La teoría de autómatas y lenguajes formales, en la práctica

José A. Troyano, Víctor J. Díaz, Fernando Enríquez de S., Javier Barroso
Dept. de Lenguajes y Sistemas Informáticos
Universidad de Sevilla
41012 Sevilla
e-mail: troyano@lsi.us.es

Resumen

Se presenta una propuesta de trabajo práctico para una asignatura de la materia Teoría de Autómatas y Lenguajes Formales. Tradicionalmente este tipo de asignaturas suelen centrar su temario en aspectos teóricos y no se aprovecha el potencial práctico que supone la implementación y aplicación de los conceptos relacionados con la descripción de lenguajes y modelos de autómatas. En este trabajo se parte del modelo de máquina de Moore, para presentar el modelo de Markov como una extensión probabilística. Por último se propone como práctica de curso el desarrollo de un etiquetador de textos en lenguaje natural basado en modelos de Markov.

1. Introducción

Una de las quejas más habituales de los alumnos de asignaturas teóricas en titulaciones aplicadas como Ingeniería Informática es *¿para qué me va a servir lo que me estás explicando?* En muchas ocasiones este tipo de quejas son injustificadas ya que se deben a la falta de capacidad del alumno para hacer un análisis global de la materia y comprender las implicaciones que puede tener un determinado concepto teórico en un problema aparentemente lejano. Precisamente la tarea del alumno es adquirir esa capacidad y nuestro trabajo como profesor consiste en, además de presentar el concepto teórico, hacerle ver esas conexiones que lo hacen necesario para una aplicación posterior.

Sin embargo, hay veces en las que esta queja puede tener fundamento. Son los casos en los que los programas de las asignaturas son diseñados en base a una bibliografía clásica [4], [5] y pueden contener conceptos interesantes desde un punto de vista teórico pero que no están vinculados con ningún otro concepto posterior ni con una aplicación práctica. Las asignaturas relacionadas con la materia troncal Teoría de Autómatas y Lenguajes Formales, incluida en las vigentes directrices propias del título de Ingeniero en Informática, suelen adolecer en ocasiones de este problema. Es habitual encontrarse en los temarios de estas asignaturas con algunos conceptos que, o bien son difícilmente aplicables en la práctica, o bien sí lo son pero dichas aplicaciones no son incluidas en el temario por considerarse que *no pegan con una asignatura teórica*.

En este trabajo nos fijaremos en uno de esos conceptos, la máquina de Moore, un modelo cercano al de los autómatas finitos y a su vez muy cercano a otro tipo de autómata, el de Markov, que puede dar bastante juego a la hora de plantear una práctica de curso muy interesante.

Sin embargo, nuestra motivación va un poco más allá de presentar una práctica más o menos interesante, nuestro verdadero interés se centra en un debate de mayor calado: cómo conseguir introducir en una asignatura de corte teórico, como es la de Lenguajes Formales y Autómatas, aspectos prácticos que complementen los conceptos teóricos. Consideramos que este planteamiento es de vital importancia en una titulación tan aplicada como lo es la Ingeniería Informática.

2. La máquina de Moore

La máquina de Moore es un modelo derivado de los autómatas finitos. Este tipo de autómata es el

modelo computacional más simple de los que se utilizan en el procesamiento de lenguajes.

2.1. Definición de autómata finito

Formalmente, un autómata finito determinista se define como la tupla (Q,Σ,δ,q_0,F) donde Q es un conjunto finito de estados, Σ es un conjunto finito de símbolos denominado alfabeto de entrada, $\delta: Q\times\Sigma\rightarrow Q$ es la función de transición, q_0 es el estado inicial y $F\subseteq Q$ es el conjunto de estados finales. La representación más usada a efectos de especificación para los autómatas finitos suele ser la gráfica (figura 1).

