Universitat de les Illes Balears

Modelos ocultos de Markov para el etiquetado de texto

Trabajo final de máster entregado a la Universitat de les Illes Balears de acuerdo con los requisitos del **Màster Universitari en Anàlisi de Dades Massives en Economia i Empresa (MADM)**

Autor
Antonio Crespí Tobeña

Tutor
Arnau Mir Torres

18 de setiembre de 2018

Modelos ocultos de Markov para el etiquetado de texto

Antonio Crespí Tobeña
Tutor: Arnau Mir Torres
Trabajo de final de Màster Universitari en Anàlisi de Dades Massives en Economia i Empresa (MADM)
Universitat de les Illes Balears
07122 Palma de Mallorca

Resumen

En este trabajo se presenta una implementación de modelos ocultos de Markov para el etiquetado de texto y se investiga como les afectan distintos parámetros y tipos de texto de entrenamiento.
En concreto, se prueban varias formas de tratar palabras desconocidas y distintas temáticas y tamaños del texto de entrenamiento para ver como se puede obtener un modelo óptimo enfocado a una tarea concreta.

Abstract

We present an implementation of a Hidden Markov Model for part-of-speech tagging and test different training corpora and parameters to see how its behaviour is affected.
In particular, we test different approaches of dealing with unknown words, different *corpus* sizes and different *corpus* categories to study how an optimal model has to be trained for a specific task.

Palabras clave: Part-of-speech tagging, Hidden Markov Models

1. Introducción

La idea de proporcionar a los ordenadores la habilidad de procesar el lenguaje humano es tan antigua como los ordenadores en sí mismos. Existen muchos campos de las matemáticas y la computación dedicados a la investigación de esta área, como el procesamiento del lenguaje natural, la tecnología del lenguaje humano, la computación neurolingüística y el reconocimiento y síntesis de voz. El objetivo de estos campos es permitir a los ordenadores realizar tareas que incluyen el lenguaje humano, permitiendo la comunicación entre humanos y máquinas, mejorando la comunicación entre humanos o permitiendo el procesamiento automático de texto y voz[10].

Varios ejemplos donde se aplican estos campos son los asistentes de voz –como *Siri* o *Google Assistant*– o la traducción automática de textos. Para que todo esto sea posible, uno de los principales temas a tratar es la ambigüedad, ya que a partir de únicamente una palabra no es posible obtener su significado preciso. Por ejemplo, la palabra *fish* en la oración '*I fish a fish*' aparece dos significados distintos. Sin aplicar ningún tratamiento al texto y sin información sobre el contexto, ambas palabras son indistinguibles.

El etiquetado de texto trata de resolver este problema asignando una categoría semántica –también llamada *POS-tag*, del inglés *Part-of-speech tag*– a cada palabra. Usando esta técnica, la frase anterior (*I fish a fish*) pasa a ser '*I/PRON fish/VERB a/DET fish/NOUN*', permitiendo diferenciar los homónimos. Si estamos construyendo un traductor automático, podemos basarnos en esta distinción para traducir '*fish/VERB*' como 'pescar' y '*fish/NOUN*', como 'pez'.

En este trabajo se describe e implementa un algoritmo para el etiquetado de texto usando modelos ocultos de Markov y se mide su desempeño según varios parámetros. Para ello, primero se detallan los modelos ocultos de Markov y como se adaptan al etiquetado de texto. Después, se realizan varias pruebas comparando distintos textos de entrenamiento y valores de los parámetros. Finalmente, se elaboran unas conclusiones sobre los resultados obtenidos.

2. Base teórica

2.1. Cadenas de Markov

Las cadenas de Markov son extensiones del autómata finito determinista. En concreto, son un tipo de autómata en el que las transiciones representan las probabilidades de ir de un estado a otro y los estados que se visitan dependen únicamente de la cadena de entrada[10]. Formalmente, podemos definir una cadena de Markov como:

$Q = q_1 q_2 ... q_N$	Conjunto de N estados.
$A = a_{00} ... a_{nn}$	Matriz de probabilidades de transición, donde cada casilla a_{ij} representa la probabilidad de ir del estado i al estado j. La suma de todos los arcos que salen de un estado i debe ser igual a 1.
q_0, q_F	Estados artificiales que se añaden al inicio y final de la cadena de entrada, respectivamente.

Además, se asume que se cumple la propiedad de que la probabilidad de estar en un estado i únicamente depende del estado anterior $i-1$. A esta propiedad se la conoce como **propiedad de Markov**:

$$P(q_i|q_0..q_{i-1}) = P(q_i|q_{i-1}) \tag{1}$$

Las cadenas de Markov resultan útiles cuando se necesita calcular la probabilidad de una secuencia de eventos observables. Tomemos por ejemplo una oficina en la que alguien se encarga de anotar la carga de trabajo diaria –para simplificar, vamos a sumir que la carga de trabajo solo puede ser 'ALTA' (A), 'MEDIA' (M) o 'BAJA' (B)–. Después un tiempo recopilando datos, esta persona construye una cadena de Markov para estudiar las probabilidades de distintas secuencias. En este caso los estados del autómata se van a corresponder con las cargas de trabajo ya que son los elementos que vamos a encontrar en la cadena de entrada. El resultado es el autómata de la figura 1.

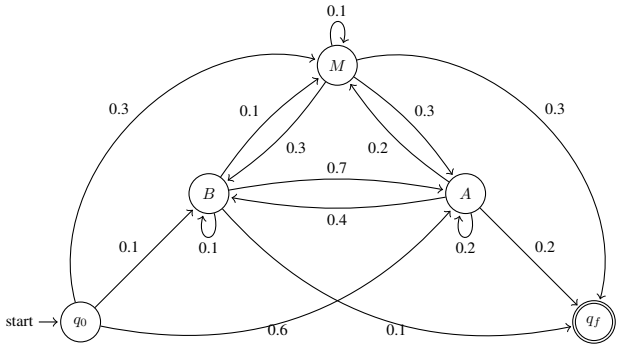

Figura 1: Ejemplo de una cadena de Markov con tres estados.

