esfuerzos en mejorar el desarrollo así como la evaluación de la tarea continúan actualmente (e.g., el programa PASCAL[7] fundado el 1 de junio de 2004 donde, entre otras cosas, se pretende examinar y evaluar metodologías de aprendizaje automático para la extracción de información).

[7] www.pascal-network.org

Capítulo 4

Extracción de Información como un Problema de Clasificación

Tomando en cuenta los aspectos positivos y negativos de las aproximaciones actuales para construir sistemas de extracción de información, en este capítulo presentamos nuestra contribución al campo. El trabajo propuesto consiste de una arquitectura para extraer información, la cual definimos como "un conjunto de componentes que en una primera etapa identifican fragmentos del texto con posibilidad de ser extraídos, y posteriormente deciden cuáles de estos fragmentos son los relevantes al dominio". Durante el capítulo se describe, en primer lugar, la idea detrás de la arquitectura propuesta (sección 4.1); posteriormente, en la sección 4.2, se presenta y detalla la arquitectura; y se concluye en la sección 4.3 con una discusión acerca de la misma.

4.1 Introducción

Como se discutió en el capítulo 3, la arquitectura general para construir sistemas de extracción de información presenta una fuerte dependencia de módulos que proporcionan estructura al texto. Hasta la fecha, la mayoría de estos módulos son incompletos, lo que se refleja en la propagación de error discutida en la sección 3.5. Además, la mayoría de aproximaciones que utilizan métodos empíricos siguen dependiendo en cierto grado de los mismos módulos, lo que perjudica la portabilidad pretendida. Porque a pesar de hacer más factible cambiar entre dominios, el cambio de idioma depende de la disponibilidad de los módulos para el nuevo lenguaje. Algo que en la actualidad es difícil de obtener, siendo el idioma inglés el que mayores recursos presenta. Por tal motivo, en este trabajo mostramos un enfoque alterno al

tradicional, donde el objetivo no es estructurar los documentos, sino más bien detectar estructuras intrínsecas dentro de las expresiones que engloban a la información que deseamos extraer. En otras palabras, no se pretende entender el contenido del documento, en lugar de eso, se aprende a decidir si un fragmento de texto debe o no ser extraído. Lo cual no es una idea nueva, como se describió en la sección 3.4, el sistema SRV y más precisamente las aproximaciones basadas en separadores lineales trabajan bajo este esquema. Sin embargo, la diferencia esencial de nuestro trabajo es que se prescinde de recursos lingüísticos sofisticados. Además, contrastando el hecho de que la propuesta aquí planteada se enfoca en analizar documentos sin formato, esto con el propósito de mostrar los alcances y limitaciones de este tipo de métodos.

Para ilustrar de mejor forma la esencia de la arquitectura, en la figura 4.1 se presenta un ejemplo de extracción de información con la idea planteada, en la cual se consideran sólo dos etapas para realizar la tarea, éstas son: en primer lugar una etapa de *detección de segmentos de texto candidatos*, que consiste en analizar el texto con el objetivo de localizar fragmentos del mismo que sean fuertes candidatos a ser extraídos (en la sección 4.1.1 se describe el análisis realizado). La tarea en esta etapa es similar a la realizada por el modulo de reconocimiento de entidades de la arquitectura general. Posteriormente, se realiza la segunda etapa llamada *selección de información relevante*, la cual se encarga de decidir cuáles de los fragmentos de texto detectados en la etapa anterior representan información relevante a la plantilla de extracción. Esta decisión se toma por analizar el contexto de cada entidad (en la sección 4.1.2 se describe el proceso realizado). El resultado de esta etapa es similar a lo que se obtiene con el módulo de concordancia de constituyentes de la arquitectura general (en la sección 3.2 se presentó la descripción de la arquitectura general).

4.1.1 Detección de Segmentos de Texto Candidatos

Uno de los objetivos principales de la tesis es simplificar el entendimiento del lenguaje en la tarea de extracción de información. Por tal motivo, en el desarrollo del

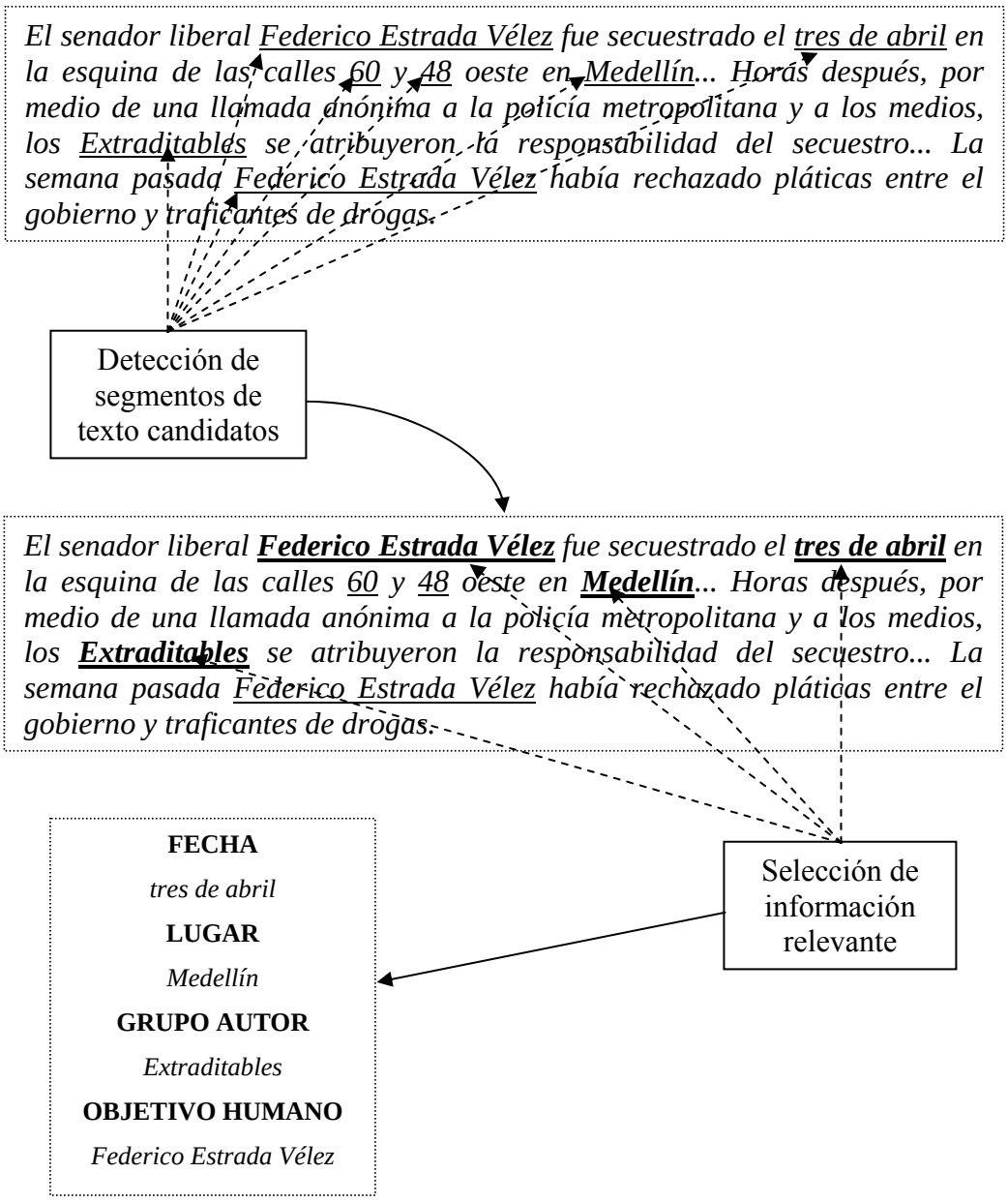


Figura 4.1: Extracción de información con la arquitectura planteada

componente para la detección de segmentos de texto candidatos, se tomaron las observaciones hechas en la referencia [16] para el reconocimiento de entidades. En las cuales se recomienda buscar palabras clave que informen el tipo de entidad que se está tratando (e.g., *Sr*., *Sra*.) y utilizar diccionarios.

Entonces, la propuesta que tenemos para detectar las entidades es utilizar un análisis de expresiones regulares. Es decir, analizar el texto palabra por palabra con un analizador léxico hasta encontrar todas las ocurrencias de las expresiones regulares buscadas. Por lo tanto, limitamos la extracción de información únicamente a entidades que presentan una forma regular, tal como ocurre con formatos de fechas, nombres propios y cantidades. La gramática empleada en el proyecto para detectar las entidades es la siguiente:

entidad_fecha	→	*mes* |
		mes *conector_fecha* *número* |
		número *conector_fecha* entidad_fecha
entidad_nombre	→	*nombre* |
		nombre *conector_nombre* entidad_nombre
entidad_cantidad	→	*número*(["." | ","]*número*)? |
		número(["." | ","]*número*)? entidad_cantidad

Tabla 4.1: Gramática útil para el reconocimiento de entidades

en esta gramática, los símbolos terminales (términos en cursiva) generan grupos de cadenas dados por las siguientes expresiones regulares:

mes	→	enero | ... | diciembre
conector_fecha	→	de | - | ... | ϵ
nombre	→	[A-Z][A-Za-z]*
conector_nombre	→	de | la | ... | ϵ
número	→	[0-9]+

además, en el análisis léxico también se emplean diccionarios de palabras que son útiles para tratar las excepciones en la gramática. Es decir, palabras que deben ser descartadas como entidades porque comúnmente inician con una letra mayúscula

pero no representan un nombre propio, así como palabras para identificar cantidades que no son reportadas con números. Un ejemplo del análisis con expresiones regulares para identificar los segmentos de texto candidatos se presentó en la figura 4.1 (los segmentos de texto subrayados son los detectados con el análisis léxico), en el texto palabras como *El*, *tres*, y *Las* son excepciones a la gramática de la tabla 4.1, las cuales se solucionan por incluir estos términos en los diccionarios usados en el análisis.

En resumen, la independencia que se tiene hacia componentes que realicen un sofisticado análisis lingüístico permite adaptar más fácilmente la idea a un idioma diferente (i.e., sólo se necesita redefinir la gramática). Además, el componente actual puede ser directamente adaptado a un nuevo dominio, siempre y cuando la información relevante del nuevo escenario cumpla el mismo formato regular. Sin embargo, un inconveniente en este tipo de análisis es que las entidades deben estar de forma explícita en los textos para poder ser detectadas.