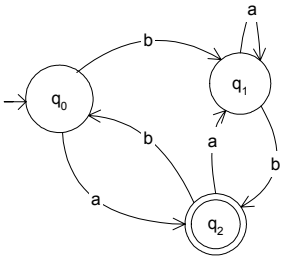

Figura 1: Autómata finito

La aplicación más común de los autómatas finitos en el ámbito del procesamiento de lenguajes es la implementación de reconocedores léxicos para lenguajes formales. Existen multitud de herramientas que permiten especificar un analizador léxico combinando expresiones regulares e instrucciones de un lenguaje de programación, y que dan como resultado una implementación de dicho reconocedor basada en autómatas finitos. Igualmente en la bibliografía se pueden encontrar distintos métodos de obtención de autómatas a partir de expresiones, que van desde los más abstractos planteados en forma de demostraciones de teoremas hasta los más aplicados expresados en forma de algoritmos, que incluso incluyen decisiones de diseño orientadas a reducir el coste computacional del proceso.

2.2. Definición de máquina de Moore

La máquina de Moore es un modelo muy cercano al de los autómatas finitos. De hecho tan solo se diferencia de un autómata en la capacidad que tiene de emitir una salida asociada al estado del autómata.

Formalmente una máquina de Moore se define como la tupla $(Q,\Sigma,\Delta,\delta,\gamma,q_0)$ donde Q, Σ, δ y q_0 significan lo mismo que en el modelo de autómatas finitos, Δ es un conjunto finito de símbolos denominado alfabeto de salida y $\gamma: Q\times\Sigma\rightarrow\Delta$ es la función de salida que calcula el símbolo emitido en cada estado. Al igual que los autómatas finitos, la representación gráfica es la más utilizada a la hora de describir las máquinas de Moore (figura 2).

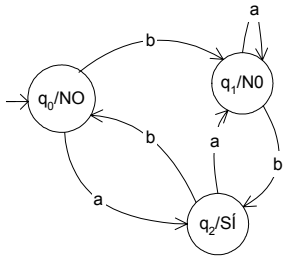

Figura 2: Máquina de Moore

En términos de descripción de lenguajes, los autómatas de las figuras 1 y 2 son equivalentes ya que ambos aceptan las secuencias de letras que conducen al estado q_2, Este ejemplo basta para mostrar que las máquinas de Moore son al menos tan potentes como los autómatas finitos. Es fácil ver que con el concepto de salida asociada al estado no sólo se pueden modelar los conceptos *final* y *no final* de los autómatas finitos sino que se pueden describir situaciones más complejas, ya que el alfabeto de salida puede ser tan amplio como se desee.

La máquina de Moore suele ser presentada en compañía de la máquina de Mealy en la que la salida se asocia a las transiciones del autómata en lugar de a los estados. Existen demostraciones de la equivalencia de ambos modelos, y se suele denominar máquina secuencial al modelo general que subyace tras ellas.

Una aplicación muy usual de las máquinas secuenciales es la de diseño de circuitos electrónicos, concretamente en situaciones en las que la simplicidad del problema a resolver hace

que no sea necesario incluir un microprocesador en el diseño del circuito.

2.3 La máquina de Moore y el procesamiento de lenguajes

La filosofía con la que están planteadas la mayoría de las asignaturas dedicadas a la Teoría de Autómatas y Lenguajes Formales se basa en mostrar la relación que existe entre formalismos de especificación de lenguajes (gramáticas y expresiones regulares) con sus correspondientes modelos de computación. Esta relación teórica es totalmente explotada en las asignaturas correspondientes a la materia troncal Procesadores de Lenguaje, en las que se aplican estos conceptos y se integran en una metodología de desarrollo.

Si es éste el planteamiento, como ocurre en nuestro caso, resulta bastante difícil justificar la inclusión de un tema dedicado a máquinas secuenciales en el temario de la asignaturas de la materia troncal Teoría de Autómatas y Lenguajes Formales (TALF), máxime si tenemos en cuenta que puede que se repitan los contenidos de las asignaturas dedicadas al diseño de circuitos.

Evidentemente, es de gran interés teórico mostrar la generalización que suponen las máquinas de Moore y de Mealy frente al modelo básico de los autómatas finitos, pero esa presentación sería didácticamente más convincente si se acompaña de una aplicación práctica. En este trabajo mostramos una aplicación de una variante de la máquina de Moore al procesamiento de textos en lenguaje natural. Esta aplicación, además de permitir diseñar una práctica de laboratorio interesante, proporciona al alumno una visión alternativa al procesamiento de lenguajes formales a través del tratamiento estadístico del lenguaje natural.