Se puede calcular la probabilidad de una secuencia de eventos (por ejemplo 'ALTA BAJA') simplemente multiplicando las probabilidades de transición entre los estados que nos interesan. En este caso:

$$\begin{aligned} P(\text{ALTA BAJA}) &= P(ALTA|q_0)P(BAJA|ALTA) \\ P(q_f|BAJA) &= 0{,}6 \cdot 0{,}4 \cdot 0{,}1 = 0{,}024 \end{aligned} \tag{2}$$

Como las cargas de trabajo son eventos observables (se encuentran presentes en la cadena de entrada), las cadenas de Markov son de gran utilidad. No obstante, para el problema de interés de este trabajo no nos sirven ya que como entrada tenemos las palabras del texto y no hay forma de relacionarlas con su categoría semántica. Para ello necesitamos una extensión de las cadenas de Markov: los modelos ocultos de Markov.

2.2. Modelo oculto de Markov

Un modelo oculto de Markov sirve para tratar tanto eventos observables (presentes en la cadena de entrada) como eventos ocultos que consideramos causales del modelo probabilístico. Estos modelos tratan de establecer la relación que hay entre los eventos observables y los estados o eventos ocultos. Se puede especificar como[10]:

$Q = q_1 q_2..q_N$	Conjunto de N estados.
$A = a_{00}...a_{nn}$	Matriz de probabilidades de transición, donde cada casilla a_{ij} representa la probabilidad de ir del estado i al estado j. La suma de todos los arcos que salen de un estado i debe ser igual a 1.
$O = o_1 o_2..o_T$	Cadena de entrada formada por una secuencia de T observaciones, donde cada una pertenece a un vocabulario $V = v_1, v_2, ..., v_V$.
$B = b_i(o_t)$	Matriz de probabilidades de emisión, donde cada una expresa la probabilidad de que la observación o_t sea generada por el estado i.
q_0, q_F	Estados artificiales que representan el inicio y el final de la cadena de entrada respectivamente.

Para poder trabajar de forma eficiente con este modelo se realizan dos suposiciones. La primera es que se cumple la propiedad de Markov, indicando que la probabilidad del estado actual depende únicamente del estado anterior.

$$P(q_i|q_0..q_{i-1}) = P(q_i|q_{i-1}) \tag{3}$$

La segunda es que la probabilidad de una observación o_i depende únicamente del estado q_i donde se ha producido la observación:

$$P(o_i|q_1...q_i, ..., q_T, o_1, ..., o_i, ..., o_T) = P(o_i|q_i) \tag{4}$$

Para tener un ejemplo al que referirnos durante la explicación, supongamos que en el año 2050 se pretende saber cuál fue la carga de trabajo en la oficina del apartado anterior en 2018. Por desgracia, se ha perdido toda la información de la compañía excepto la libreta del secretario, donde anotaba las unidades de cápsulas de café que se consumían cada día.

En este caso tenemos unos eventos observables (cápsulas de café consumidas en un día determinado) mediante los que pretendemos explicar los eventos ocultos (carga de trabajo en la oficina). Para simplificar un poco el problema, de nuevo supondremos que la carga de trabajo solo puede ser 'ALTA', 'MEDIA' o 'BAJA'. Con estos datos podemos definir un modelo oculto de Markov de la siguiente manera:

- Q Carga de trabajo ('ALTA', 'MEDIA', 'BAJA').
- A Matriz de probabilidad de transición entre los distintos estados ocultos. Por ejemplo, la probabilidad de pasar de 'ALTA' a 'BAJA'.
- O Matriz con las probabilidades que relaciona el consumo de café con los distintos estados. Por ejemplo, la probabilidad de que consumir 30 cápsulas de café en un día signifique una carga de trabajo 'ALTA'.

Una vez definida la estructura del modelo oculto de Markov, hay tres problemas que se deben resolver:

- Estimación de probabilidades: Dado un modelo oculto de Markov λ y una secuencia de observaciones O, determinar la probabilidad de que se dé O ($P(O|\lambda)$).
- Decodificación: Dado un modelo oculto de Markov λ y una secuencia de observaciones O, determinar la mejor secuencia de estados ocultos (es decir, aquella con mayor probabilidad asociada).
- Aprendizaje: Dada una secuencia de observaciones O y un conjunto de estados del modelo oculto de Markov, calcular los parámetros óptimos A y B del modelo. Es decir, a partir de una secuencia de observaciones de entrenamiento O calcular los valores de A y B para los cuales la probabilidad de que se produzca O asociada a los estados ocultos del modelo sea máxima.

2.2.1. Estimación de probabilidades

El primer problema trata de calcular la probabilidad de que se dé una secuencia de eventos observables O. Como se ha visto en el apartado 2.1, para una cadena de Markov donde los elementos de la cadena de entrada coinciden con los estados del autómata se puede calcular la probabilidad de una secuencia multiplicando las probabilidades de transición entre los estados que nos interesan. No obstante para un modelo oculto de Markov no sabemos que estados se corresponden con los elementos de la cadena de entrada.

En lugar de abordar directamente el problema completo, vamos a empezar por una versión simplificada. Supongamos que sabemos la secuencia de estados ocultos (carga de trabajo en la oficina) y queremos calcular la probabilidad de una secuencia de cápsulas de café consumidas. Por ejemplo, sabemos que para dos días la carga de trabajo fue 'ALTA' 'BAJA', y queremos calcular la probabilidad de que se hayan consumido '30' y '10' cápsulas de café durante esos días.

Sabemos que cada evento observable se corresponde con únicamente un estado oculto, de modo que las dos secuencias van a ser de la misma longitud. Teniendo en cuenta esta correspondencia uno-a-uno y la propiedad de Markov, podemos definir la probabilidad de una secuencia de observaciones $O = o_1, o_2, ..., o_T$ asociada a unos estados ocultos $Q = q_0, q_1, q_2, ..., q_T$ como[10]:

$$P(O|Q) = \prod_{i=1}^{T} P(o_i|q_i) \tag{5}$$

Que para el ejemplo anterior sería:

$$P(30\ 10|ALTA\ BAJA) = P(30|ALTA)P(10|BAJA) \tag{6}$$

Como en el problema real no se sabe la secuencia de estados ocultos, para calcular la probabilidad de consumir 30 y 10 cápsulas de café hay que ponderar la probabilidad anterior con la probabilidad de que se de la secuencia de eventos ocultos:

$$P(O,Q) = P(O|Q)P(Q) = \prod_{i=1}^{T} P(o_i|q_i) \prod_{i=1}^{T} P(q_i|q_{i-1}) \tag{7}$$

Que aplicado al ejemplo:

$$\begin{aligned} P(30\ 10, ALTA\ BAJA) =& P(ALTA|q_0)P(BAJA|ALTA) \\ & P(30|ALTA)P(10|BAJA) \end{aligned} \tag{8}$$

Finalmente, sumando las probabilidades anteriores para todos los estados posibles, hallaremos la probabilidad de obtener la cadena de observaciones:

$$P(O) = \sum_{Q} P(O,Q) = \sum_{Q} P(O|Q)P(Q) \tag{9}$$

Para un modelo con N estados ocultos y una secuencia de T observaciones existen N^T posibles secuencias de estados ocultos. Si ambos números son grandes–como suele ser habitual–, no podemos realizar el cálculo de forma eficiente. Existen métodos para simplificar el cálculo, pero como para este trabajo no se van a calcular probabilidades de cadenas no vamos a entrar en detalles[10].