4.1.2 Selección de Información Relevante

Una vez identificadas las entidades con posibilidad de ser extraídas, el componente para seleccionar la información relevante se encarga de filtrar las entidades que son consideradas útiles para llenar la plantilla de extracción. Este proceso de selección de entidades (i.e., decidir si debe o no ser extraída) se basa en las palabras que la rodean (i.e., su contexto); en otras palabras, la idea es clasificar su contexto para deducir su identidad. Por lo tanto, un clasificador es necesario, el cual puede ser creado de manera similar a un clasificador de textos (ver sección 2.3). Entonces, siguiendo esta aproximación basta definir el proceso inductivo y contar con una colección de contextos previamente clasificados para crear el componente. En resumen, podemos decir que definimos una parte importante de la extracción de información como una tarea de clasificación de textos.

Un inconveniente de está aproximación es que las técnicas actuales para representar textos no resultan adecuadas para el caso de contextos, esto se debe a que

el indexado en la clasificación de textos tiene como objetivo determinar el peso que tienen ciertas palabras en el documento sin importar su posición en el mismo (ver sección 2.3.1), caso contrario ocurre en la clasificación de contextos donde la ubicación de las palabras si importa. Además, generalmente en la clasificación de textos se recomienda eliminar tanto las palabras vacías como los sufijos de los términos debido a que no proporcionan información del tema, pero en la clasificación de contextos estos elementos representan información indispensable para la clasificación. Para ejemplificar lo anterior tenemos los siguientes fragmentos de texto:

Federico Estrada Vélez fue secuestrado,
Los Extraditables fueron los secuestradores

si nuestro propósito es hacer un clasificador de textos para decidir cuándo un fragmento trata sobre secuestros, sin duda las palabras *fue* y *secuestro* resultarían primordiales sin importar su posición en el texto. Pero, si el objetivo es clasificar a la entidad de la primera línea (i.e., *Federico Estrada Vélez*) como el *objetivo humano* y a la entidad de la segunda línea (i.e., *Extraditables*) como el *grupo autor*. Entonces, si empleamos las técnicas actuales para indexar documentos en la clasificación de textos podríamos tener como resultado que ambos segmentos representan la misma clase de información (i.e., *Federico Estrada Vélez* y *Extraditables* son ambos el objetivo humano o bien el grupo autor).

Para tratar con este inconveniente, en el proyecto se utiliza una lista de atributos nominales para representar a cada una de las instancias. Es decir, cada atributo se corresponde con un token del contexto, donde un token puede ser una palabra tomada directamente del texto o bien una etiqueta que indique la presencia de una entidad (información que se obtiene en la etapa previa, i.e., en la detección de segmentos de texto candidatos). En la figura 4.2 se ilustra la representación utilizada en el proyecto, el texto del ejemplo fue tomado de la noticia presentada en la figura 4.1 y representa el contexto de una entidad relevante al dominio (i.e., el objetivo humano: *Federico Estrada Vélez*).

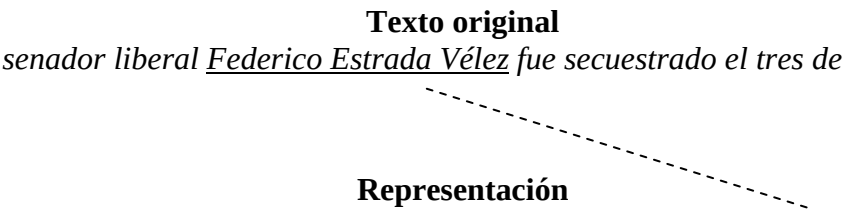


Figura 4.2: Ejemplo de indexado en la clasificación de contextos

Cabe destacar que el orden que guardan los atributos en la lista es el mismo que guardan los tokens en el contexto, a excepción del último atributo en la lista, que se utiliza para indicar la clase de entidad que representa el contexto. Por ejemplo, en la figura 4.2 *Federico Estrada Vélez* es la entidad que queremos clasificar, entonces el último de los atributos en la representación es la categoría que deseamos que se le asigne a ese ejemplo, en este caso el *objetivo humano*. El resto de los atributos en la representación son palabras tomadas del contexto (e.g., *fue* y *secuestrado*), o etiquetas de las entidades que ocurren (e.g. en el contexto se encuentra la entidad *tres de abril*, entonces en la representación se pone una etiqueta que indica la presencia de una *entidad fecha*).

Una vez definida la representación de las instancias, dos puntos más necesitan ser determinados en la metodología:

1. El tamaño de contexto adecuado, i.e., cuántos tokens a la izquierda y derecha de la entidad deben ser tomados como atributos.
2. El algoritmo de aprendizaje empleado para crear el clasificador.

Estas dos incógnitas deben ser determinadas mediante experimentación, debido a que dependen directamente del dominio de extracción y de la información que se desea extraer. Además, no precisamente se debe tener un único clasificador para todas las clases de entidades, es importante mencionar que en este componente pueden coexistir más de un clasificador, debido a que en ocasiones es preferible tratar por separado los diferentes contextos. Por ejemplo, un clasificador para discriminar contextos de fechas, otro para discriminar contextos de nombres propios y otro para

los contextos de cantidades (el caso de estudio presentado en el capítulo 5 es un ejemplo de cómo determinar las incógnitas pendientes).

4.2 Arquitectura

Finalmente, para contestar a una de las dos preguntas generales que se plantearon en la introducción de la tesis, i.e., *¿cómo construir sistemas de extracción de información sin utilizar un análisis lingüístico sofisticado?*, en la figura 4.3 se presenta la arquitectura propuesta, donde se puede ver el escaso o nulo entendimiento del lenguaje que se tiene (i.e., no se utilizan recursos lingüísticos sofisticados, en lugar de eso, se emplea sólo un análisis léxico y métodos de aprendizaje supervisado).

En la arquitectura se distinguen dos fases de desarrollo, la primera es lo que se hace fuera de línea y la segunda es lo que se realiza en línea (i.e., una etapa de entrenamiento y desarrollo, y una etapa de operación). En esta primera fase, el objetivo es definir las expresiones regulares y diccionarios necesarios en la etapa de detección de segmentos de texto candidatos. Además, en la fase fuera de línea también se desarrolla el clasificador de entidades necesario para la etapa de selección de información relevante, para lo cual es necesario definir tanto la representación de las instancias como el algoritmo de aprendizaje automático a utilizar. Posteriormente, en la fase en línea el objetivo es tomar documentos nuevos como entrada, y producir como salida las plantillas de extracción de información correspondientes. En esta fase el proceso que se sigue es: primero se detectan los segmentos de texto candidatos, lo que produce como salida la lista de contextos y entidades a ser analizados. Posteriormente, se aplica el proceso de indexado a los contextos y entidades, de esta forma se obtiene la representación adecuada para trabajar con los métodos de aprendizaje. Finalmente, a la representación de cada instancia se le aplica el proceso de clasificación, donde la categoría asignada a cada instancia sirve para decidir si la entidad que representa debe ser extraída, en otras palabras, se selecciona la información relevante.

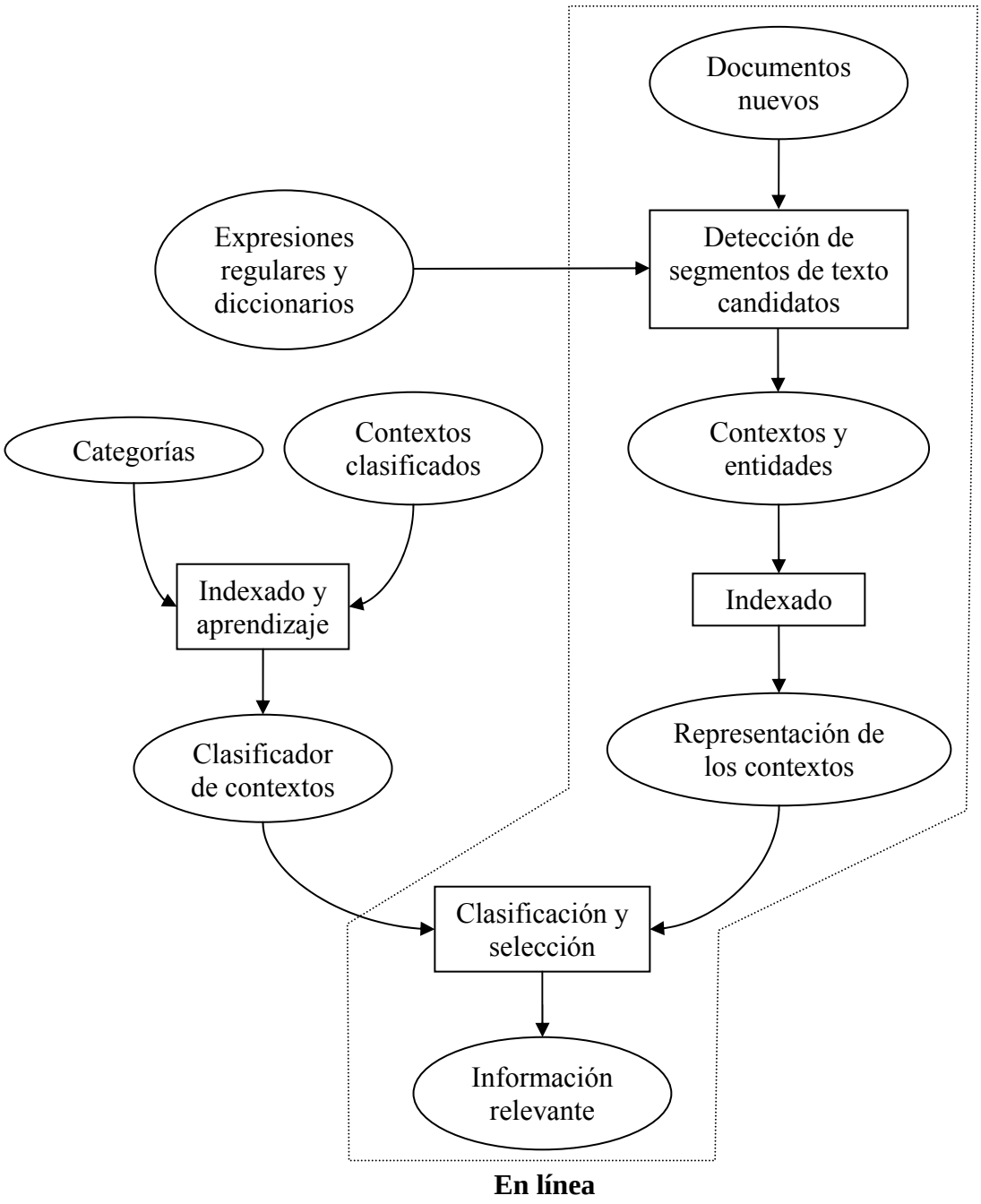


Figura 4.3: Arquitectura de extracción de información basada en clasificadores

Además, tomando en cuenta que es más probable que los textos pertenecientes al dominio de estudio contengan la información requerida, en la arquitectura propuesta también se decidió incluir una etapa previa a la extracción de información,

llamada el *filtrado de documentos*. El objetivo principal es filtrar los documentos relevantes al dominio de estudio, pero también esta etapa de filtrado es útil para llenar registros en la plantilla de extracción que tienen que ver con información acerca de la *categoría* del documento y el tipo de *evento* que trata. En la figura 4.4 se presenta un ejemplo, el cual consiste de un texto catalogado como noticia de *secuestro*, y por lo tanto, se puede concluir que su categoría es *ataque terrorista*.