3. El modelo de Markov

El modelo de Markov se acerca mucho al modelo de la máquina secuencial de Moore, también se puede interpretar como una generalización de un autómata finito en la que cada estado tiene asociada una salida. La diferencia fundamental con el modelo de Moore está en la introducción de probabilidades tanto para las transiciones como para la emisión de la salida correspondiente a un estado. De manera que en términos simplistas podríamos decir que un modelo de Markov es una máquina de Moore probabilística.

Formalmente un modelo de Markov μ se define como la tupla (Q, Δ, π, A, B) donde:

- $Q=\{1,...,N\}$ es el conjunto de estados del modelo. Etiquetaremos los estados con números naturales y denotaremos con q_t el estado en el instante de tiempo t.
- $V=\{v_1,v_2,...,v_M\}$ es el conjunto de las posibles salidas o sucesos observables del modelo. Cada estado podrá emitir varias de estas salidas con una determinada probabilidad.
- $\pi=\{\pi_1,\pi_2,...,\pi_N\}$ es la distribución de probabilidad del estado inicial de forma que $\pi_i=P(q_1=i)$. Se cumple:
$$\Sigma_i \pi_i = 1$$
- $A=\{a_{11},...,a_{ij},...,a_{NN}\}$ es la distribución de probabilidad de las transiciones entre estados, de forma que:
$$a_{ij}=P(q_t=j|q_{t-1}=i)$$
Es decir la probabilidad de transitar al estado j condicionada a estar en el estado i en el instante de tiempo anterior. Para simplificar se denotará también con $P(j|i)$. Se cumple:
$$\Sigma_j P(j|i) = 1$$
- $B=\{b_{11},...,b_{kj},...,b_{NM}\}$ es la distribución de probabilidad de los sucesos observables, de forma que:
$$b_{kj}=P(o_t=v_k|q_t=j)$$
Es decir la probabilidad de emitir la salida v_k condicionada a estar en el estado j. Para simplificar se denotará también con $P(v_k|j)$. Se cumple:
$$\Sigma_k P(v_k|j) = 1$$

Pese a su aparentemente compleja definición, el modelo de Markov no es más que el modelo de Moore en el que los conceptos de estado inicial, transición etiquetada y salida se sustituyen, respectivamente, por probabilidad inicial, probabilidad de transición y probabilidad de emisión de salida.

Originalmente el modelo de Markov fue diseñado por Andrei A. Markov para construir un modelo de las secuencias de letras de las palabras en la literatura rusa. Pero posteriormente se adoptó como herramienta estadística de propósito general para describir procesos estocásticos.

Podemos adaptar la representación gráfica de los autómatas finitos y la máquina de Moore para

dar cabida a las probabilidades del modelo de Markov tal y como se muestra en la figura 3. Dicha figura representa un modelo muy simplificado de la evolución del tiempo atmosférico. Los estados denotan tres posibles situaciones: tiempo *bueno*, *malo* e *inestable*, y la salida observable son los fenómenos atmosféricos más probables en dichas situaciones: *sol*, *nubes* y *lluvia*.

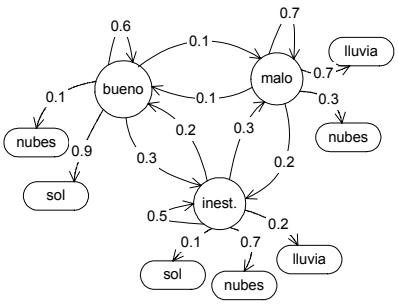

Figura 3: Modelo de Markov

3.1. Propiedades del modelo de Markov

Desde el punto de vista de la descripción de procesos estocásticos, el de Markov es un modelo realmente simple. De hecho, en la definición del modelo que hemos presentado hay implícitas dos simplificaciones o propiedades que marcan notablemente la filosofía con la que se debe utilizar este modelo. Estas dos características son:

- Propiedad del horizonte limitado: esta propiedad permite limitar la dependencia del estado actual con respecto a los anteriores, de manera que sólo se tengan en cuenta los n últimos de los estados anteriores. En los modelos de Markov de orden uno se asume que la dependencia se extiende sólo al estado anterior, por tanto:
  $P(q_t=j|q_{t-1}=i,q_{t-2}=k,...)= P(q_t=j|q_{t-1}=i)$
- Propiedad del tiempo estacionario: esta propiedad establece que la probabilidad de transitar de un estado a otro es la misma independientemente del instante de tiempo. Se expresa:
  $P(q_1=j|q_0=i)=...= P(q_t=j|q_{t-1}=i)=P(j|i)$

Obviamente, estas propiedades dan lugar a modelos que se aproximan a la realidad pero que no la formalizan por completo. En este sentido, la limitación más importante de los modelos de Markov es la incapacidad de modelar adecuadamente relaciones de larga distancia.