En el caso de *POS-tagging* este cálculo solo resultaría de interés si quisiéramos calcular la probabilidad de una frase concreta (por ejemplo la probabilidad de la frase "Hoy hace calor.").

2.2.2. Decodificación

El segundo problema trata de determinar la secuencia de estados ocultos que mejor se adapta a una cadena de eventos observables, es decir, encontrar los estados que mejor explican una secuencia de observaciones. A este proceso se le llama decodificación. Para el ejemplo de la oficina, el objetivo de la decodificación es calcular la carga de trabajo (estados ocultos) que hubo en una serie de días según el consumo de café (eventos observables).

El método de decodificación más usado para modelos ocultos de Markov es el **algoritmo de Viterbi**. Este algoritmo usa programación dinámica para procesar la secuencia de observaciones de izquierda a derecha rellenando una tabla de probabilidades. Cada celda $v_t(j)$ de la tabla representa la probabilidad de que nos encontremos en el estado j después de haber recorrido los t primeros elementos de la cadena de entrada pasando por la secuencia de estados más probable $q_0, q_1, ..., q_{t-1}$[10]. Formalmente:

$$v_t(j) = \max_{q_0, q_1, ..., q_{t-1}} P(q_0, q_1..q_{t-1}, o_1, o_2...o_t, q_t = j) \tag{10}$$

Se trata de un algoritmo recursivo ya que para el paso t necesitaremos el camino más probable en $t-1$. Si escribimos la fórmula anterior de forma recursiva obtenemos:

$$v_t(j) = \max_{i=1}^{N} v_{t-1}(i)\ a_{ij}\ b_j(o_t) \tag{11}$$

Donde:

$v_{t-1}(i)$	Probabilidad del camino más probable para el paso anterior.
a_{ij}	Probabilidad de transición del estado anterior q_i al estado actual q_j.
$b_j(o_t)$	Probabilidad de emisión para el evento observable o_t dado el estado q_j.

El caso base de la recursión es cuando $t = 1$, donde se elige como primer estado el más probable para la observación o_1 y se construye el camino a partir de allí según la fórmula anterior.

Para la implementación del algoritmo se utilizan dos variables:

$viterbi$	Tabla de programación dinámica descrita anteriormente.
$backpointer$	Variable en la que se recoge el camino de estados más probable en un momento determinado.

Algorithm 1 Algoritmo de Viterbi

function VITERBI(O, λ)
  $T \leftarrow len(O)$
  $N \leftarrow len(\lambda)$
  $viterbi \leftarrow Matrix(N + 2, T)$
  for each state *s* **from** 1 **to** *N* **do**
    $viterbi[s, 1] \leftarrow a_{0,s} * b_s(o_1)$
    $backpointer[s, 1] \leftarrow 0$
  end for
  for each state *t* **from** 2 **to** *T* **do**
    for each state *s* **from** 1 **to** *N* **do**
      $viterbi[s, t] \leftarrow \max_{s'=1}^{N} viterbi[s', t-1] *$
        $a_{s',s} * b_s(o_t)$
      $backpointer[s, t] \leftarrow \text{argmax}_{s'=1}^{N}($
        $viterbi[s', t-1] * a_{s',s})$
    end for
  end for
  $viterbi[q_F, T] \leftarrow \max_{s=1}^{N} viterbi[s, T] * a_{s,q_F}$
  $backpointer[q_F, T] \leftarrow \text{argmax}_{s=1}^{N}(viterbi[s, T] *$
    $a_{s,q_F})$
  return backtrace of backpointers from T to 0
end function

2.2.3. Entrenamiento

El tercer problema de los modelos ocultos de Markov tiene como objetivo aprender los parámetros del modelo. Es decir, a partir de una secuencia de observaciones O y un vocabulario de posibles estados Q, calcular las matrices A y B[10].

Para ello existe un algoritmo estándar llamado **algoritmo de esperanza-maximización** (EM) [13] que estima las probabilidades iterativamente, obteniendo una mejor aproximación cada iteración. Se trata de un algoritmo general que funciona para cualquier problema para el que queramos aplicar un modelo oculto de Markov.

No obstante, para el etiquetado de texto no hace falta iterar para obtener las probabilidades ya que tenemos una secuencia de eventos observables y ocultos con los que entrenar el modelo. Es decir, disponemos de un texto previamente etiquetado. Por ejemplo, 'Un/DET hombre/NOMBRE baila/VERBO', donde DET, NOMBRE y VERBO representan etiquetas o estados ocultos y 'Un', 'hombre' y 'baila' son las observaciones de la cadena de entrada (ver subsección 2.3 para más detalles).

Gracias a que conocemos de antemano los eventos ocultos asociados a la secuencia de entrenamiento podemos obtener directamente las probabilidades de transición y las probabilidades de emisión:

$$a_{i,j} = P(j|i) = \frac{C(i,j)}{C(i)} \tag{12}$$

Mediante esta fórmula obtenemos la estimación de máxima verosimilitud para $a_{i,j}$ contando el número de veces que encontramos que el evento i viene seguido de j y normalizando por el total de veces que se ha pasado por el estado i.

$$b_j(o_t) = P(o_t|j) = \frac{C(j,o_t)}{C(j)} \tag{13}$$

Para las probabilidades de emisión, contamos cuantas veces la observación o_t se encuentra asociada al evento j (por ejemplo, cuantas veces consumir 30 cápsulas de café se asocia con una carga de trabajo 'MEDIA') y se normaliza por el número de veces que ha aparecido el evento j.