El senador liberal Federico Estrada Vélez fue secuestrado el tres de abril en la esquina de las calles 60 y 48 oeste en Medellín... Horas después, por medio de una llamada anónima a la policía metropolitana y a los medios, los Extraditables se atribuyeron la responsabilidad del secuestro... La semana pasada Federico Estrada Vélez había rechazado pláticas entre el gobierno y traficantes de drogas.

Figura 4.4: Ejemplo de filtrado de documentos

Para construir el componente de filtrado de documentos se procede como se describió en la sección 2.3. Es decir se crea un clasificador de textos para el dominio de estudio. Por lo tanto, es necesario contar con una colección de documentos previamente clasificados, además de decidir cuál es el método de indexado y el algoritmo de aprendizaje que mejor se adaptan a la tarea. En general, estas decisiones dependen del dominio en estudio y por lo tanto es difícil generar (en el capítulo 5 se presenta un caso de cómo elegir los elementos necesarios).

4.3 Discusión

En este capítulo se presentó una arquitectura para sistemas de extracción de información, la cual se distingue por el uso de métodos empíricos junto con un análisis a nivel léxico de los textos. Sin embargo, un inconveniente en el uso de métodos empíricos, y más precisamente métodos de aprendizaje supervisado, es la necesidad de contar con una colección de ejemplos previamente clasificados para implementar tanto el filtrado de documentos como la extracción de información. Sin embargo, si tomamos en cuenta que clasificar manualmente una colección de textos es más fácil que construir y ajustar un conjunto de reglas, en otras palabras, es más simple caracterizar un concepto extencionalmente (i.e., seleccionar instancias de éste) que intencionalmente (i.e., describir el concepto en palabras), entonces la propuesta aquí planteada presenta posibilidades reales para competir con las arquitecturas actuales para crear sistemas de extracción de información.

Capítulo 5

Caso de Estudio

El presente capítulo tiene como propósito dar respuesta a la última de las preguntas generales planteadas en la introducción de la tesis, i.e., *¿cómo se comporta la arquitectura propuesta en la extracción de información?*. Por lo tanto, con el objetivo de contestar a la pregunta se desarrolló un sistema de extracción de información para el dominio de desastres naturales, el cual está inspirado en una de las tareas realizadas por la red de estudios sociales en prevención de desastres (LA RED[8]). El trabajo consiste en llenar una base de datos con información de catástrofes provocadas por fenómenos naturales. Una definición completa del dominio se presenta en la sección 5.1. Además, los detalles de implementación así como los resultados alcanzados por el sistema son expuestos en la sección 5.2. Posteriormente, en el capítulo 5.3 se hace un análisis de los resultados en torno al inventario de desastres creado por LA RED; y finalmente, en la sección 5.4 se discuten los logros y limitaciones de la arquitectura teniendo como base el caso de estudio.

5.1 Definición del Dominio

Los esfuerzos de LA RED no sólo se limitan a analizar la información relacionada con características y efectos de los diversos fenómenos naturales, también se encargan de definir metodologías para el correcto registro de la misma. Por tal motivo, se decidió tomar como base para la plantilla de extracción la "ficha de información de desastres" establecida por este organismo en su guía metodológica de

[8] www.lared.org.pe

DesInventar[9] (el término designa al sistema de inventario de desastres creado por LA RED). En la tabla 5.1 se presenta la plantilla de extracción considerada.

INFORMACIÓN DEL DESASTRE	
FECHA	Fecha de ocurrencia del desastre
LUGAR	Nombre del lugar o lugares donde ocurrió el fenómeno
MAGNITUD	Valores de magnitud internacionalmente usados para sismo y huracán, para otros tipos de eventos variables cuantificadas
INFORMACIÓN DE LAS PERSONAS	
MUERTOS	Número de personas fallecidas por causas directas
HERIDOS	Número de personas que resultan afectadas en su salud o integridad física, sin ser víctimas mortales, por causa directa del desastre
DESAPARECIDOS	Número de personas cuyo paradero a partir del desastre es desconocido
DAMNIFICADOS	Número de personas que han sufrido grave daño directamente asociados al evento en sus bienes o servicios
AFECTADOS	Número de personas que sufren efectos secundarios asociados a un desastre
INFORMACIÓN DE LAS VIVIENDAS	
DESTRUIDAS	Número de viviendas arrasadas, sepultadas, colapsadas o deterioradas de tal manera que no son habitables
AFECTADAS	Número de viviendas con daños menores, no estructurales o arquitectónicos, que pueden seguir siendo habitadas
INFORMACIÓN DE LA INFRAESTRUCTURA	
HECTÁREAS	Número de áreas de cultivo, pastizales o bosques destruidas o afectadas
ECONÓMICA	Monto de las pérdidas directas causadas por el desastre

Tabla 5.1: Plantilla de extracción para el caso de estudio

Debido a que los tipos de fenómenos naturales que ocurren son diversos, para el desarrollo del sistema se decidió limitar el dominio de extracción a sólo cinco clases de eventos (ver tabla 5.2). Los desastres elegidos son, según los registros de DesInventar, los más frecuentes en México. Además, el objetivo del sistema, en la

[9] www.desinventar.org

tesis[10], es mostrar las capacidades de la arquitectura propuesta más que presentar un análisis exhaustivo del dominio.

EVENTO	DEFINICIÓN
FORESTAL	Incluye todos los incendios en campo abierto en áreas rurales, sobre bosques nativos, bosques cultivados y praderas
HURACÁN	Anomalía atmosférica violenta que gira a modo de torbellino caracterizado por fuertes vientos, acompañados por lluvia
INUNDACIÓN	Subida de aguas que supera la sección del cauce de los ríos o que se relaciona con el taponamiento de alcantarillas
SEQUÍA	Temporada anormalmente seca, sin lluvias, o con déficit de lluvias. En general se trata de períodos prolongados
SISMO	Todo movimiento de la corteza terrestre que haya causado algún tipo de daño o efecto adverso sobre comunidades o bienes

Tabla 5.2: Dominio de extracción para el caso de estudio

5.2 Características Técnicas y Resultados Experimentales

Para el desarrollo del sistema se decidió tomar como conjunto de entrenamiento noticias en español relacionadas a los desastres naturales, las cuales fueron obtenidas de varios periódicos mexicanos que están disponibles en Internet (publicados entre los años de 1996 a 2004). La decisión de trabajar con noticias se debe a que son documentos no estructurados y con una diversidad de estilos, características que nos permite obtener conclusiones de la arquitectura más generales. Además, otro aspecto relevante de tomar noticias, es que habitualmente la información se reporta de manera factual, lo que implica que entidades de información como las buscadas en la arquitectura (i.e., fechas, nombres propios y cantidades) pueden encontrarse de forma explicita, lo cual es un requisito de la metodología empleada. Sin embargo, esta información no siempre es concreta debido a la necesidad de los medios de hacer el reporte lo más pronto posible aún cuando la información no esté totalmente

[10] El sistema forma parte también del proyecto "Recolección, Extracción, Búsqueda y Análisis de Información a Partir de Textos en Español" (Proyecto CONACYT U39957-Y), donde se propone validar las técnicas y procedimientos en información sobre desastres en México y generar un repositorio con información del tema que sirva para estudios exploratorios y preventivos posteriores.

confirmada. Un ejemplo de esta afirmación se presenta en el siguiente párrafo (parte de una noticia publicada el 16 de junio de 1999 en el periódico El Universal):

> *En el peor temblor del siglo en Puebla, ... que afectó los principales edificios, templos y monumentos del centro histórico de la capital y su periferia, dejó ayer un saldo parcial de por lo menos once muertos y 200 heridos entre graves y leves, y un número indeterminado de daños físicos y humanos. Al interior del estado, ... se contabilizaron 120 derrumbes de casas, ..., como resultado del sismo de 6.7 grados en la escala de Richter registrado ayer...*

En resumen, el corpus de noticias obtenidas para implementar y evaluar la arquitectura se enlista en la tabla 5.3. En la tabla se distinguen los ejemplos utilizados en el componente de filtrado de textos (i.e., textos de noticias), de los empleados en el componente de extracción de información (i.e., contextos de entidades), estos últimos ejemplos se obtuvieron analizando trescientas de las noticias relevantes. Por noticias relevantes entendemos todas aquellas que contienen palabras o frases a ser extraídas, mientras que las irrelevantes son las que contienen palabras usadas comúnmente en la descripción de un desastre natural, pero que en estos casos se emplean en contextos diferentes. En el caso de las entidades, las relevantes son las que representan información que debe ser extraída, y las irrelevantes son fragmentos de texto detectados por el reconocedor de entidades pero que no interesan al estudio. En la tabla 5.4 se ejemplifica lo anterior.