4. Procesando textos en lenguaje natural

Los modelos de Markov se han hecho muy populares en el campo del procesamiento del lenguaje natural [6]. Su simplicidad hace que los procesadores obtenidos sean bastante eficientes, y su carácter probabilístico permite que los modelos se aprendan de manera automática a partir de conjuntos de ejemplos.

En general los modelos de Markov se han aplicado con bastante éxito a todos aquellos problemas en los que se necesite asociar una etiqueta (o una categoría) a una palabra en el texto. En términos del modelo de Markov, la palabra sería la observación y la etiqueta que se le asocia sería el estado.

En este trabajo presentaremos la aplicación del modelo de Markov al problema del etiquetado gramatical. Con pequeñas modificaciones se puede adaptar el ejemplo a muy diversos problemas, en el último apartado de esta sección describiremos estos problemas que pueden dar pie a prácticas similares.

4.1. Etiquetado gramatical

El etiquetado gramatical consiste en asociar a cada palabra la categoría gramatical a la que pertenece, esta tarea suele ser una de las primeras etapas en cualquier sistema de procesamiento de textos. La mayor dificultad de este problema viene provocada por la ambigüedad que presentan numerosas palabras, que pueden desempeñar a diferentes funciones gramaticales. Por ejemplo la palabra *casa* puede ser nombre y verbo, desempeñará una función u otra dependiendo de las palabras que la rodeen, es decir su contexto. Esta ambigüedad hace que la solución al etiquetado gramatical no pueda hacerse simplemente consultando un diccionario electrónico y haya que acudir a soluciones más complejas que hagan uso de la información que proporciona el contexto de cada palabra.

Salida	El	perro	bebe	agua
Estado	DET	NOM	VER	NOM

Figura 4: Etiquetado gramatical

Una de estas soluciones pasa por construir un modelo de Markov en el que las palabras son las salidas y las categorías son los estados. La figura 4 muestra un ejemplo de etiquetado gramatical en estos términos, las etiquetas DET, NOM, VER se refieren a las categorías determinante, nombre y verbo respectivamente.

Las probabilidades de transición entre estados modelarán las secuencias de etiquetas posibles, mientras que las probabilidades de emisión de símbolos modelarán qué palabras pueden desempeñar una determinada función gramatical.

En la figura 5 se muestra un fragmento de un modelo de Markov que refleja las relaciones entre categorías gramaticales y palabras del ejemplo anterior.

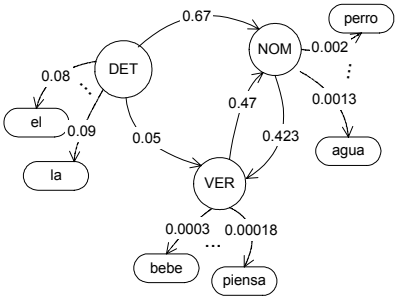

Figura 5: Fragmento del modelo de Markov

4.2. Obtención y aplicación del modelo

Con lo visto hasta ahora sabemos cómo modelar los distintos aspectos de un problema mediante un modelo de Markov, pero aún faltan por concretar dos cuestiones muy importantes, cómo construir y cómo aplicar un modelo de Markov. En el problema concreto que estamos tratando esto se traduce en dar una respuesta algorítmica a las siguientes preguntas:

- ¿Cómo obtener un modelo de Markov que se ajuste a las relaciones reales entre etiquetas gramaticales y palabras?
- Dada una frase sin etiquetar ¿cuál es la etiqueta más probable para cada palabra de la frase según el modelo de Markov?