Si para el ejemplo de la oficina tuviéramos una cadena de observaciones etiquetadas ('30/ALTA 20/ALTA 10/BAJA 25/ALTA ...'), podríamos adaptar las fórmulas anteriores de la siguiente manera:

$$a_{ALTA,BAJA} = P(BAJA|ALTA) = \frac{C(ALTA,BAJA)}{C(ALTA)} \tag{14}$$

$$b_{ALTA}(30) = P(30|ALTA) = \frac{C(ALTA,30)}{C(ALTA)} \tag{15}$$

De esta forma, se pueden calcular las matrices A y B realizando un recuento de las distintas observaciones y estados ocultos. Esto supone una gran mejoría de rendimiento, ya que se puede hacer con un único recorrido del texto.

2.3. Modelos ocultos de Markov aplicados al etiquetado de texto

El etiquetado de texto es un problema cuyos requisitos se adaptan a la forma natural de los modelos de markov. Por su

descripción resulta evidente que los eventos observables van a ser las palabras y los eventos ocultos, su categoría semántica. Adaptado a la especificación formal de un modelo oculto de Markov tenemos:

$Q = q_1 q_2 .. q_N$	Conjunto de estados. Cada estado se corresponde con una categoría semántica (Nombre, Verbo, Adverbio, ...). Existen diversos modelos de categorías según su exhaustividad, para información más detallada ver apéndice A.
$A = a_{00} .. a_{ij} .. a_{nn}$	Probabilidad de que después de una palabra de categoría semántica i venga una palabra de categoría semántica j.
$O = o_1 o_2 .. o_T$	Secuencia de T palabras.
$B = b_i(o_t)$	Probabilidad de que una palabra o_t pertenezca a la categoría semántica i.
q_0, q_F	Estados artificiales que representan el inicio y el final de la oración respectivamente.

Como a través de Internet se pueden encontrar textos largos etiquetados, podemos aplicar las fórmulas vista en el apartado anterior en lugar del algoritmo de entrenamiento iterativo habitual de los modelos ocultos de Markov. La única modificación respecto los algoritmos teóricos es añadir un método para tratar palabras desconocidas.

2.3.1. Observaciones desconocidas

Del algoritmo de Viterbi (Algoritmo 1) y de las ecuaciones (12) y (13) podemos ver que si nos encontramos una observación o_t que no ha aparecido durante el entrenamiento, puede haber problemas. Concretamente, cuando para un estado s y una observación o_t calculamos la probabilidad del camino máximo que nos ha llevado a ese punto:

$$viterbi[s,t] \leftarrow \max_{s'=1}^{N} viterbi[s', t-1] * a_{s',s} * b_s(o_t)$$

Si no conocemos o_t entonces $b_s(o_t) = 0$ y por tanto, $viterbi[s,t] = 0$. Esto implicará que toda la columna de la matriz $viterbi$ que hace referencia a la observación o_t sea 0. Para la siguiente observación o_{t+1} nos vamos a encontrar lo siguiente:

$$viterbi[s,t+1] \leftarrow \max_{s'=1}^{N} viterbi[s', t] * a_{s',s} * b_s(o_{t+1})$$

Donde $viterbi[s', t] = 0, \forall s'$ ya que se desconocía la observación o_t. Esta naturaleza recursiva del algoritmo provocará que una vez nos encontremos una observación desconocida, el resto de la cadena tenga probabilidad 0 y no podremos decidir cual es la mejor secuencia de estados[9].

Para el etiquetado de texto éste resulta un problema crítico, ya que siempre aparecen nuevas palabras debido a errores gramaticales, números, nombres propios, direcciones de correo electrónico y palabras inventadas entre otras. Uno de los primeros en describir este fenómeno fue Heaps (1978) [4] que describió la relación entre el tamaño del texto y el tamaño del vocabulario como

$$v = k \cdot n^{\beta} \tag{16}$$

donde v es el tamaño del vocabulario para un texto de n palabras y k y β son parámetros que cambian según el tipo de texto.

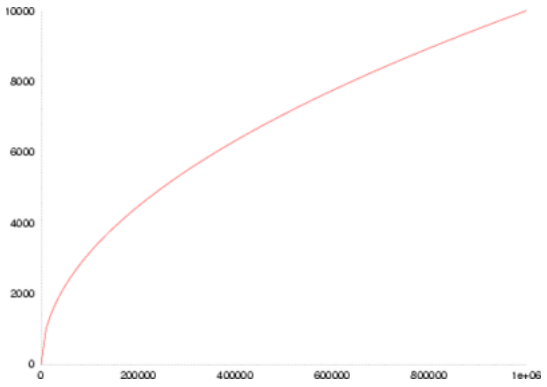

Figura 2: Relación entre el tamaño del vocabulario y el tamaño del texto para $\beta \in (0, 1]$[17].

Como podemos ver en la figura 2, el crecimiento del vocabulario es elevado al principio y después va decayendo, aunque nunca para de crecer. Para un modelo de Markov enfocado al etiquetado de texto esto implica que siempre hay que estar preparado para tratar palabras desconocidas, independientemente del tamaño del texto de entrenamiento.

Existen diversos métodos para tratar con este problema. Para este trabajo se han implementado dos casos:

- Alisado de Laplace.
- Tag más frecuente.

Hay modelos más complejos, como los basados en normas (si la palabra termina en '-ing' es un verbo, si empieza en mayúscula es un nombre propio, ...) o modelos estadísticos elaborados[12], pero su desempeño es similar al método del tag más frecuente[9].

Alisado de Laplace.

El alisado de Laplace, o alisado aditivo, modifica el cálculo de las probabilidades de emisión[9]:

$$b_j(o_t) = P(o_t|j) = \frac{C(j, o_t) + \alpha}{C(j) + \alpha|V|} \tag{17}$$

Donde:

α	Ajuste del alisado. Se recomienda $\alpha = 1$ aunque se puede usar cualquier otro valor[11].
$\|V\|$	Tamaño del vocabulario para el texto de entrenamiento.

Cuando no haya aparecido la observación o_t tendremos que $C(j, o_t) = 0$ y por tanto $b_j(o_t) = \frac{\alpha}{C(j)+\alpha|V|}$.

Tag más frecuente.