	Filtrado de documentos	**Extracción de información**
Relevantes	439	2025
Irrelevantes	229	7926

Tabla 5.3: Corpus de ejemplos para el caso de estudio

	Noticia	Entidad
Relevante	*El huracán Isidore dejó en la península de Yucatán 300 mil personas damnificadas y el deceso de una persona*	*En el peor temblor del siglo en Puebla, 11 muertos*
Irrelevante	*Cuando Beijing estaba en el ojo del huracán de la neumonía atípica, dejó 2 mil 561 enfermos, de los cuales 192 murieron*	*El palacio municipal, construido en 1536, fue el monumento que presentó los daños más severos*

Tabla 5.4: Ejemplo de información relevante e irrelevante al caso de estudio

La porción de información relevante contra la irrelevante, presentada en la tabla 5.3, es un reflejo de la dificultad que implica el filtrado de documentos y la extracción de información para el caso de estudio. Es decir, en el filtrado de documentos además de tener que rechazar aquellas noticias que no tienen nada semejante al dominio, también se debe tener cuidado con aquellas que contienen términos comunes pero son irrelevantes (aprox. el 34% en el corpus del dominio); mientras que en la extracción de información se tiene que de las 300 noticias relevantes que se analizaron, sólo el 20% de las entidades identificadas representan información relevante y el resto es información que no debe ser extraída. A continuación se presentan las características de implementación y los resultados de experimentación obtenidos para cada una de las tareas.

5.2.1 Filtrado de Documentos

Como se describió en la sección 4.2, el componente de filtrado de documentos es la implementación de un clasificador de textos. Por lo tanto, para su desarrollo se necesita definir tanto la representación de los textos como el algoritmo de aprendizaje adecuado para el dominio. En la figura 5.1 se resumen los experimentos realizados, los cuales involucran los ponderados Booleano y frecuencia de término (descritos en la sección 2.3.1), y los algoritmos de aprendizaje naive Bayes, C4.5, k-vecinos más

cercanos y máquinas de vectores de soporte (examinados en la sección 2.2). En la gráfica, la exactitud fue la métrica de evaluación empleada, la cual se obtuvo aplicando el método de evaluación 10FCV sobre una parte del corpus del dominio (ver sección 2.1.2 para una explicación de la evaluación).

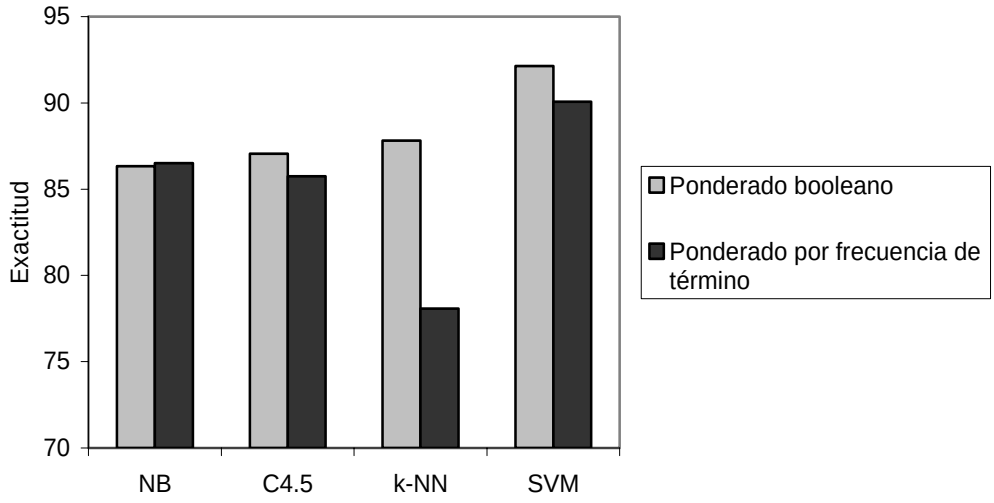


Figura 5.1: Comparación de tipos de indexado y algoritmos de aprendizaje

En resumen, el indexado por ponderado Booleano es el que mejores resultados presenta, esto se debe a la poca frecuencia en los términos característicos (i.e. términos que ayudan a discriminar entre categorías). Con respecto al algoritmo, el que mejor se adapta a la tarea es SVM, hecho que reafirma los estudios actuales, donde a este algoritmo lo ubican entre los mejores para la tarea general de clasificar textos (ver tabla 2.4 para un ejemplo). Una explicación del porqué SVM trabaja bien en la clasificación de textos se puede encontrar en [20], donde se argumenta principalmente que al algoritmo no le afectan la alta dimensionalidad del espacio de características y los problemas con instancias esparcidas (i.e., los vectores que representan a los documentos pueden tener un diccionario de más de 10 mil términos y donde la mayoría de sus entradas pueden tener un valor de cero).

Recordando la sección 2.3.1, un problema en la clasificación de textos es la alta dimensionalidad del espacio de valores característicos, en los experimentos

anteriores este espacio consistió de 134,162 términos (los cuales se obtuvieron por eliminar palabras vacías y sufijos de los 291,766 términos del corpus, i.e., sólo se tomó como diccionario el 46% de los términos originales), y aunque este tamaño parece no afectar drásticamente la efectividad (i.e., existe un caso con una exactitud mayor del 92%), en aspectos de eficiencia resulta costoso tener que hallar tantos términos en un documento para construir su representación.

Con el objetivo de afrontar el problema de la alta dimensionalidad, al clasificador que resultó mejor evaluado (i.e, indexado con ponderado Booleano y SVM como el algoritmo de aprendizaje) se le realizó una reducción de dimensionalidad por aplicar un umbral en la ganancia en la información (como se argumento en la sección 2.3.1 este método se encuentra entre los más efectivos según los experimentos presentados en [53]). En la figura 5.2 se ilustra tanto la efectividad del clasificador en las diferentes categorías así como el número de términos conforme se va aumentando el umbral, en este caso la efectividad se reporta con la medida F_1.

Como se muestra en la figura, cuando se aplicó un primer umbral de GI>0, se eliminaron 133,510 de los términos empleados en los experimentos antes descritos, además de que la efectividad aumentó a un 97%. Posteriormente, se fue aumentando el umbral y como se distingue en las gráficas el número de términos se fue reduciendo y la efectividad del clasificador encontró un punto máximo. Donde el punto máximo corresponde a un umbral de GI>0.025 (línea vertical en la figura), y el número de términos en este punto son 647 (i.e., sólo el 0.5% de los términos del diccionario son útiles para predecir la clase). Además, durante el estudio de la ganancia en la información se encontraron términos característicos de las diferentes categorías, por mencionar algunos se tienen: *incendio*, *bomberos* y *hectáreas* para forestal; *tormenta*, *vientos* y *tropical* para huracán; *encharcamientos, lluvia* y *centímetros* para inundación; *escasez, cultivos* y *agricultura* para sequía; y *richter, sacudió* y *epicentro* para sismo.

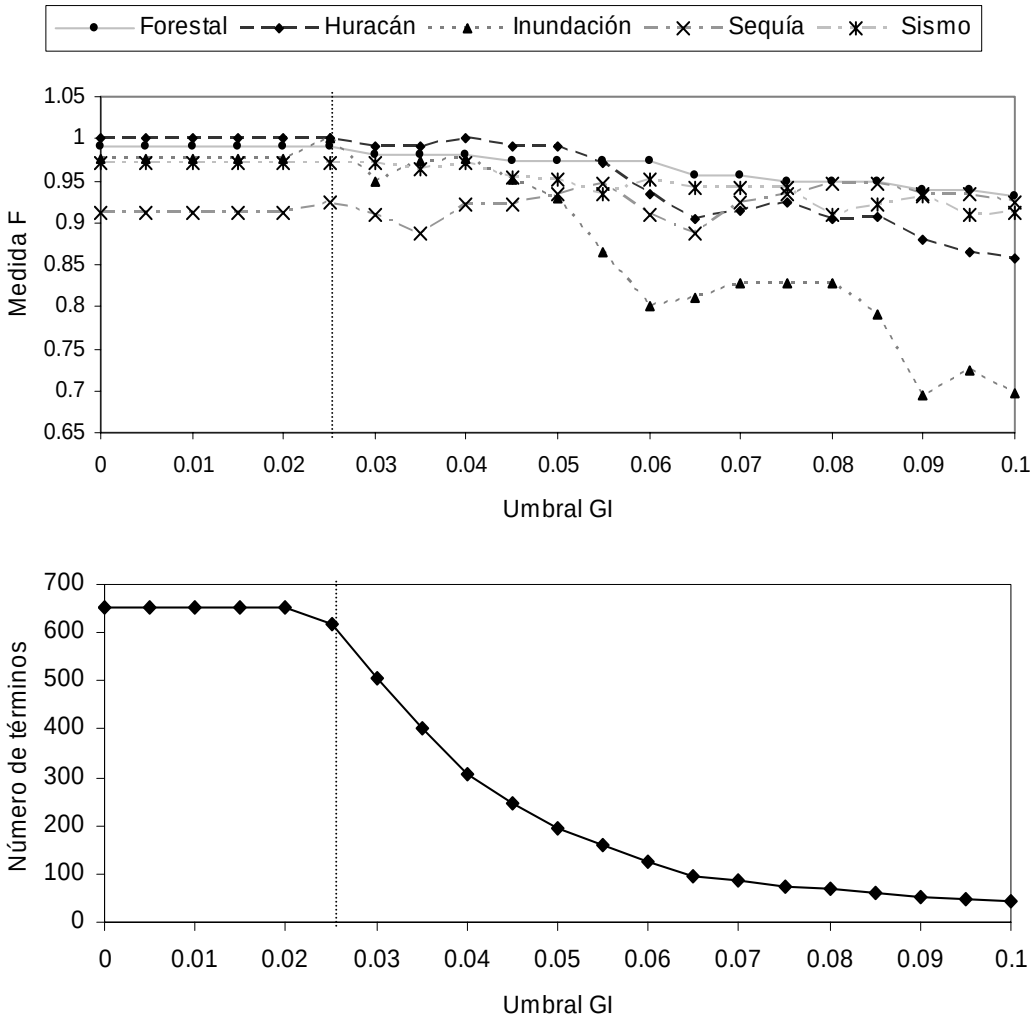


Figura 5.2: Reducción de dimensionalidad por aplicar un umbral a la GI

En conclusión, el clasificador adoptado para el componente es el que tiene un indexado con ponderado Booleano y un espacio de características de 647 términos, donde el algoritmo de aprendizaje es SVM. Además, en una evaluación final del componente, donde se dividió al corpus de noticias en conjunto de entrenamiento y conjunto de prueba, el resultado fue una F_1 igual a 94.6%, lo que reafirma el éxito de aplicar técnicas de aprendizaje automático en la construcción del componente de filtrado de documentos.

5.2.2 Extracción de Información

Como se explicó en la sección 4.1, para extraer información con la arquitectura propuesta se requieren las siguientes dos tareas: la detección de segmentos de texto candidatos y la selección de información relevante. A continuación se dan los detalles técnicos de cada una de las tareas, y se analizan los resultados de evaluación obtenidos.

