Modelos markovianos para secuencias y aplicaciones a la predicción de genes

Tesis de Maestría en Ingeniería Matemática

Andrea Mesa

Tutor: Dr. Gustavo Guerberoff

Co-Tutor: Dr. Fernando Álvarez

Tribunal:

Dr. Enrique Lessa

Dr. Juan Kalemkerian

Dr. Marco Scavino

Universidad de la República

Facultad de Ingeniería

Montevideo, Uruguay

10 de noviembre 2010

Índice general

Índice de figuras

Índice de cuadros

Resumen

En esta tesis se estudian técnicas para modelar secuencias de datos, en particular secuencias de ADN. Si bien las técnicas de secuenciación de genomas han avanzado mucho en estos últimos años, las herramientas para la anotación, es decir para la identificación de las regiones codificantes o genes, no se han desarrollado con la misma velocidad, por lo que es de gran importancia el estudio y perfeccionamiento de procedimientos que permitan la predicción de genes.

Como posibles predictores de genes se estudian los modelos de Markov ocultos (HMM) y modelos probabilísiticos condicionados como los modelos de Markov de máxima entropía (MEMM) y los campos aleatorios condicionados (CRF), con un especial énfasis en los primeros. Se detallan los algoritmos de entrenamiento y posterior inferencia de los HMM y se aplican al caso particular de búsqueda de genes VSG en *Trypanosoma brucei*.

Capítulo 1

Introducción

En el análisis de problemas biológicos es frecuente la utilización de modelos matemáticos para describir, explicar y predecir el fenómeno bajo estudio. Un modelo matemático permite la representación de un problema de una manera objetiva, en la cual es posible definir un conjunto de relaciones entre las mediciones cuantitativas del problema y sus propiedades.
Uno de los problemas básicos que se plantea en biología es el de la transmisión hereditaria, que consiste en inferir los mecanismos que transforman elementos biológicos, como el ADN, en características fenotípicas, como por ejemplo el color de ojos.

El primer modelo para describir la transmisión hereditaria de caracteres fue presentado por Gregor Mendel en la Sociedad de Historia Natural de Brünn en 1865 y publicado al año siguiente bajo el título *Experimentos sobre hibridación de plantas*. En sus experimentos de cruce de semillas, en particular de variedades de guisantes de la especie *Pisum sativum*, Mendel observó que aquellas variaciones fenotípicas como el color verde o amarillo, ocurrían siempre en proporciones numéricas simples. A las características fenotípicas las llamó *caracteres*, mientras que para referirse a las entidades hereditarias utilizó el nombre de *elementos*. Estos *elementos* son lo que hoy se denominan *genes*. Las versiones diferentes de genes responsables de un determinado fenotipo reciben el nombre de alelos, por lo que los guisantes verdes y amarillos corresponden entonces a distintos alelos del gen responsable del color.

Años más tarde, en 1944, se establece que los genes son segmentos de ácido desoxirribonucleiclo o ADN. El ADN es un polímero (o sea una macromolécula) de nucleótidos cada uno de ellos compuesto por azúcar (desoxirribosa), fosfato y una de las cuatro bases nitrogenadas que los identifica: adenina (A), guanina (G), citosina (C) y timina (T).

En 1953 James Watson y Francis Crick [31] publican un trabajo en el cual presentan la estructura de doble hélice del ADN, donde cada par de bases complementarias (la adenina y la timina son complementarias, mientras que la guanina es el complemento de la citosina) se une mediante un puente de hidrógeno. En cada hebra de esta doble hélice dos nucleótidos adyacentes se conectan por un enlace químico entre el azúcar de uno y el fosfato del siguiente. Así en una secuencia de nucleótidos la hebra