Aunque la idea tras el método sea muy sencilla, los resultados que proporciona son muy buenos. Este método asume que las palabras desconocidas siempre son de la etiqueta más frecuente. Para ello, si la observación o_t es desconocida se le da $b_i(o_t) = 1$ cuando i es la etiqueta más frecuente y $b_i(o_t) = 0$ para el resto.

3. Implementación

Para este trabajo se ha realizado una implementación en Python disponible en GitHub [1]. En el repositorio se pueden encontrar:

- Herramientas para solucionar los dos problemas relevantes de los modelos ocultos de Markov para etiquetado de texto –aprendizaje y decodificación–.
- Herramientas para asignar una categoría semántica a cada palabra de un texto dado un modelo previamente entrenado.
- Distintas medidas de desempeño: matriz de confusión, precisión (*precision*), exhaustividad (*recall*) y exactitud (*accuracy*) [2].
- Herramientas para facilitar el entreno de modelos, permitiendo aplicar validación cruzada sobre un texto de entrenamiento.

En este documento no se va a entrar en detalle sobre el código ya que la implementación es muy próxima al modelo teórico explicado en la sección anterior. Documentación adicional sobre el código se encuentra disponible en la página de GitHub.

4. Resultados

En este apartado se presentan tres estudios. Primero, se va a analizar el desempeño del modelo según el tamaño del texto de entrenamiento. Después, se comparan los métodos usados para tratar palabras desconocidas vistos en el apartado 2.3.1. Finalmente, se estudia como afecta la temática del texto de entrenamiento a la hora de realizar predicciones.

4.1. Textos previamente etiquetados

Para las pruebas descritas en este documento se ha usado *Brown Corpus*, la primera colección de textos etiquetados en inglés con más de un millón de palabras. Fue creada en el año 1961 por la Universidad Brown y contiene textos de más de 500 fuentes organizados según su categoría[14].

Aunque la colección de textos de Brown se encuentra públicamente, se ha utilizado como interfaz la librería Natural Language Toolkit (NLTK)[3] de Python ya que además de los textos etiquetados dispone de herramientas para leer y preprocesarlos, haciendo el análisis más sencillo.

Existen muchas formas de etiquetar un texto. Los dos conjuntos de etiquetas más utilizados son el de Penn Treebank y el universal[4]. Para este trabajo se han utilizado las etiquetas del modelo universal.

4.2. Eficacia del modelo según el tamaño del texto de entrenamiento

La mayoría de modelos estadísticos necesitan un mínimo de ejemplos de entrenamiento para tener un desempeño aceptable. Esto también es válido para los modelos ocultos de Markov, ya que cuantas más oraciones se le presenten más se van a ajustar las probabilidades de las matrices A y B. Hablamos de oraciones y no de palabras porque la posición de una palabra en la frase condiciona la probabilidad de la etiqueta. Por ejemplo, si encontramos "Will" a principio de frase es más probable que sea un nombre –aunque depende del contexto, podría ser un modal si se trata de una pregunta– mientras que si viene después de un nombre es más probable que sea un modal –aunque de nuevo, también depende del contexto ya que podría ser la segunda parte de un nombre compuesto–.

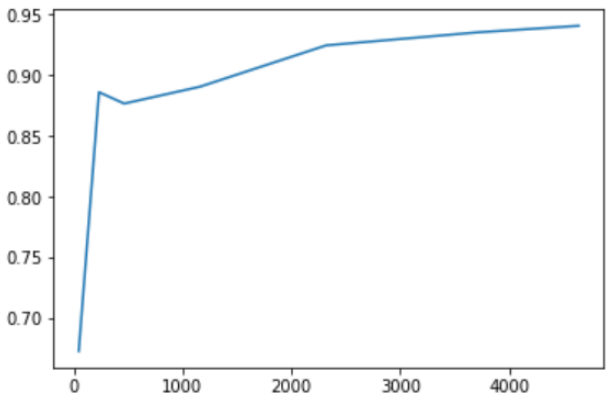

Figura 3: Acierto del modelo en función del tamaño del texto de entrenamiento.

En la figura 2 podemos ver como evoluciona el acierto del modelo en función de las oraciones que se han usado para entrenarlo.

Hasta 500 oraciones el aprendizaje es muy pronunciado. Vemos que en este período el acierto mínimo es aproximadamente un 66 %, un porcentaje alto para un modelo poco entrenado. Eso se debe a que la distribución de las etiquetas está muy desequilibrada, tal y como se puede ver en la figura 4. Los estados $< s >$ y $< e >$ son los estados artificiales q_0 y q_F respectivamente, así que es de esperar que no hayan aparecido durante el texto de entrenamiento.

Entre 500 y 2500 frases el modelo sigue aprendiendo a un buen ritmo, proporcionando un acierto de entre el 85 y el

[1] https://github.com/Tonicrespit/hmm_pos_tagger

[2] Ver apéndice B para más detalles sobre las medidas de desempeño.

[3] https://www.nltk.org/

[4] Ver apéndice A para información detallada sobre los dos modelos de etiquetado

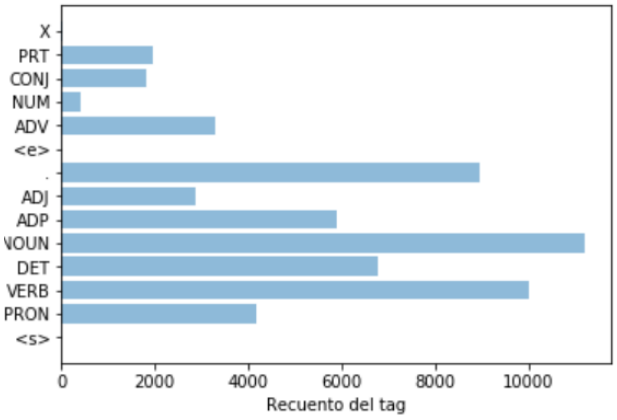

Figura 4: Distribución de los tags en el texto de entrenamiento.

93 %. Ya lo podemos considerar como bueno.

Con más de 2500 oraciones de entrenamiento el modelo sigue mejorando, pero ya de forma lenta.

Al ser un acierto muy próximo al 95 % resulta interesante ver donde se concentran los fallos, que es la información que nos proporciona la matriz de confusión de la figura 4.