Con respecto al componente de detección de segmentos de texto candidatos, el objetivo es revelar fechas que puedan reportar la ocurrencia del evento, nombres propios que identifiquen el lugar del desastre, y cantidades que contabilicen las catástrofes ocurridas a personas, viviendas e infraestructura. Para realizar la tarea se utiliza el componente tal y como fue descrito en la sección 4.1.1 (i.e., emplear un análisis con expresiones regulares y diccionarios). La evaluación del componente se realizó sobre el corpus del dominio útil en la extracción de información y utilizando las métricas del MUC para la tarea de nombrar entidades (ver sección 3.3). En la tabla 5.5 se resumen los resultados.

ENTIDAD	REC	PRE	UND	OVG	ERR
FECHA	100	97	0	2	2
NOMBRE	100	70	0	10	10
CANTIDAD	98	100	2	0	2

Tabla 5.5: Evaluación de la tarea NE para el caso de estudio

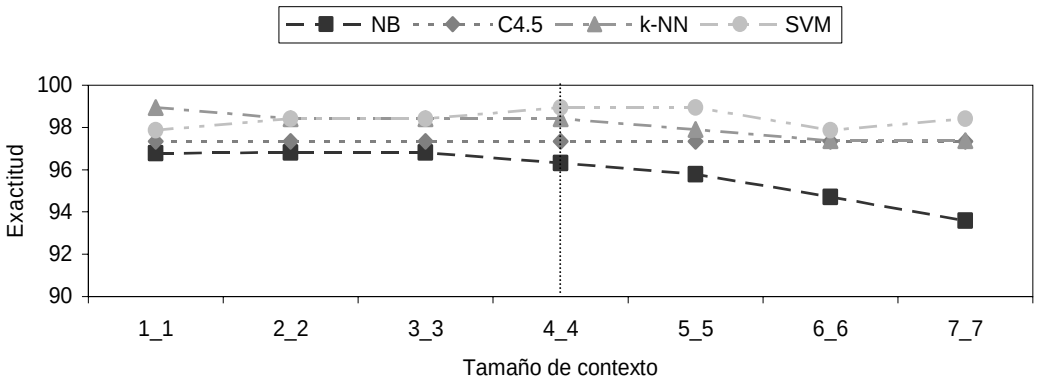
En los valores de la tabla anterior se puede ver que la cobertura (REC), objetivo principal del componente, es casi perfecta a excepción de las cantidades, donde el principal problema es la subgeneración (UND) provocada por la incompletez del diccionario que se utiliza para detectar números reportados con letras. En contraste, la precisión (PRE) se ve afectada por la sobregeneración (OVG) de nombres y fechas, donde el motivo del problema es que hay términos que inician con letra mayúscula y son identificados como nombres propios sin realmente serlo; además de que en el dominio hay nombres de lugares que tienen el formato de una fecha (e.g., *la colonia 8 de febrero*). Sin embargo, a pesar de que las técnicas

empleadas en el componente pueden ser consideradas rudimentarias, el resultado final del mismo es una F_1 de 93%, donde el máximo error en respuestas (ERR) ocurre en los nombres.

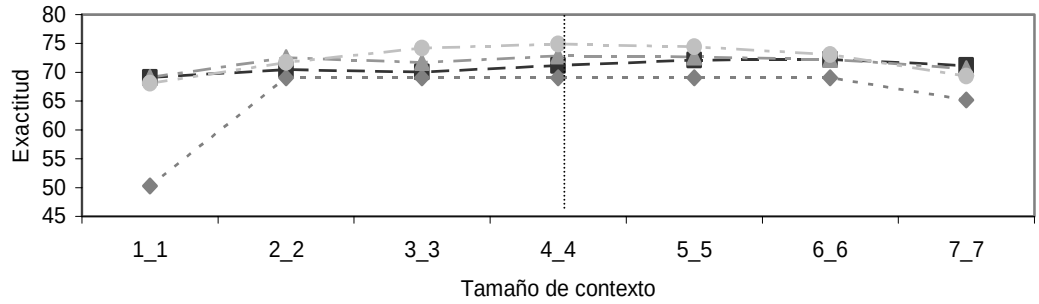
El otro componente necesario para la extracción de información es el de selección de información relevante, el cual es un tipo de clasificador de textos pero que clasifica entidades en base a su contexto. Donde el contexto es representado como una lista de palabras y etiquetas de entidad (ver sección 4.1.2). Para implementar el componente se decidió utilizar tres clasificadores independientes, cada uno de ellos especializado en fechas, nombres y cantidades respectivamente. En la tarea de encontrar el tamaño de contexto y el algoritmo de aprendizaje adecuado para cada clasificador, se experimentó con una parte del corpus de entidades tomado de forma aleatoria y conservando la distribución original. El tamaño del contexto fue tomado de forma simétrica entre uno y siete tokens a la izquierda y derecha de cada entidad (i.e., contextos de mínimo dos tokens y máximo catorce). En la figura 5.3 se resumen los experimentos, donde la evaluación de los clasificadores fue su exactitud obtenida mediante un 10FCV (en la sección 2.1.2 se definió esta evaluación).

De los resultados se puede deducir, bajo el criterio que se prefiere el clasificador con mayor exactitud y menor tamaño de contexto, que el algoritmo SVM es el mejor para todos los casos (i.e., el algoritmo no sólo es útil para problemas con una alta dimensionalidad y con instancias esparcidas como en el caso de la sección 5.2.1). Donde, un contexto de tamaño ocho (i.e., cuatro tokens a la izquierda y cuatro a la derecha) es lo que mejor se adapta para clasificar fechas y nombres, mientras que para el caso de cantidades, un contexto de tamaño seis es el adecuado (i.e., tres tokens a la izquierda y tres a la derecha). Una explicación de lo anterior es que contextos pequeños (e.g. un token a la izquierda y derecha) tienden a ser muy parecidos y por lo tanto es difícil establecer criterios de discriminación; por el otro lado, contextos demasiado grandes (e.g. siete tokens a la izquierda y derecha) tienden a incluir términos que no son relevantes para identificar la entidad y por lo tanto encontrar patrones es más difícil. En consecuencia, los contextos de tamaño medio (e.g. entre tres y cuatro tokens a la izquierda y derecha) son los que mejor se adaptan a la tarea,

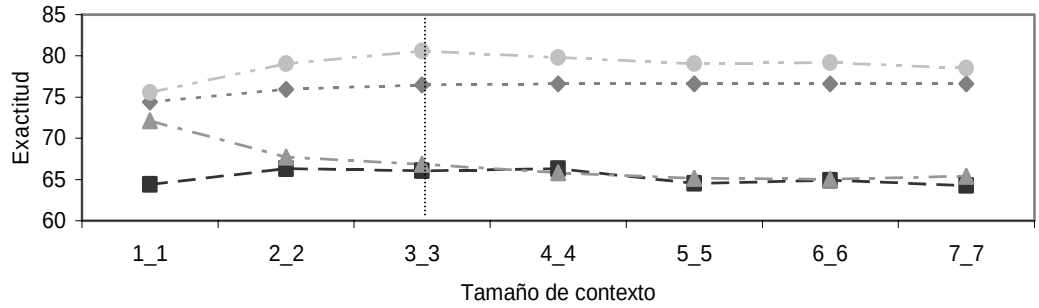
tal que ese tamaño de contexto es suficiente para contener el verbo, un elemento importante para la clasificación (ver tabla 5.6 para un ejemplo).



Entidad fecha



Entidad nombre



Entidad cantidad

Figura 5.3: Comparación de tamaños de contexto y algoritmos de aprendizaje

CONTEXTO DE TAMAÑO PEQUEÑO
unas,personas→PER_DAMNIFICADAS
de,personas→PER_MUERTAS
CONTEXTO DE TAMAÑO MEDIO
evacuacion,de,unas,personas,indico,el→PER_DAMNIFICADAS
ENT_NOMBRE,mas,de,personas,murieron,o→PER_MUERTAS
CONTEXTO DE TAMAÑO GRANDE
muertos,y,forzaron,la,evacuacion,de,unas,personas→PER_DAMNIFICADAS
de,personas,murieron,otras,fueron,dadas,como,desaparecidas→ PER_MUERTAS

Tabla 5.6: Ejemplo de diferentes tamaños de contexto

Para tener un análisis más amplio de los resultados, en la tabla 5.7 presentamos la evaluación del MUC para el sistema, donde tal evaluación se corresponde a la tarea original y más difícil de la extracción de información (i.e., la evaluación de plantillas de escenario), esto porque las tareas que involucran el análisis de módulos que profundizan el entendimiento del lenguaje no se aplican a la arquitectura propuesta (i.e., las evaluaciones de resolución de correferencia, plantillas de elementos y relación de plantillas no son posibles).

	REC	PRE	UND	OVG	ERR
EVE_FECHA	99	95	1	5	6
EVE_LUGAR	39	56	61	44	70
EVE_MAGNITUD	83	83	17	17	30
PER_MUERTAS	72	74	28	26	42
PER_HERIDAS	69	91	31	9	36
PER_DESAPARECIDAS	37	94	63	6	64
PER_DAMNIFICADAS	51	58	49	42	63
PER_AFECTADAS	32	75	68	25	71
VIV_DESTRUIDAS	48	76	52	24	59
VIV_AFECTADAS	43	87	57	13	59
INF_HECTAREAS	76	73	24	27	40
INF_ECONOMICA	29	60	71	40	76

Tabla 5.7: Evaluación de la tarea ST para el caso de estudio (dist. original)

Los resultados de la tabla anterior fueron obtenidos por dividir el corpus de entidades en conjunto de entrenamiento y conjunto de prueba, donde la extracción del lugar es una de las tareas con mayores errores en la extracción de información, un error que se puede estar propagando del componente de detección de segmentos de texto candidatos (i.e., se están analizando entidades que no representan un lugar). También, errores notables entre las cantidades que reportan daños a personas y viviendas están ocurriendo, los cuales se deben principalmente a la similitud de sus contextos (i.e., la información es extraída pero no se coloca en el registro correcto). Otro error importante es la extracción de la pérdida económica que dejó el desastre, sin embargo este error se lo atribuimos principalmente a que se cuentan con pocos ejemplos de entrenamiento para este tipo de información, por lo que se tiene una mala cobertura. En promedio, el resultado que se tiene del sistema es una cobertura del 61% y una precisión del 83%, situación que se refleja en una alta subgeneración (47%) y una poca sobregeneración (26%).