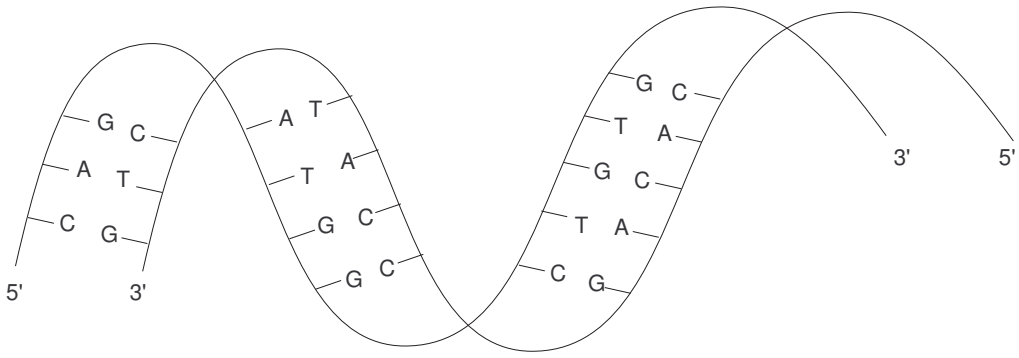

Figura 1.1: La doble hélice del ADN.

positiva de izquierda (extremo $5'$) a derecha (extremo $3'$) se complementa con la hebra en la dirección opuesta (Figura 1.1).

Dentro de las células, el ADN se encuentra en los cromosomas. En las células eucariotas, como las de los animales, plantas y hongos, el ADN se almacena en su mayor parte en el núcleo celular, mientras que en las procariotas (células carentes de núcleo como las de las bacterias y arqueas) se encuentra en el citoplasma. El ADN contenido en cada célula de un organismo es el mismo y es además característico de cada especie. Esta información genética se denomina genoma[1].

Si bien en las células eucariotas el ADN se encuentra en el núcleo, las proteínas se crean en el citoplasma, siendo el ARN (ácido ribonucleico) la molécula que transfiere la información para la síntesis de proteínas necesaria para la actividad y desarrollo celular. El ARN está formado por sólo una hebra y las mismas bases nitrogenadas que el ADN pero sustituyendo timina (T) por uracilo (U).

El dogma central de la biología molecular, propuesto por Francis Crick en 1958 y publicado en 1970 [6], es el proceso por el cual el ADN se transcribe a ARN mensajero y éste se traduce en proteínas. En una secuencia de ADN se tienen genes y regiones intergénicas, los genes contienen partes que se traducen en proteínas (exones) y partes que no se traducen (intrones) y son removidas en la transcripción.