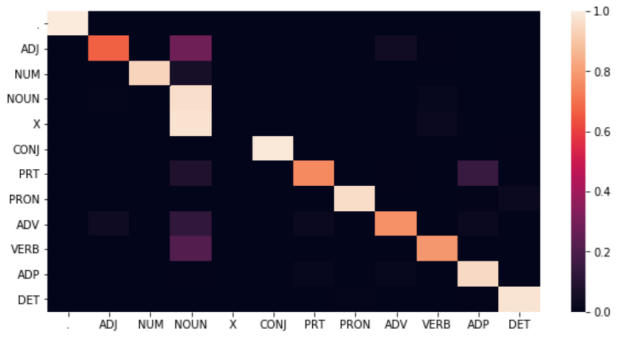

Figura 5: Matriz de confusión normalizada para un modelo.

Al estar la distribución de clases muy descompensada la matriz de confusión se encuentra normalizada, de modo que cada casilla en lugar de representar el número de ejemplos clasificados correctamente o el número de ejemplos en los que se ha producido un error de tipo 1 o de tipo 2, representan porcentajes. Un color claro significa un porcentaje muy alto y un color oscuro, un porcentaje bajo.

La mayoría de los errores se producen clasificando nombres, ya que se confunden con una gran variedad de clases. Eso se debe a que el método usado para tratar palabras desconocidas asigna esta clase a cualquier observación que no haya aparecido durante el entrenamiento. El caso más extremo se da con la clase 'X'[5], de la que se confunden la mayoría de sus apariciones con la clase más frecuente. Se trata de un caso particular ya que esta etiqueta siempre acompaña a palabras a las que no se les puede asignar ninguna clase y por tanto, son poco frecuentes. Por ese motivo no suelen aparecer en el texto de entrenamiento y para las predicciones se consideran como palabras nuevas, asignándoles la clase más frecuente.

4.3. Comparativa de los distintos métodos usados para tratar palabras desconocidas

En la sección 2.3.1 se han presentado dos métodos para tratar con palabras desconocidas: el alisado de Laplace y el tag más frecuente. En esta sección se comparan los resultados de ambos para el mismo texto de entrenamiento.

Para el experimento se ha entrenado un modelo utilizado validación cruzada[7] con $k = 10$. Se han sacado tres estadísticas: exactitud, precisión y exhaustividad[6].

Método	Exactitud	Precisión	Exhaustividad
Tag más frecuente	94.55 %	94.68 %	94.55 %
Alisado de Laplace	93.04 %	92.16 %	93.04 %

En todos los aspectos resulta mejor usar el método del tag más frecuente. De nuevo, el motivo de ello se encuentra relacionado con la distribución de clases y la matriz de confusión expuesta en el apartado anterior. Al haber tanta diferencia entre el número de apariciones de cada etiqueta, asignar la más frecuente provoca que la probabilidad de acertar sea alta.

4.4. Comparativa de la eficacia de modelos entrenados con distintas temáticas de textos

Según el tipo de texto que tratemos nos vamos a encontrar con un estilo de redacción distinto. Por ejemplo, para artículos técnicos o legales se usa un lenguaje preciso para evitar ambigüedades mientras que en poesía se juega con el doble sentido de las palabras.

En este apartado se ha diseñado un experimento para ver cómo afectan los distintos estilos de redacción a un modelo oculto de Markov. Para ello se ha entrenado un modelo usando un texto de aventuras mediante validación cruzada con k = 10. Con el modelo obtenido se van a realizar predicciones sobre textos de noticias, religión y romance. Para reducir el sesgo del modelo, las predicciones sobre el mismo género (aventuras) se realizaran sobre un fragmento separado previamente.

Género predicho	Exactitud	Precisión	Exhaustividad
Aventuras (train)	94.55 %	94.68 %	94.55 %
Noticias	89.41 %	79.15 %	89.41 %
Textos religiosos	88.50 %	78.42 %	88.50 %
Romance	92.98 %	78.75 %	92.98 %

Como podemos ver en la tabla, los resultados cambian ligeramente según la temática y el estilo del texto, siendo mejores para el género con el que se ha entrenado el modelo.

[5] Para más información sobre el significado de cada clase ver apéndice A.

[6] Para más información de las métricas , ver apéndice B

Resulta curioso notar que el acierto del modelo nos puede servir como medida de distancia entre estilos, aunque no sea muy eficiente de calcular. Por ejemplo, vemos que el género de aventuras –el usado para entrenar el modelo– resulta más similar al romance que a las noticias.

5. Conclusiones

Después de ver como funcionan los modelos ocultos de Markov y estudiar como se comportan según su entrenamiento, se pueden detallar algunas pautas para maximizar su rendimiento. Primero, se debe tener un conjunto de textos etiquetados suficientemente grande, de al menos 2000-3000 oraciones. En segundo lugar, hay que intentar que el texto de entrenamiento sea de temática similar al que queremos predecir. Por último, se debe considerar cuidadosamente el método que se quiere usar para tratar palabras desconocidas ya que condicionará el acierto del modelo y la distribución de los errores.

Es importante destacar que los resultados de los experimentos presentados son únicamente válidos para textos etiquetados usando el conjunto de etiquetas universal[7]. Si se quisiera utilizar un conjunto de etiquetas más extenso –por ejemplo Penn TreeBank– un texto de 2000-3000 oraciones no nos proporcionaría un modelo oculto de Markov con buen desempeño ya que no se le presentarían suficientes palabras de cada etiqueta. Además, habría que revisar como se tratan palabras desconocidas ya que para un conjunto de etiquetas más extenso asignar siempre el *tag* más frecuente no proporcionaría tan buenos resultados.

Apéndice A. *Tagsets*

Para etiquetar un texto existen varios conjuntos de etiquetas. En este apéndice se detallan dos de los más usados –el conjunto universal y el de Penn Treebank– y se exponen algunos de sus puntos fuertes y débiles.