Debido al alto error en respuestas del experimento anterior (aprox. un 56%) se realizaron nuevos experimentos, en este caso modificando la distribución del conjunto de entrenamiento. Es decir, en nuestro experimento anterior conservamos la distribución del corpus para los ejemplos de entrenamiento, ahora evaluamos el comportamiento cuando la distribución es uniforme (i.e., igual número de ejemplos para todas las categorías). El resultado con respecto al error en respuestas fue el mismo (56%); sin embargo, se obtuvo información relevante con respecto a las otras medidas, éstas son: una cobertura del 97% y una precisión del 55%. En otras palabras, ocurrió que la subgeneración tuvo una disminución (12%) y la sobregeneración aumentó (53%). En consecuencia, se buscó encontrar un punto medio entre ambas distribuciones por realizar un promedio del número de ejemplos en cada categoría para los casos antes tratados. Por ejemplo, si del lugar donde ocurrió el evento se tomaron 190 ejemplos de entrenamiento en la distribución original y 582 para la distribución uniforme, entonces para una distribución media se tomaron (190+582)/2 ejemplos de entrenamiento. El resultado de esta nueva distribución se presenta en la tabla 5.8, donde el error de respuesta disminuyó a un

38%, teniendo una cobertura del 85% y una precisión del 69%, situación que provocó una subgeneración del 15% y una sobregeneración del 31%.

	REC	PRE	UND	OVG	ERR
EVE_FECHA	99	95	1	5	6
EVE_LUGAR	60	50	40	50	62
EVE_MAGNITUD	96	73	4	27	29
PER_MUERTAS	73	66	27	34	47
PER_HERIDAS	84	91	16	9	22
PER_DESAPARECIDAS	93	78	7	22	27
PER_DAMNIFICADAS	84	62	16	38	45
PER_AFECTADAS	82	60	18	40	47
VIV_DESTRUIDAS	83	70	17	30	39
VIV_AFECTADAS	83	72	17	28	38
INF_HECTAREAS	92	66	8	34	38
INF_ECONOMICA	95	49	5	51	52

Tabla 5.8: Evaluación de la tarea ST para el caso de estudio (dist. promedio)

Finalmente, con el objetivo de mejorar los resultados hasta ahora obtenidos, pero además comprobar la hipótesis planteada en la sección 4.1.2 (i.e., que la eliminación de palabras vacías y sufijos es recomendable para la clasificación de textos pero no para la extracción de información), se realizaron una serie de nuevos experimentos donde los contextos fueron preprocesados de acuerdo a lo hecho en la clasificación de textos. En resumen, los resultados confirmaron la hipótesis obteniendo una peor evaluación los clasificadores con contextos preprocesados.

Para ejemplificar la afirmación anterior, en la figura 5.4 se presenta la comparación de los resultados obtenidos en la clasificación de contextos de cantidades, en está gráfica se puede ver que la clasificación sin palabras vacías tiene su punto máximo antes que la clasificación con palabras y etiquetas, esto se debe a que cuando se eliminan términos (i.e., las palabras vacías), las palabras clave en la clasificación (e.g., los verbos) pueden alcanzarse en contextos más pequeños, sin embargo los términos omitidos ayudan a mejoran la discriminación, motivo por el cual la exactitud disminuye. Por otro lado, cuando clasificamos sin sufijos, el comportamiento es similar a cuando lo hacemos con palabras y etiquetas, pero como

se mencionó en la sección 4.1.2, estos elementos son indispensables en la extracción, y por lo tanto la evaluación empeora.

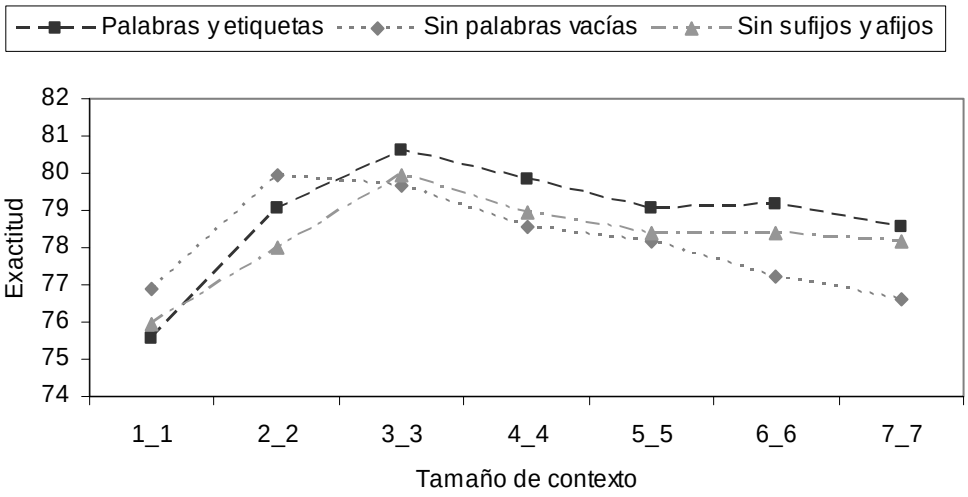


Figura 5.4: Comparación de métodos de preproceso para la extracción

5.3 Posibilidad Real ante el Caso de Estudio

Hasta el momento, ya se han contestado a las preguntas generales planteadas en la introducción de la tesis; además, también se han respondido a varias de las preguntas secundarias de la misma, motivo por el cual ya se tienen definidos los detalles técnicos del sistema de extracción de información para desastres naturales, al que denominamos *Topo*[11]. Sin embargo, un aspecto más que nos queda por aclarar es *¿qué posibilidades tiene esta propuesta de contribuir en el conocimiento del caso de estudio?*, i.e., tomando en cuenta el conocimiento que actualmente se tienen sobre desastres naturales, ¿qué posibilidades hay de que el sistema contribuya al mismo?.

Para tratar de aclarar la duda, aunque sea de forma parcial, se realizó una comparación de la información extraída manualmente por LA RED (i.e., su inventario de desastres DesInventar) y la encontrada con Topo. Para la comparación,

[11] En referencia al mamífero con manos especializadas para excavar y también en honor al grupo de auxilio surgido durante el terremoto que azotó a la ciudad de México en 1985.

se tomaron sólo en cuenta registros de desastres ocurridos en México durante el año de 1999, esto debido a que tenemos acceso a las noticias del Universal para ese año, y es de donde vamos a extraer la información con Topo. En la figura 5.5 se muestra una gráfica donde se compara la cantidad de registros existentes en DesInventar con los obtenidos por Topo, como se puede ver en la gráfica, los registros obtenidos por Topo (226) son más de los que hay en DesInventar (179), esto puede ser en parte por la redundancia en los medios de información y por lo tanto Topo genera información repetida. Otro punto importante en la gráfica es que a pesar de la diferencia en el número de registros, la distribución de eventos se conserva, excepto en el caso de incendio forestal, lo que es un reflejo del porqué esa categoría alcanzó la menor cobertura en las evaluaciones hechas al componente de filtrado de documentos.

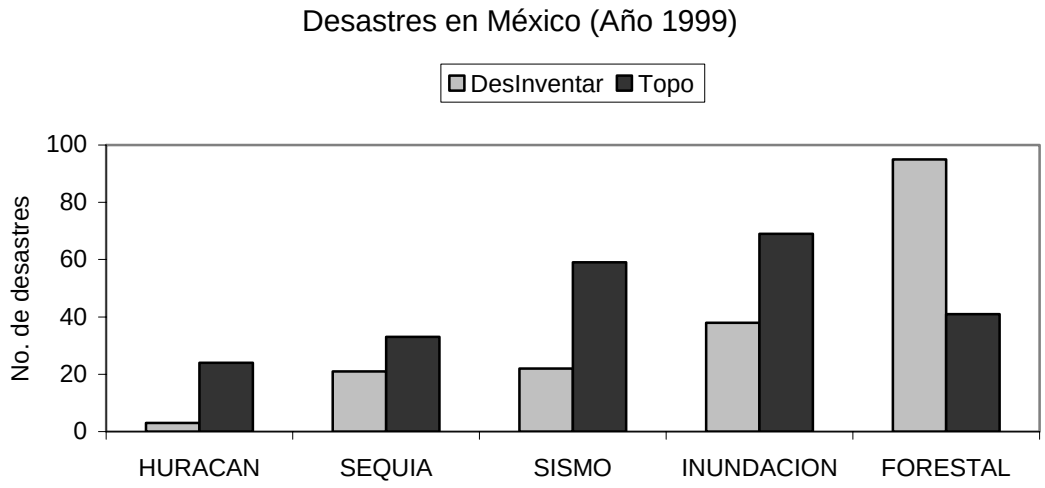


Figura 5.5: Comparación del número de registros en DesInventar y Topo

En el filtrado de información Topo presenta resultados aceptables, pero qué pasa en la extracción, i.e., la información que se obtiene ¿es veraz?. Para analizar lo anterior, a continuación comparamos un registro de DesInventar con los obtenidos por Topo para el mismo evento (ver tabla 5.9). El criterio tomado para fusionar los diferentes registros en Topo es elegir el valor más frecuente o más reciente sobre los

demás, y en el caso de no tener la certeza de qué valor tomar sólo se activa una casilla de verificación indicando la presencia o ausencia de información.