[1] *Genome* es un acrónimo de *gen*e y chromos*ome*

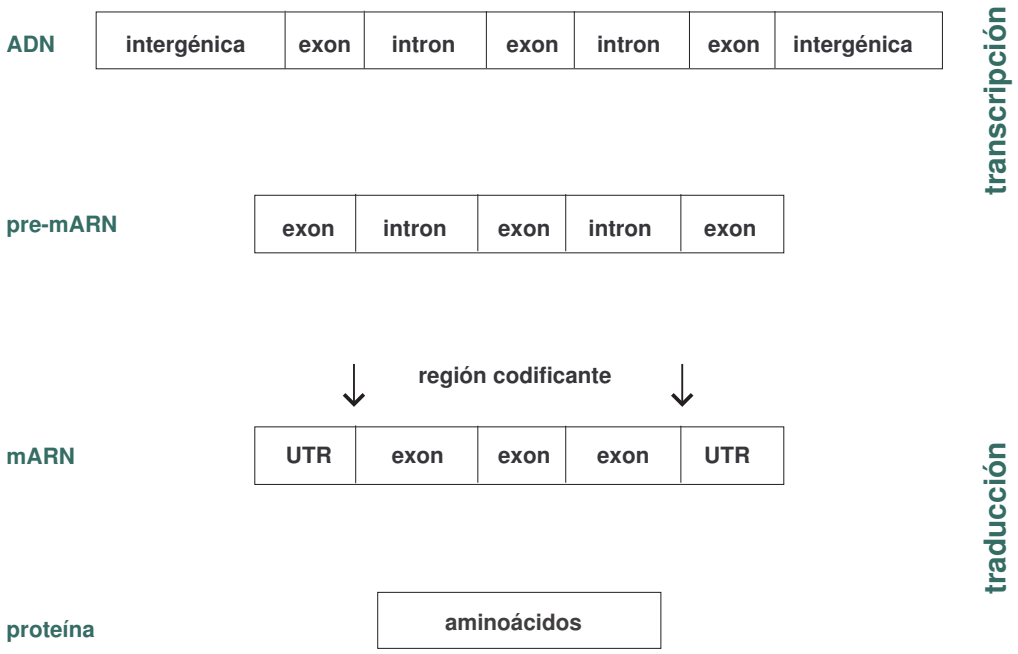

Figura 1.2: Dogma central de la Biología molecular.

En el primer paso de este proceso, la transcripción, se crea una cadena de ácido ribonucleico mensajero (pre-mARN). El pre-mARN es una secuencia de nucleótidos cuyas bases son las mismas que se encuentran en el ADN, salvo que en el ARN la adenina se complementa con uracilo. En el pre-mARN se separan los exones de los intrones y los exones se acoplan para formar el mARN.

A partir del mARN se crea una proteína en el proceso de traducción que comienza en un codón (triplete de nucleótidos consecutivos) de inicio y se detiene en un codón de parada. Las regiones restantes no se traducen y llevan el nombre de *untranslated regions* (UTR). En la etapa de traducción, los ribosomas (macromoléculas cuya función es ensamblar proteínas a partir de la información obtenida del ADN) asocian los codones con los aminoácidos correspondientes (Figura 1.2). Existen veinte aminoácidos, cada uno de los cuales se corresponde con uno o más codones de ARN. Finalmente se crea una macromolécula de forma tridimensional formada por cadenas lineales de aminoácidos denominada proteína. Las proteínas desarrollan una variedad de funciones en la célula (enzimáticas, reguladores, defensivas), dependiendo de su estructura y de la secuencia de aminoácidos que las componen.

En resumen, el ADN es una secuencia de nucleótidos que contiene instrucciones para la síntesis de proteínas. Ahora, dado que no todos los elementos de la cadena contribuyen a este propósito, resulta difícil determinar de antemano qué proteínas serán generadas por una secuencia arbitraria de ADN.

Estas regiones del ADN que codifican en proteínas son los genes, por lo que tiene especial interés la búsqueda de subsecuencias de ADN que correspondan a genes.

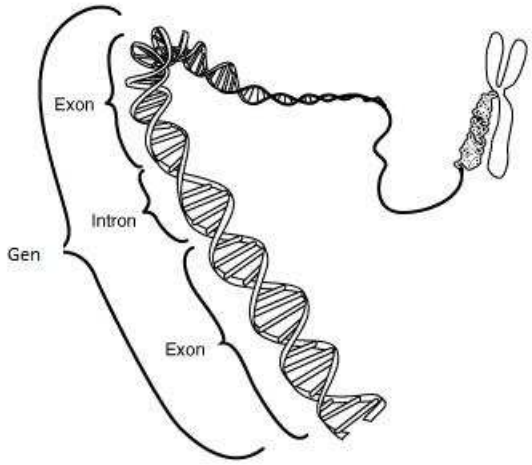

Figura 1.3: Estructura de un gen.

Entre los métodos más importantes a la hora de la predicción o búsqueda de genes, se encuentran los modelos de Markov ocultos, *Hidden Markov Models* (HMM) [26] cuya utilización ha sido muy difundida en procesamiento y reconocimiento de secuencias de texto. Debido a la similitud existente entre las cadenas de ADN y las secuencias en el lenguaje, estas metodologías han podido ser adaptadas a la búsqueda de genes.