Apéndice A.1. *Tagset* universal

Este conjunto de etiquetas es el más sencillo y fue descrito por Slav Petrov, Dipanjan Das y Ryan McDonald en el artículo *A Universal Part-of-Speech Tagset* [15]. Su objetivo era diseñar un conjunto de etiquetas común para la mayoría de idiomas que sirviera como estándar a los investigadores. Consta de doce etiquetas[5]:

Etiqueta	Significado
ADJ	Adjetivo.
ADP	Adposición.
ADV	Adverbio.
CONJ	Conjunción.
DET	Artículo o determinante.
NOUN	Nombre.
NUM	Numeral –tanto "twenty-two" como "22" pertenecen a esta categoría–.
PRT	Partícula.
PRON	Pronombre.
VERB	Verbo.
.	Signo de puntuación.
X	Otras palabras. Sirve como comodín cuando una palabra no pertenece a algún *tag* anterior. La mayoría de veces se debe a abreviaturas (como "gr8" en lugar de "*great*") o erratas.

Esta forma de etiquetado resulta fácil de entender ya que es muy próxima a las categorías semánticas básicas. Además, al no tener mucho nivel de detalle no se necesita un texto de entrenamiento tan grande como para modelos más complejos.

No obstante, esta simplicidad también hace que no podamos distinguir entre distintas subcategorías. Por ejemplo, "Will/NOUN" –nombre propio– y "will/NOUN" –"voluntad", nombre común– son indistinguibles.

Apéndice A.2. Penn Treebank

Este conjunto de etiquetas fue desarrollado por Helmut Schmid en el *Institute for Computational Linguistics of the University of Stuttgart* y es el usado para etiquetar los textos de TreeTagger[1]. Consta de 36 etiquetas[16]:

Etiqueta	Significado
CC	Conjunciones coordinantes.
CD	Número cardinal.
DT	Determinante.
EX	*Existential there*, que hace referencia al uso de *there* antes de un verbo –*There is a river*–.
FW	Palabra extranjera –*Foreign word*–.
IN	Preposición o conjunción subordinante.
JJ	Adjetivo.
JJR	Adjetivo comparativo.
JJS	Adjetivo superlativo.
LS	Marcador de lista.
MD	Modal.
NN	Nombre singular o de masa.
NNS	Nombre plural.
NNP	Nombre propio singular.
NNPS	Nombre propio plural.
PDT	Predeterminante.
POS	Terminación posesiva.
PRP	Pronombre personal.
PRP$	Pronombre personal posesivo.
RB	Adverbio.
RBR	Adverbio comparativo.
RBS	Adverbio superlativo.
RP	Partícula.
SYM	Símbolo.
TO	Palabra *to*.

[7] Ver apéndice A.

UH	Inyección.
VB	Verbo en infinitivo.
VBD	Verbo en pasado.
VBG	Verbo en gerundio o *present participle*.
VBN	Verbo en *past participle*.
VBP	Verbo en presente sin la forma de 3ª persona.
VBZ	Verbo en presente con la forma de 3ª persona.
WDT	Determinante que empieza por *wh- –what, which, whose, whatever, whichever–*.
WP	Pronombre que empieza por *wh- –what, who, whom, which...–*.
WP$	Pronombre posesivo que empieza por *wh-*.
WRB	Adverbio que empieza por *wh-*.

Sus etiquetas tienen mayor grado de detalle que las del universal y nos proporcionan herramientas para distinguir entre homónimos como el nombre propio "*Will/NNP*" y el nombre común "*will/NN*". No obstante, usar estas etiquetas implica entrenar el modelo oculto de Markov con un texto de mayor tamaño para que tenga suficientes ejemplos para cada etiqueta.

Cabe destacar que el conjunto de etiquetas expuesto es para textos en inglés.

Apéndice B. Medidas para el desempeño de un modelo

Para medir el desempeño de los modelos de clasificación existen distintas métricas pensadas para evaluar distintos aspectos del modelo. En este apéndice se presentan tres de las más usadas: exactitud (*accuracy*), precisión (*precision*) y exhaustividad (*recall*).

Apéndice B.1. Matriz de confusión

Antes de entrar en detalle sobre las métricas es necesario entender como interpretar un matriz de confusión, ya que a partir de ella se pueden calcular todas las medidas presentadas en este apéndice.

		Predicción	
		Positivos	**Negativos**
Observación	**Positivos**	Verdaderos Positivos (VP)	Falsos Negativos (FN)
	Negativos	Falsos Positivos (FP)	Verdaderos Negativos (VN)

Figura 6: Matriz de confusión[18]

Una matriz de confusión es una herramienta que permite visualizar el desempeño de un algoritmo de aprendizaje supervisado. Cada columna representa el número de predicciones realizadas para cada clase y cada fila, las instancias de la clase real. Para un problema de clasificación binaria tenemos la matriz de la figura 6, donde:

- Verdaderos positivos (VP) es la cantidad de positivos que fueron clasificados correctamente por el modelo.
- Verdaderos negativos (VN) es la cantidad de negativos que fueron clasificados correctamente por el modelo.
- Falsos negativos (FN) es la cantidad de positivos que fueron clasificados incorrectamente como negativos.
- Falsos positivos (FP) es la cantidad de negativos que fueron clasificados incorrectamente como positivos.

Esto se puede trasladar directamente a un problema de clasificación con más de dos clases, creando una matriz de nxn. En esa matriz, cada celda i, j contendrá el número de elementos de la clase i a los que se les ha asignado la clase j. La diagonal principal van a ser los elementos clasificados correctamente.

Apéndice B.2. Exactitud

La exactitud es una métrica usada para evaluar modelos de clasificación. Informalmente, se corresponde a la fracción de predicciones que el modelo ha realizado correctamente[6].

$$\text{Exactitud} = \frac{\text{Predicciones correctas}}{\text{Total de predicciones}} \tag{18}$$

Se puede sacar a través de una matriz de confusión de la siguiente manera:

$$\text{Exactitud} = \frac{\text{VP} + \text{VN}}{\text{VP} + \text{VN} + \text{FP} + \text{FN}} \tag{19}$$

Resulta una buena medida de desempeño para conjuntos de datos balanceados (donde hay un número similar de ejemplos de cada clase en los datos de entrenamiento) pero para datos desequilibrados resulta muy pobre. Por ejemplo, si tenemos un 99 % de datos que pertenecen a una clase y un 1 % a otra, un modelo con un 99 % de exactitud sería malo ya que probablemente clasifique a todas las observaciones como la clase mayoritaria.

Apéndice B.3. Precisión y exhaustividad

La precisión representa la proporción de clasificaciones positivas correctas y se define como[8]:

$$\text{Precision} = \frac{\text{VP}}{\text{VP} + \text{FP}} \tag{20}$$

De manera informal, es la proporción de los elementos clasificados como positivos que realmente son positivos.