	DesInventar	**Topo**
FUENTE DE INFORMACIÓN		
FECHA	16/24-jun-99	16,20,30-Jun-99
NOMBRE	El Universal	El Universal
INFORMACIÓN DEL DESASTRE		
TIPO	Sismo	Sismo
FECHA	15 de junio	15 de junio
LUGAR	Puebla	Puebla
MAGNITUD		4.2
INFORMACIÓN DE LAS PERSONAS		
MUERTOS	11	11
HERIDOS	200	200
DESAPARECIDOS		
DAMNIFICADOS		
AFECTADOS		
INFORMACIÓN DE LAS VIVIENDAS		
DESTRUIDAS	120	36
AFECTADAS		
INFORMACIÓN DE LA INFRAESTRUCTURA		
HECTÁREAS		
PÉRDIDA ECONÓMICA	280,000,000	

Tabla 5.9: Comparación de un registro de DesInventar con uno de Topo

Como se puede ver en la tabla anterior, DesInventar y Topo difieren en tres de trece registros, una de estas diferencias es el número de viviendas destruidas, donde la cantidad en DesInventar es la correcta debido a que es más reciente que la extraída

por Topo. Con respecto a las otras dos diferencias, se tiene que un registró sí presenta información y el otro no, la pregunta que surge es ¿cuál es el correcto?. De primera instancia uno pensaría que DesInventar es el correcto; sin embargo, experimentando con el sistema que fue entrenado con ejemplos de distribución uniforme y que por lo tanto tiene una mayor cobertura (ver sección 5.2.2), se encontró que tanto DesInventar y Topo son correctos de forma parcial, i.e., si hubo un sismo de 4.2 grados (información extraída por Topo), pero el que provocó más daños fue el que ocurrió un poco antes de 6.7 grados; mientras que los 280 millones en pérdida económica eran válidos durante el mes de junio de 1999 (información extraída por LA RED), pero el 28 de julio del mismo año se reportó una pérdida por 400 millones. En conclusión, la extracción de información no sólo se ve afectada por la complejidad del dominio y la plantilla de extracción, sino que además por el transcurso del tiempo y la veracidad de la fuente de información.

Finalmente, cabe destacar que la base de datos de LA RED depende de terceras personas que de forma altruista proporcionan el registro de los eventos que ocurren, lo cual puede provocar que no se tenga el informe de todos los eventos, o peor aún, que la información no sea completa o exacta debido a que no se continúan rastreando los efectos de un desastre por largo tiempo. Por el otro lado, aunque Topo tampoco es completo y exacto, le tomó aproximadamente 25 minutos[12] generar los 226 registros relevantes a partir de las 33,654 noticias publicadas por el Universal en ese año, esto sin duda representa una aportación importante al estudio del dominio.

5.4 Discusión

En resumen, se obtuvo un sistema de extracción de información para documentos no estructurados con una exactitud de F_1 igual a 94.6% para filtrar documentos y un F_1 de 76% para la tarea ST en la extracción de información. Aunque por el dominio e idioma estudiado no es posible comparar estos resultados directamente con el estado actual de la tecnología, pueden observarse los valores

[12] En una computadora de escritorio con un procesador Pentium 4 a 2.40 GHz y 512 MB en RAM.

expuestos en las tablas 2.4 y 3.6 para tener una idea de lo relevante de la metodología aquí expuesta.

Sin embargo, a pesar de que se obtuvo la mejor evaluación en los experimentos donde se manipuló la distribución en el conjunto de entrenamiento, es preferible entrenar al sistema con un conjunto que conserve la distribución original. Esto porque aunque la información que se extrae es menor que en los otros casos (i.e., menor cobertura), ésta resulta ser más veraz (i.e., mayor precisión). Además, la falta de cobertura puede ser compensada por la redundancia en los medios, donde la información acerca de un evento se reporta más de una vez, ya sea en la misma noticia o en diferentes, y esta redundancia se incrementa si se analizan más de una fuente al mismo tiempo (e.g., tomar la información de varios periódicos).

Finalmente, en el capítulo también se realizó un análisis del sistema respecto a registros obtenidos de forma manual, donde se comprobó que el trabajo puede ser útil para incrementar el conocimiento del dominio, teniendo entre sus principales ventajas el continuar recolectando información de un evento aún cuando éste ya no sea vigente.

Capítulo 6

Conclusiones

Se presentó una arquitectura para desarrollar sistemas de extracción de información, así como una aplicación real implementada bajo este planteamiento, con esto creemos haber proporcionado las ideas para contestar a las dos preguntas de investigación generales planteadas en el capítulo 1 (i.e., *¿cómo construir sistemas de extracción de información sin utilizar un análisis lingüístico sofisticado?* y *¿cómo se comporta esta propuesta en la extracción de información?*). Además, también se realizó una comparación del sistema con un trabajo de extracción realizado de forma manual, donde se comprobó que la propuesta puede ser útil para incrementar el conocimiento de un dominio.

En resumen, la arquitectura aquí expuesta presenta una fuerte carencia de entendimiento del lenguaje reemplazado por métodos de aprendizaje supervisado, las ventajas y desventajas que esto implica son:

- **El uso de métodos empíricos mejoran la portabilidad entre dominios.** Los algoritmos de aprendizaje supervisado como base de la arquitectura proporcionan un ahorro de esfuerzo en mano de obra del experto para generar el conocimiento necesario, lo cual se traduce en una portabilidad entre dominios más eficiente.
- **Utilizar sólo un análisis a nivel léxico hace más factible la portabilidad entre idiomas.** Debido a que las técnicas empleadas en la arquitectura emplean únicamente características a nivel léxico de los textos para lograr su objetivo, la portabilidad de la misma a nuevos idiomas resulta más simple, esto por no requerir de herramientas que realicen un análisis lingüístico sofisticado.

- **Desventajas de la falta de entendimiento del lenguaje.** Estas desventajas tienen que ver principalmente con la falta de información del dominio y el nulo análisis del discurso, i.e., la información implícita no puede ser extraída y si se tienen registros con múltiples valores la información no se puede interpretar fácilmente (e.g., ver tabla 6.2)

	Texto	**Plantilla**
Información implícita	*Insuficiente, la ayuda a colimenses tras sismo,... Hoy se sabe que hay 25 mil viviendas dañadas que no han sido reparadas, además, la ayuda del Fondo de Desastres Naturales (Fonden) no llega.*	LUGAR DEL DESASTRE: ? VIVIENDAS AFECTADAS: *25 mil*
Información múltiple	*Al menos 11 muertos, cuatro desaparecidos y más de 4.500 refugiados en albergues dejaron los coletazos del huracán Keith en Centroamérica, ... En México, Guatemala, Belice, Costa Rica, Honduras, El Salvador y Panamá, las autoridades establecieron alertas de distinta consideración.*	LUGAR DEL DESASTRE: *México,Guatemala,Belice, Costa Rica,Honduras, Salvador,Panamá* PERSONAS MUERTAS: *11* PERSONAS DESAPARECIDAS: *cuatro* PERSONAS DAMNIFICADAS: *4.500*

Tabla 6.1: Desventajas de la arquitectura

Además, durante el trabajo de investigación se obtuvieron algunas conclusiones importantes, éstas son:

- **La inflexión de las palabras y los términos independientes del tema son importantes para la extracción de información.** Como se comprobó en los experimentos, las palabras vacías y sufijos son importantes para discriminar entre información relevante e irrelevante, caso contrarío ocurre cuando el objetivo es descubrir el tema de un documento.

- **La extracción de información es un proceso de largo plazo.** Como ya se había demostrado en las competencias del MUC, la extracción de información no es una tarea fácil, ni para las personas. En nuestros experimentos, además de la dificultad de la tarea, también comprobamos que para algunos casos se requiere que la fuente de información sea monitoreada por un largo periodo de tiempo, esto con el objetivo de obtener la información correcta. Por lo tanto, esta conclusión reafirma la importancia de automatizar la tarea.

6.1 Trabajo futuro

Tomando en cuenta las restricciones y desventajas que presenta la arquitectura, algunas ideas que siguen la filosofía de no entender el contenido del texto y que no han sido exploradas son:

- **Emplear técnicas de aprendizaje no supervisado.** Para implementar la arquitectura es necesario contar con un corpus de entrenamiento previamente clasificado y etiquetado. Una opción a esta restricción es evaluar técnicas de aprendizaje no supervisado para ayudar a generar el corpus necesario (para un estudio de estas técnicas ver [34]).
- **Realizar un análisis del discurso por fusionar plantillas.** En documentos donde se reporta información múltiple hay confusión con la información obtenida, una opción es llenar varias plantillas de forma parcial y posteriormente establecer criterios para su fusión. Esta idea también puede ser útil para fusionar plantillas extraídas de diferentes textos, pero que se refieren a la misma información.
- **Clasificar las entidades y no sólo detectarlas.** Los nombres son una de las entidades que más problemas presentan con respecto a la precisión, un identificador de segmentos de texto más especializado (i.e., que no sólo identifique los nombres propios, sino que además ayude a decidir si corresponde a un lugar, una persona, entre otros) puede mejorar los resultados actuales, evitando así el error que se propaga al tener que analizar en el

proceso de selección a entidades que no interesan al estudio (ver [44] para un ejemplo de la clasificación de entidades).

- **Elección de los ejemplos de entrenamiento en el corpus por emplear técnicas de agrupamiento de texto.** En el proceso de entrenar los algoritmos de aprendizaje, se tomó una parte del corpus del dominio como conjunto de entrenamiento, pero la decisión de qué elementos tomar fue hecha por utilizar un proceso de búsqueda aleatorio. Una opción que puede mejorar los resultados es tomar el conjunto de entrenamiento como una porción de los grupos obtenidos por aplicar técnicas de agrupamiento de textos basadas en aprendizaje automático no supervisado.
- **Evaluar nuevos métodos de indexado y aprendizaje.** El objetivo principal de la tesis fue definir una arquitectura de propósito general para construir sistemas de extracción de información, motivo por el cual algunos de los aspectos técnicos que dependen del dominio en estudio como son formas de indexado y métodos de aprendizaje supervisado no han sido aún evaluados. Ejemplos de métodos que proponemos evaluar son: utilizar el indexado semántico latente para la clasificación de textos, emplear la información de las partes de la oración para indexar contextos y probar con ensambles de clasificadores.
- **Evaluar la arquitectura en nuevos escenarios de extracción:** La extracción de información puede ser llevada a muchos escenarios, e.g., correos electrónicos [4], páginas Web, artículos científicos, entre otros.

Lista de Figuras

Lista de Tablas

Referencias

[1] K. Aas and L. Eikvil. Text categorization: A survey. *Technical Report*, Norwegian Computing Center, 1999.

[2] D. Appelt, J. Bear, J. Hobbs, D. Israel, and M. Tyson. Description of the JV-FASTUS system used for MUC-4. In *Proceedings of the 4th Message Understanding Conference*, 1992.

[3] D. Appelt and J. Israel. Introduction to information extraction technology. A *Tutorial Prepared for IJCAI-99*, 1999.

[4] J. Arrazola, A. Téllez, F. Zacarías, M. Alvarado, and G. Ramirez. Intelligent agents supported on both ASP and ML. *Journal of the Telecommunications and Informatics*, *Lecture Notes in Computer Science*, Springer-Verlag, 2004.