Para responder a la primera pregunta hay que estimar las probabilidades de transición entre estados $P(i|j)$ y las de emisión de palabras $P(v_k|j)$. Para ello se hace uso de un corpus etiquetado, que no es más que un conjunto de textos en el que cada palabra tiene asignada la etiqueta apropiada. En base a la frecuencias de etiquetas consecutivas y las frecuencias de asignación de etiquetas a palabras en los ejemplos del corpus, las probabilidades anteriores se calculan fácilmente con las siguientes fórmulas:

$$P(i \mid j) = \frac{C(j,i)}{C(j)}$$

$$P(v_k \mid j) = \frac{C(v_k, j)}{C(j)}$$

Donde $C(j,i)$ es el número de veces que la etiqueta i aparece tras la etiqueta j, $C(j)$ es el número de veces que aparece la etiqueta j, y $C(v_k,j)$ es el número de veces que a la palabra v_k se le asigna la etiqueta j.

Se suele denominar entrenamiento al proceso de obtención del modelo a partir de los ejemplos contenidos en el corpus. Uno de los parámetros que marcan la calidad de una estimación es el tamaño del corpus, cuantos más ejemplos tengamos más fiables serán las probabilidades estimadas.

La segunda de las preguntas es la que nos permite solucionar el problema original, es decir determinar cuál es la categoría gramatical más probable de cada palabra de una frase. Aplicando ciertas transformaciones y simplificaciones se demuestra que el mejor etiquetado (el más probable) es aquel que maximiza la siguiente expresión:

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

En dicha expresión el índice i indica la posición de cada palabra dentro de la frase (en la definición original del modelo se corresponde con el instante de tiempo i), t_i y t_{i-1} son respectivamente las etiquetas asignadas a las palabras que ocupan la posición i e $i-1$ en la frase y w_i es la palabra que ocupa la posición i.

Para calcular el mejor etiquetado basta con generarlos todos y quedarse con aquel que maximice la fórmula anterior. Utilizando técnicas de programación dinámica, el algoritmo de Viterbi [6] consigue el mismo resultado sin tener que generar todas las posibilidades, lo que redunda de forma significativa en la eficiencia del proceso.

4.3. Otras aplicaciones

Estas ideas se han aplicado con éxito a muchos problemas de procesamiento de textos en lenguaje natural. En general cualquier problema en el que haya que asignar una etiqueta a una palabra o a un conjunto de palabras se puede adaptar para expresarlo en términos de un modelo de Markov. Algunos de ellos son:

- Reconocimiento de entidades con nombre: consiste en la identificación de nombres de personas, lugares, organizaciones.
- Análisis sintáctico superficial: Consiste en identificar estructuras sintácticas en una frase sin realizar el análisis sintáctico completo.
- Desambiguación de significados: consiste en determinar el significado apropiado de una palabra polisémica.
- Extracción de información: consiste en identificar los fragmentos más informativos de un texto conociendo de antemano el tema sobre el que trata.

En general estos problemas son complejos, no obstante con las debidas simplificaciones todos pueden ser apropiados para una práctica similar a la presentada en este trabajo.

5. Planteamiento de la práctica

Ya hemos presentado los conceptos necesarios para plantear la práctica, tan sólo nos falta decidir cómo presentar dichos conceptos a los alumnos.

En primer lugar hay una serie de contenidos que son más apropiados para una clase de tipo teórico, en general todos aquellos que tengan que ver con la definición formal del modelo. No serán conceptos difíciles de presentar ya que a base de pequeñas ampliaciones se puede llegar del modelo básico de autómatas finitos, bien conocido por los alumnos, hasta el modelo de Markov pasando a través de la máquina de Moore tal y como hemos hecho en la presentación.

Tomando, por tanto, como punto de partida el conocimiento teórico por parte de los alumnos del modelo de Markov y de los algoritmos de entrenamiento (estimación de las probabilidades del modelo) y aplicación (Viterbi), lo que queda por presentar es la arquitectura del sistema a implementar y los recursos y herramientas que van a utilizarse.

5.1. La arquitectura

La figura 6 muestra los distintos elementos que constituyen el sistema a implementar. Cabe destacar que tanto el proceso de entrenamiento como el de aplicación contienen un componente importante de tratamiento de textos ya que todas sus entradas y salidas serán ficheros de texto. Esto hace que la práctica no sea sólo interesante por aplicar conceptos vistos en teoría, sino también por el tipo de herramientas que habrá que utilizar para desarrollarla, que deberán incluir aspectos como expresiones regulares que faciliten el tratamiento de estos textos. De esta forma tanto a través del objeto de aplicación como de los medios de implementación se aplican conceptos relacionados con la teoría de autómatas y lenguajes formales.