La exhaustividad representa la proporción de positivos reales que se clasificaron correctamente. Formalmente[8]:

$$\text{Exhaustividad} = \frac{\text{VP}}{\text{VP} + \text{FN}} \tag{21}$$

De manera informal, es la proporción de ejemplos positivos que se clasifican como positivos. Para una descripción gráfica de ambas métricas, ver la figura 7.

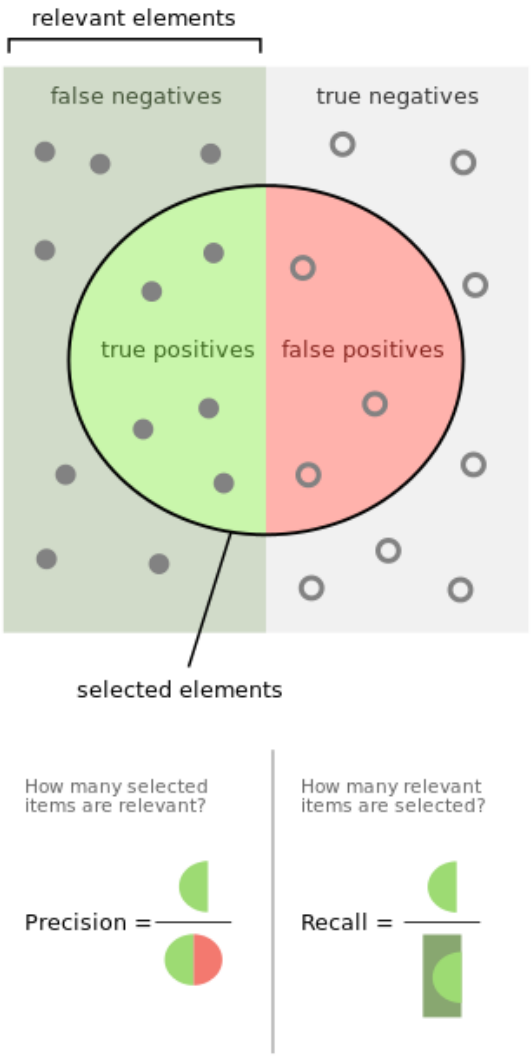

Figura 7: Relación entre precisión y exhaustividad.[3]

Las dos medidas se encuentran en equilibrio, de modo que si se intenta aumentar una, la otra va a disminuir. El mismo problema se presenta en estadística cuando tratamos con errores de tipo I y de tipo II. En algunos campos, como en el contraste de hipótesis, se decide fijar un error de tipo I máximo llamado nivel de significación α y se minimiza el error de tipo II sin que el error de tipo I supere α. Para los modelos de clasificación no hay una solución universal y va a depender de la aplicación que vaya a tener el modelo.

Como la relación entre las dos métricas no es lineal, para modelos estadísticos normalmente se busca un umbral donde haya un equilibrio. Para el ejemplo de la figura 8 el umbral sería próximo a 0.45.

Para problemas con más de dos clases, se calcula la precisión y la exhaustividad para cada clase y después se calcula la media. Según el balanceo de clases, se puede usar la media común o la media ponderada por el soporte de cada clase.

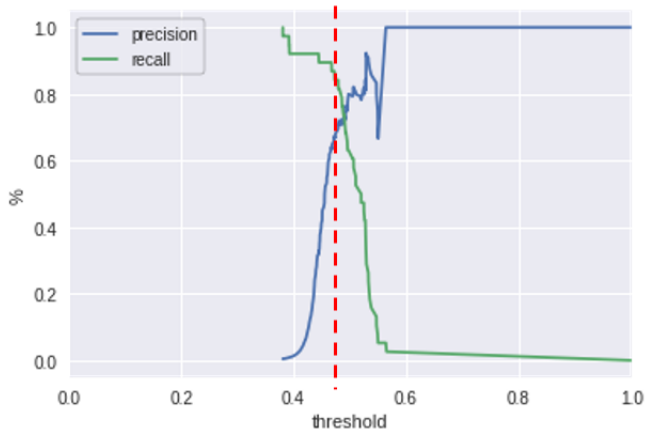

Figura 8: Equilibrio entre precisión y exhaustividad.[2]

Referencias

[1] Treetagger - a part-of-speech tagger for may languages. http://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/.

[2] Visualizing the confusion matrix. https://www.sanyamkapoor.com/machine-learning/confusion-matrix-visualization/.

[3] Wikipedia, precisión y exhaustividad. https://www.quora.com/What-is-the-difference-between-Precision-and

[4] Bruce Croft, Donald Metzler, and Trevor Strohman. *Search Engines: Information Retrieval in Practice*. Pearson., 2010.

[5] Universal Dependencies. Etiquetas del modelo de etiquetado universal. http://universaldependencies.org/u/pos/.

[6] Google Developers. Accuracy. https://developers.google.com/machine-learning/crash-course/classification/accuracy.

[7] Google Developers. Conjuntos de entrenamiento y prueba: Sepración de datos. https://developers.google.com/machine-learning/crash-course/training-and-test-sets/splitting-data.

[8] Google Developers. Precision-recall. https://developers.google.com/machine-learning/crash-course/classification/precision-and-recall.

[9] Martin Haulrich. Different approaches to unknown words in a hidden markov model part-of-speech tagger. 2009.

[10] Dan Jurafsky and James H. Martin. *Speech and Language Processing*. Stanford University, 2017, draft.
[11] Bill MacCartney. Nlp lunch tutorial: Smoothing. 2005.
[12] Andrew McCallum, Dayne Freitag, and Fernando Pereira. Maximum entropy markov models for information extraction and segrmentation.
[13] Andrew Ng. The em algorithm.
[14] International Computer Archive of Modern and Medieval English. Recopilación de textos por la universidad brown. http://icame.uib.no/brown/bcm-los.html.
[15] Slav Petrov, Dipanjan Das, and Ryan McDonald. A universal part-of-speech tagset. 2012.
[16] Department of Linguistics University of Pennsylvania. Etiquetas del modelo de etiquetado penn treebank. https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html.
[17] Wikipedia. Heap's law. https://en.wikipedia.org/wiki/Heaps%27_law.
[18] Carlos Zelada. Evaluación de modelos de clasificación. 2017.