Como el problema de la predicción de genes consiste en, dada una secuencia de nucleótidos, encontrar las regiones de ésta que codifican en proteínas, puede definirse entonces como un problema de asignación de las etiquetas gen y región intergénica a cada nucleótido de la cadena.

Una secuencia de ADN puede pensarse como una palabra de un alfabeto de cuatro letras {A, C, G, T}, por lo que una secuencia de longitud n puede representarse entonces como $x = (x_1, x_2, \ldots, x_n)$ donde cada x_j toma los valores A, C, G o T.

Los HMM son modelos generativos en los cuales se representa la probabilidad conjunta de la estructura oculta de genes y y la secuencia observada de nucleótidos x:

$$\mathbb{P}(x, y) = \prod_{i=1}^{n} \mathbb{P}(y_i|y_{i-1})\, \mathbb{P}(x_i|y_i)$$

El modelo genera la secuencia de nucleótidos x y la secuencia de etiquetas y está formada por los estados correspondientes a cada nucleótido. Luego, para una secuencia x dada, puede calcularse el camino

de estados y que la genera con mayor probabilidad. Por ejemplo, dada la secuencia de nucleótidos $(x_1, x_2, x_3, x_4, x_5)$, una vez ajustado el modelo se puede obtener una secuencia de etiquetas como (gen, gen, gen, intergénica, intergénica) lo cual indica que las tres primeras observaciones forman parte de un gen y las dos siguientes de la región intergénica.

El número de etiquetas o estados del modelo depende del problema a resolver, por ejemplo puede ser de interés además de predecir la ubicación de los genes en una secuencia de ADN, identificar las regiones que corresponden a exones e intrones (Figura 1.3). En este caso el número de estados aumentaría a tres: exon, intron, intergénica.

Los HMM se generalizan dando lugar a los modelos de Markov ocultos generalizados, *Generalized Hidden Markov Models* (GHMM), donde el modelo no emite un nucleótido individual por estado sino un segmento o subsecuencia de nucleótidos correspondientes a la misma etiqueta.

Mejorar la precisión de los modelos de Markov ocultos implica incrementar su complejidad lo que redunda en falta de eficiencia, razón por la cual se han explorado otras técnicas como los campos aleatorios condicionados, *Conditional Random Fields* (CRF), introducidos por Lafferty et al en 2001 [22] para el procesamiento de textos.

Dado que en el problema de la búsqueda de genes la salida del modelo consiste en secuencias de etiquetas, corresponde entonces la utilización de CRF para secuencias, esto es, *Linear Chain Conditional Random Fields* (LCCRF). Un LCCRF modela la probabilidad condicional de la secuencia oculta de estados y dada la secuencia observada de nucleótidos x como

$$\mathbb{P}(y|x) = \frac{1}{Z_\lambda(x)} \prod_{i=1}^{n} \exp\left(\sum_{j=1}^{m} \lambda_j f_j(y_{i-1}, y_i, x, i) \right)$$

donde i indica la posición en la secuencia de observaciones x, λ_j son parámetros reales, f_j funciones características y Z_λ es una constante de normalización.

La parte fundamental en el diseño de un CRF es la elección de las funciones características f_j que, debido a la constante de normalización Z_λ, pueden ser funciones a valores reales cualesquiera y no necesariamente distribuciones de probabilidad, lo que permite incorporar información al modelo de una manera sencilla.

En el análisis de secuencias de ADN, los CRF pueden entonces introducir, en forma de funciones características, información adicional al modelo proveniente por ejemplo de secuencias alineadas de otros genomas o bien de *Expressed Sequence Tag* (EST), esto es, subsecuencias cortas de una secuencia de nucleótidos ya transcrita.