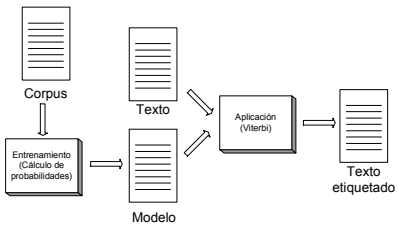

Figura 6: Arquitectura

Una vez propuesta la arquitectura del sistema el siguiente paso será decidir qué recursos y herramientas son necesarios para realizarla. La presentación de la práctica contendrá una propuesta de implementación, no obstante el problema es lo suficientemente abierto y flexible como para invitar a los propios alumnos a que investiguen y busquen otras alternativas. Con unas breves indicaciones y la curiosidad innata en algunos alumnos, en Internet se pueden encontrar con facilidad este tipo de recursos.

5.2 Recursos y herramientas

Los alumnos dispondrán de los siguientes recursos para el desarrollo de la práctica:

- Corpus etiquetado
- Implementación del algoritmo de Viterbi

El primero es imprescindible ya que para obtener unos resultados un poco satisfactorios se requiere un volumen de datos de entrenamiento considerable, lo que hace inviable que el alumno pueda construirse su propio corpus. Este tipo de recursos se pueden encontrar en Internet [7], se proporcionará uno junto con el enunciado, y se invitará a los alumnos a que busquen otras alternativas por su cuenta.

El algoritmo de Viterbi no es muy complejo y es viable que el alumno lo desarrolle por su cuenta a partir de las explicaciones en clase. No obstante existen diversas implementaciones genéricas de los modelos de Markov que ya incorporan dicho algoritmo y evitan ese trabajo [3]. En este sentido consideramos más interesante que el alumno realice el trabajo de transformación de textos para adaptar su entrada al formato que exija una determinada implementación del modelo de Markov, que hacerle desarrollar un algoritmo que desde el punto de vista práctico está más cercano a la asignatura *Estructuras de Datos y Algoritmos*.

La última decisión que queda por tomar es el lenguaje de programación. Dada la importancia de la componente textual, Perl es una magnífica opción debido a su potencia en el manejo de cadenas y expresiones regulares.

Cabe puntualizar que los resultados obtenidos por los alumnos al desarrollar esta herramienta no pasarán de ser un prototipo muy elemental. Las herramientas reales hacen uso de técnicas más complejas y refinadas para obtener resultados más satisfactorios. Esto no resta nada al interés didáctico de la práctica, es más el conocimiento de estas limitaciones por parte de los alumnos puede incitar a aquellos más interesados en el tema a continuar investigando en busca de mejoras al modelo básico.

6. Contexto académico y planificación temporal

En esta sección completaremos la presentación de esta propuesta docente enmarcándola en el contexto de la titulación de Ingeniería Informática de la Universidad de Sevilla [2] y proponiendo una planificación temporal para el desarrollo de la práctica a lo largo de un cuatrimestre.

6.1. Las asignaturas de la materia troncal TALF en Sevilla

El título de Ingeniero en Informática dedica dos asignaturas al desarrollo de los conceptos relacionados con la Teoría de Autómatas y Lenguajes Formales, una en segundo curso denominada *Lenguajes Formales y Autómatas* (LFA) y otra en tercero llamada *Ampliación de Lenguajes Formales y Autómatas* (ALFA), ambas de 4,5 créditos. La práctica presentada en este trabajo está diseñada para que se desarrolle en las sesiones de laboratorio de la asignatura ALFA, ya que requiere unos conocimientos mínimos de procesamiento de lenguajes que el alumno habrá adquirido en la asignatura LFA.