[5] R. Baeza and B. Ribeiro. *Modern Information Retrieval*. Addison Wesley, 1999.

[6] A. Bagga and A. Biermann. Analyzing the complexity of a domain with respect to an information extraction task. In *10th International Conference on Research on Computational Linguistics*, pp. 175-194, 1997.

[7] M. Califf. *Relational Learning Techniques for Natural Language Information Extraction*. Ph.d. thesis, University of Texas at Austin, 1998.

[8] N. Chinchor. MUC-7 test scores introduction. In *Proceedings of the 7th Message Understanding Conference*. Morgan Kaufmann, 1997.

[9] W. Cohen and H. Hirsh. Joins that generalize: text classification using WHIRL. In *Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining (KDD'98)*, AAAI Press, pp. 169 –173, 1998.

[10] W. Cohen and Y. Singer. Context-sensitive learning methods for text categorization. *ACM Transactions on Information Systems*, 17(2):141 –173, 1999.

[11] J. Cowie and W. Lehnert. Information extraction. *Communications of the ACM*, 39(1):80-91, 1996.

[12] S. Dumais, J. Platt, D. Heckerman, and M. Sahami. Inductive learning algorithms and representations for text categorization. In *Proceedings of the Seventh International Conference on Information Retrieval and Knowledge Management (ACM-CIKM'98)*, pp. 148-155, 1998.

[13] L. Eikvil, Information extraction from world wide web – A survey. *Technical Report*, Norwegian Computing Center, 1999.

[14] D. Freitag. *Machine Learning for Information Extraction in Informal Domains*. Ph.d. thesis, Computer Science Department, Carnegie Mellon University, 1998.

[15] D. Freitag and A. McCallum. Information extraction with HMM structures learned by stochastic optimization. In *Proceedings of the 21st National Conference on Artificial Intelligence*, 2000.

[16] R. Grishman. Where is the syntax?. In *Proceedings of the 6th Message Understanding Conference*, 1995.

[17] R. Grishman. Information extraction: Techniques and challenges. In MT. Pazienza, editor, *Proceedings of the Summer School on Information Extraction*, LNCS/LNAI. Springer-Verlag, 1997.

[18] M. Hearst, B. Schölkopf, S. Dumais, E. Osuna, and J. Platt. Trends and controversies - Support vector machines, *IEEE Intelligent systems*, pp. 18-28, 1998.

[19] J. Hobbs. The generic information extraction system. In *Proceedings of the 5th Message Understanding Conference*, Morgan Kaufmann, pp. 87-92, 1993.

[20] T. Joachims. Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of the Tenth European Conference on Machine Learning (ECML'98)*, Lecture Notes in Computer Science, Number 1398, pp. 137-142, 1998.

[21] T. Joachims. A statistical learning model of text classification with support vector machines. In *Proceedings of the 24th ACM International Conference on Research and Development in Information Retrieval,* ACM Press, 2001.

[22] A. Kehler, J. Bear, and D. Appelt. The need for accurate alignment in natural language System Evaluation. *Computational Linguistics*, 27(2):231-248, 2001.

[23] N. Kushmerick, E. Johnston, and S. McGuinness. Information extraction by text classification. In *Working Notes of the Adaptive Text Extraction and Mining Workshop in the Seventeenth International Join Conference on Artificial Intelligence*, pp. 44-50, 2001.

[24] A. Lavelli, M. Califf, F. Ciravegna, D. Freitag, C. Giuliano, N. Kushmerick and L. Romano. A critical survey of the methodology for IE evaluation. In *Proceedings of the 4th International Conference on Language Resources and Evaluation*, Lisbon, Portugal, 2004.

[25] W. Lehnert, C. Cardie, D. Fisher, E. Riloff, and R. Williams. Description of the CIRCUS System as Used for MUC-3. In *Proceedings of the 3rd Message Understanding Conference*, 1991.

[26] W. Lehnert, C. Cardie, D. Fisher, J. McCarthy, E. Riloff, and S. Sonderland. Evaluating an information extraction system. *Journal of Integrated Computer-Aided Engineering*, *Lecture Notes in Computer Science*, Springer-Verlag, 1(6), 1994.

[27] D. Lewis. Evaluating text categorization. In *Proceedings of the Speech and Natural Language Workshop*, Asilomar, CA, 1991.

[28] D. Lewis and M. Ringuette. A comparison of two learning algorithms for text classification. In *Proceedings of the 3rd Annual Symposium on Document Analysis and Information Retrieval*, pp. 81-93, 1994.

[29] D. Lewis. Naive (Bayes) at forty: The independence assumption in information retrieval. In *Proceedings of 10th European Conference on Machine Learning*, Springer Verlag, pp. 4-15, 1998.

[30] C. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing.* MIT Press, 1999.

[31] T. Mitchell. *Machine Learning*. McGraw-Hill, 1997.

[32] S. Muggleton and L. Raedt. Inductive logic programming: Theory and methods. *Journal of Logic Programming*, 19(20):629-679, 1994.

[33] I. Muslea. Extraction patterns for information extractions tasks: A survey. In *Proceedings of the National Conference on Artificial Intelligence Workshop on Machine Learning for Information Extraction*, 1999.

[34] B. Novak. Use of unlabeled data in supervised machine learning. In *SIKDD'2004 at multiconference IS 2004*, Ljubljana, Slovenia, 2004.

[35] F. Peng. Models development in IE tasks - A survey. *CS685 (Intelligent Computer Interface) course project*, Computer Science Department, University of Waterloo, 1999.

[36] J. Platt, Fast training of SVMs using sequential minimal optimization. In *Advances in Kernel Methods - Support Vector Learning*, B. Schölkopf, C. Burges, and A. Smola, editors., MIT Press, Cambridge, Mass., 1998.

[37] J.R. Quinlan. Induction of decision trees. *Machine Learning*, 1:81-106, 1986.

[38] J.R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[39] C. van Rijsbergen. *Information Retrieval*. Butterworths, London, 1979.

[40] E. Riloff. Automatically constructing a dictionary for information extraction task. In *Proceedings of the 11th National Conference on Artificial Intelligence*, pp. 811-816, 1993.

[41] D. Roth and W. Yih. Relational learning via propositional algorithms: An information extraction case study. In *Proceedings of the 15th International Conference on Artificial Intelligence*, 2001.

[42] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1-47, 2002.

[43] K. Seymore, A. McCallum, and R. Rosenfeld. Learning hidden markov model structure for information extraction. In *Proceedings of the 20th National Conference on Artificial Intelligence*, 1999.

[44] T. Solorio and A. López. Learning named entity classifiers using support vector machines. In Alexander Gelbukh, editor, *Computational Linguistics and Intelligent Text Processing: 5th International Conference, Lecture Notes in Computer Science*, Springer-Verlag, 2945:158–167, 2004.

[45] S. Soderland, D. Fisher, J. Aseltine, and W. Lehnert. CRYSTAL: Inducing a conceptual dictionary. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pp. 1314-1321, 1995.

[46] S. Sonderland. Learning information extraction rules for semi-structured and free text. *Machine Learning*, 34:233-272, 1999.

[47] J. Turno. Information extraction, multilinguality and portability. *Revista Iberoamericana de Inteligencia Artificial*. 22:57-78, 2003.

[48] M. Vilain. Inferential information extraction. In M.T. Pazienza, editor, *Information Extraction: Towards Scalability, Adaptable Systems. Lecture Notes in Artificial Intelligence*, Springer-Verlag, volume 1714, 1999.

[49] S. Weiss, C. Apté, F. Damerau, D. Johnson, F. Oles, T. Goetz, and T. Hampp. Maximizing text-mining performance. *IEEE Intelligent Systems*, 14(4):63-69, 1999.

[50] Y. Wilks and R. Catizone. Can we make information extraction more adaptive? *University of Sheffield, Computer Science Dept. Memoranda in Computer and Cognitive Science,* CS-00-02, 2000.

[51] Y. Wilks. Information extraction as a core language technology. *Lecture Notes in Computer Science*, number 1299, pp. 1-9, 1997.

[52] I. Witten and E. Frank. *Data Mining, Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, 2000.

[53] Y. Yang and J. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of the 14th International Conference on Machine Learning*, pp. 412-420, 1997.

[54] T. Yang and X. Liu. A re-examination of text categorization methods. In *Proceedings of SIGIR-99, 22nd ACM International Conference on Research and Development in Information Retrieval*, Berkeley, CA, pp. 42-49, 1999.

[55] J. Zavrel, P. Berck, and W. Lavrijssen. Information extraction by text classification: Corpus mining for features. In *Proceedings of the workshop Information Extraction meets Corpus Linguistics*, Athens, Greece, 2000.

Publicaciones y Reconocimientos

A. Téllez, M. Montes, and L. Villaseñor. A machine learning approach to information extraction. In Alexander Gelbukh, editor, *Computational Linguistics and Intelligent Text Processing: 6th International Conference, Lecture Notes in Computer Science*, Springer-Verlag, 2005. *(POR APARECER)*

A. Téllez, M. Montes, y L. Villaseñor. Aplicando la clasificación de texto en la extracción de información. En *memorias del Taller de Tecnologías del Lenguaje Humano, 5º. Congreso Internacional de Ciencias de la Computación*, pp. 259-266, 2004.

A. Téllez, M. Montes, y L. Villaseñor. Un sistema de extracción de información sobre desastres naturales. *Recientes avances en la ciencia de la computación en México*. Instituto Politécnico Nacional, Centro de Investigación en Computación, 7:89-98, 2004.

A. Téllez, M. Montes, y L. Villaseñor. Extracción de información con algoritmos de clasificación. En *memorias del 4º. Encuentro de Investigación INAOE*, pp. 253-256, 2003.

A. Téllez, M. Montes, O. Fuentes, y L. Villaseñor. Clasificación automática de textos de desastres naturales en México. En *memorias del 10º. Congreso Internacional de Investigación en Ciencias Computacionales*, pp. 269-263, 2003.

A. López, M. Montes, L. Villaseñor, M. Pérez, y A. Téllez. Gestión automática de información de desastres naturales en México. En *reporte de la 3ª Reunión Anual de RITOS2 y las IX Jornadas Iberoamericanas de Informática*, 2003.

Topo: Un ambiente para la extracción de información en el dominio de desastres naturales. Ficha descriptiva, Primer lugar en el concurso de desarrollo de software, 5º. Congreso Nacional de Computación, Instituto Politécnico Nacional, Centro de Investigación en Computación, 2004.