Los CRF trabajan entonces no sólo sobre la secuencia bajo estudio sino que incorporan otro tipo de información lo cual implica que sean modelos más eficientes que los HMM (que en definitiva pueden verse como un caso particular de éstos) pero más complejos y sin duda más costosos desde el punto de vista computacional.
En resumen, la gran diferencia entre los predictores de genes basados en HMM o en GHMM y aquellos cuyo soporte son los CRF, es que los primeros toman como entrada sólo la secuencia objetivo y los últimos incluyen además otras referencias codificadas en las funciones características.
Existen varios predictores de genes que utilizan CRF, a modo de ejemplo se pueden citar CONRAD (CONditional RAnDom Fields)[2] y CONTRAST (CONditionally TRAined Search for Transcripts)[3].
CONRAD presentado por DeCaprio et al [8] es un predictor de genes basado en *Semi-Markov Conditional Random Fields* (SMCRF) que incorpora información de ESTs y fue diseñado para predecir genes en *Cryptococcus neoformans* y *Aspergillus nidulans*.
CONTRAST (CONditionally TRAined Search for Transcripts) desarrollado por Gross et al [16] y probado para el genoma humano, incorpora datos de genomas informantes combinando además CRF con *Support Vector Machine* (SVM).

En esta tesis se aborda el problema de la modelización de secuencias, se estudian diferentes modelos aplicables a este tipo de problemas y se analiza una aplicación a un conjunto de datos reales.
En el capítulo 2 se describen los modelos de Markov ocultos (HMM), detallando los algoritmos para su entrenamiento y posterior inferencia.
En el tercer capítulo se introducen los modelos probabilísticos condicionados, en particular los modelos de Markov de máxima entropía (MEMM), presentando el Principio de máxima entropía y la representación de los MEMM como modelos exponenciales.
El capítulo 4 se centra en los Campos aleatorios condicionados (CRF), en particular en aquellos utilizados para modelar secuencias. Se incluye además de una descripción del modelo, la aplicación de éstos a la predicción de genes descrita en la literatura.
En el último capítulo se aborda el problema de la búsqueda de genes VSG en el genoma de *Trypanosoma brucei*, detallando la elección del modelo a aplicar, los experimentos llevados a cabo y los resultados obtenidos.

[2] http://www.broadinstitute.org/annotation/conrad/
[3] http://contra.stanford.edu/contrast

Capítulo 2

Modelos de Markov ocultos

2.1. El modelo

2.1.1. Definición del modelo

Los modelos de Markov ocultos, *Hidden Markov Model* (HMM), son modelos de Markov en tiempo discreto que se utilizan para modelar secuencias de observaciones.
La teoría básica de los HMM fue introducida por Leonard Baum a fines de la década de 1960 en sus publicaciones sobre funciones probabilísticas de cadenas de Markov [1], [2]. En 1989, Lawrence Rabiner publica un tutorial sobre HMM donde describe la teoría de Baum aplicada al reconocimiento del habla que constituye una referencia en el estudio de este tipo de modelos.

Dados una secuencia de observaciones $x = (x_1, x_2, \ldots, x_n)$, donde cada x_j es un símbolo proveniente de un alfabeto finito $\mathcal{A}$, y un conjunto también finito de estados o etiquetas $\mathcal{E}$, un HMM describe la forma en que se genera la secuencia de observaciones x y de estados $y = (y_1, y_2, \ldots, y_n)$ que corresponden a cada observación x_j, donde $y_j \in \mathcal{E}$ para $j = 1, \ldots, n$. Un HMM es por tanto un modelo generativo que representa la probabilidad conjunta $\mathbb{P}(x, y)$ de la secuencia de observaciones x y de estados y.
Como en general se conoce la secuencia de observaciones pero no la secuencia de estados, se puede decir que esta última se encuentra oculta, de ahí la denominación de modelo de Markov *oculto*.