La asignatura LFA está dedicada a la presentación de los conceptos básicos de la Teoría de Lenguajes Formales. En ella el alumno tomará contacto con la idea de lenguaje y con los formalismos generativos y de reconocimiento asociados a las dos familias más simples de lenguajes de la jerarquía de Chomsky, los lenguajes regulares y los independientes del contexto. Desde el punto de vista práctico, se utilizarán herramientas clásicas de procesamiento de lenguajes (flex, bison) que aplican directamente expresiones regulares para especificar los aspectos léxicos de un lenguaje y gramáticas independientes del contexto para los aspectos sintácticos.

En el aspecto teórico, la asignatura ALFA se centra en la ampliación de los conceptos vistos en LFA abarcando los temas de máquinas secuenciales y probabilísticas [1], límites teóricos de lenguajes regulares e independientes del contexto, lenguajes dependientes del contexto y lenguajes con estructura de frases.

La práctica propuesta para la asignatura ALFA cubre varios objetivos al mismo tiempo. Por un lado implementa uno de los aspectos vistos en teoría ya que el modelo de Markov se puede considerar una extensión de los autómatas probabilísticos. En segundo lugar presenta al alumno una aplicación que trata textos en lenguaje natural, de una naturaleza totalmente distinta al

tipo de problemas que resolvió en la asignatura LFA, centrada en el tratamiento de lenguajes formales. Por último las herramientas que se utilizan en la práctica, en concreto el lenguaje Perl, proporcionan al alumno una alternativa de desarrollo de procesadores de lenguajes que para ciertas aplicaciones puede resultar más adecuada que las herramientas clásicas basadas en expresiones regulares y gramáticas independientes del contexto.

6.2. Planificación temporal

La práctica está pensada para que se desarrolle a durante un cuatrimestre. Con idea de planificar el trabajo de los alumnos se dividirá en siete sesiones de laboratorio, de las cuales las tres primeras se dedicarán a presentar el lenguaje Perl, nuevo para ellos, y las cuatro últimas a desarrollar distintos módulos de la práctica. En concreto las sesiones son:

- Introducción al lenguaje Perl
- Expresiones regulares en Perl
- Tablas asociativas en Perl
- Cálculo de frecuencias de etiquetas y palabras a partir del corpus
- Ejecución y prueba de la implementación del modelo de Markov disponible
- Obtención del modelo de Markov a partir de las frecuencias
- Aplicación del modelo a un texto sin etiquetar

Evidentemente algunos de los problemas propuestos son de una cierta envergadura y será muy difícil completar su implementación en una sola sesión. El montaje y prueba final de la aplicación queda como trabajo individual del alumno que deberá ser presentado como práctica final de la asignatura.

7. Conclusiones

Se ha presentado una práctica de curso para una asignatura de Teoría de Autómatas y Lenguajes Formales. Las conclusiones más destacables de esta propuesta son las siguientes:

- La introducción de aspectos muy aplicados en el ámbito de una asignatura de enfoques tradicionalmente teóricos.
- La aplicación de conceptos avanzados sobre autómatas a un problema de procesamiento de lenguajes.
- La presentación al alumno de una aplicación que procesa textos en lenguaje natural, lo que le proporciona una visión alternativa al procesamiento clásico de lenguajes formales.
- La utilización de un lenguaje de programación como Perl que integra mecanismos de manejo de textos de una forma sensiblemente distinta a las herramientas usadas por el alumno en cursos anteriores.

Referencias

[1] M. Alfonseca, J. Sancho, M. Martínez. *Teoría de Lenguajes, Gramáticas y Autómatas.* Ediciones Universidad y Cultura. Madrid. 1987.

[2] Boletín Oficial del Estado. *Plan de Estudios de Ingeniero en Informática de la Universidad de Sevilla*. 18 de noviembre de 1997.

[3]T. Brants. *TnT an statistical Part-of-Speech Tagger*. http://www.coli.uni-sb.de/~thorsten/tnt/

[4] D. I. A. Cohen. *Introduction to Computer Theory*. John Wiley & Sons. New York. 1991.

[5] J. E. Hopcroft, R. Motwani, J. D. Ullman. *Introducción a la Teoría de Autómatas, Lenguajes y Computación*. Addison-Wesley, 2002.

[6]C. Mannig, H. Schütze. *Foundations of Statistical Natural Language Processing*, MIT Press. Cambridge. 1999.

[7]G. Sampson. *Downloadable research resources, Corpus Sussane*. www.grsampson.net