Los HMM inicialmente utilizados en problemas de reconocimiento y extracción de información de secuencias de texto, habla y escritura, resultan aplicables en otras áreas, como por ejemplo en Bio-

informática, donde constituyen una herramienta para la búsqueda de genes. La predicción de genes es un problema que consiste en, dada una secuencia de ADN, asignar las etiquetas gen y no gen (o región intergénica) a cada nucleótido de la secuencia. Al ser x una secuencia de ADN se tiene que $\mathcal{A}$={A, C, G, T} y cada x_j representa un nucleótido de la cadena. El conjunto de estados podría ser por ejemplo $\mathcal{E} = \{\text{gen}, \text{región intergénica}\}$. En este problema el modelo de Markov oculto representa la probabilidad conjunta de la estructura oculta de genes y y la secuencia observada de nucleótidos x.

Un HMM modela el proceso por el cual se genera una secuencia de observaciones de la siguiente manera: el modelo comienza en un estado que es escogido según un vector de probabilidades de inicio, éste emite una observación, transita a un nuevo estado que emite una nueva observación y así hasta alcanzar el final.

Los parámetros del modelo son las probabilidades de transición de estados y las probabilidades de emisión de observaciones, parámetros que se estiman mediante el método de máxima verosimilitud (algoritmo de Baum - Welch). Una vez entrenado el modelo éste puede utilizarse, por ejemplo, para encontrar la secuencia de estados que más probablemente se corresponde con una secuencia de observaciones dada (algoritmo de Viterbi).

Un HMM consiste entonces de un conjunto finito de estados $\mathcal{E}$, un conjunto finito de símbolos $\mathcal{A}$ y dos conjuntos de parámetros: probabilidades de transición de estados y probabilidades de emisión de símbolos.
La probabilidad de transición del estado k al estado l se define como $\mathbb{P}\big(y_{i+1} = l | y_i = k\big)$, es decir la probabilidad condicional de que el modelo se encuentre en el estado l en la posición $i+1$ de la secuencia dado que en el lugar i se encontraba en el estado k. Por simplicidad a veces se nota la probabilidad de transición del estado en la posición i de la secuencia al estado en la posición $i+1$ como $\mathbb{P}\left(y_{i+1}|y_i\right)$.
Para cada estado k y para cada símbolo x_i se define la probabilidad de que el estado k emita el símbolo x_i, esto es $\mathbb{P}\left(x_i|y_i = k\right)$. Igual que en el caso de las probabilidades de transición se escribe esta probabilidad como $\mathbb{P}\left(x_i|y_i\right)$.

Un HMM genera una secuencia de símbolos paso a paso. El primer paso consiste en generar el estado de inicio con probabilidad $\mathbb{P}(y_1)$. Luego en cada paso, el modelo genera aleatoriamente un símbolo y se traslada hacia un nuevo estado hasta llegar al estado final. Tanto el símbolo como el

nuevo estado, sólo dependen del estado actual. Si el estado actual es y_i, el símbolo x_i se genera con probabilidad $\mathbb{P}(x_i|y_i)$ y se pasa al siguiente estado con probabilidad $\mathbb{P}(y_{i+1}|y_i)$.

En n pasos el modelo genera una secuencia $x = (x_1, x_2, \ldots, x_n)$ para lo cual transita por una secuencia de etiquetas o camino de estados $y = (y_1, y_2, \ldots, y_n)$. Para una longitud fija n el modelo define una distribución de probabilidad sobre todas las secuencias de observaciones x y sobre todos los posibles estados y. Luego la probabilidad de que el modelo transite por el camino y y genere la secuencia x está dada por

$$\mathbb{P}(x, y) = \mathbb{P}(y_1)\left(\prod_{i=1}^{n-1} \mathbb{P}(x_i|y_i)\,\mathbb{P}(y_{i+1}|y_i)\right)\mathbb{P}(x_n|y_n)$$

Es frecuente, para simplificar la notación, escribir la probabilidad de inicio $\mathbb{P}(y_1)$ como $\mathbb{P}(y_1|y_0)$, de esta forma la probabilidad conjunta se escribe como

$$\mathbb{P}(x, y) = \prod_{i=1}^{n} \mathbb{P}(y_i|y_{i-1})\,\mathbb{P}(x_i|y_i)$$

2.1.2. Supuestos del modelo

Supuesto de Markov

En las probabilidades de transición de estados definidas como $\mathbb{P}\big(y_{i+1}|y_i\big)$, se asume que el próximo estado sólo depende del estado actual, lo que recibe el nombre de *supuesto de Markov* de primer orden. Cuando el próximo estado depende de los j estados anteriores el modelo de Markov se dice de orden j y en ese caso las probabilidades de transición se definen como

$$\mathbb{P}\big(y_{i+1}|y_i, y_{i-1}, \ldots, y_{i-j+1}\big)$$

En este trabajo sólo se consideran modelos de Markov de primer orden.

Supuesto de modelo homogéneo

Se asume que las probabilidades de transición de estados son homogéneas, esto es, son independientes de su ubicación en la secuencia. Así la probabilidad de pasar de un estado al siguiente no depende de la posición de la secuencia en la cual ocurre esta transición.

Se supone que las probabilidades de emisión son también homogéneas.

Supuesto de independencia condicional de las secuencias de observaciones

En un HMM cada observación x_i sólo depende del estado y_i que la genera, $\mathbb{P}(x_i|y_i)$, por lo que cuando el modelo predice el valor de la etiqueta y_i no toma en cuenta las observaciones de la secuencia $x = (x_1, x_2, \ldots, x_n)$ a excepción de x_i. Se asume entonces que las observaciones $x_1, x_2, \ldots, x_n$ son

condicionalmente independientes.

2.1.3. Estructura gráfica del modelo

Los modelos probabilísticos pueden representarse de forma gráfica, donde cada nodo o vértice del grafo representa una variable aleatoria y las aristas que unen dos nodos las relaciones probabilísticas entre estas variables. El grafo representa la forma en la cual la distribución conjunta de todas las variables aleatorias puede descomponerse en un producto de factores donde cada uno depende sólo de un subconjunto de variables.

El concepto de independencia condicional de variables, que es representado en el grafo por la ausencia de arista entre estas variables, permite simplificar la estructura del modelo y los cálculos necesarios tanto en el entrenamiento como en la inferencia, dado que admite la posibilidad de factorizar una distribución de probabilidad compleja como un producto de probabilidades.

Dos variables a y b son independientes dada una tercera variable c si son independientes en su distribución condicional, esto es, si se verifica $\mathbb{P}(a, b|c) = \mathbb{P}(a|c)\mathbb{P}(b|c)$. Así, condicionando en c, la probabilidad conjunta de a y b se factoriza en el producto de las probabilidades marginales de a y de b (ambas condicionadas en c) y se representa con la ausencia de arista entre los nodos correspondientes a las variables a y b y aristas dirigidas desde el nodo c a los dos primeros (Figura 2.1).

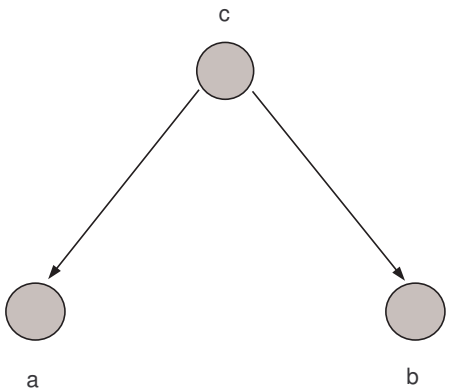

Figura 2.1: Independencia condicional de las variables a y b dada c.

Si $G = (V, E)$ es un grafo con vértices V y aristas E, los vértices (simbolizados por círculos) corresponden a las variables aleatorias que representan las observaciones x y las salidas y del modelo . Un HMM es un *grafo dirigido* donde la probabilidad conjunta $\mathbb{P}(x, y)$ puede ser escrita como producto de probabilidades condicionales de cada nodo $v_j \in V$ condicionado a sus nodos padres $v_j^p \in V$ o también como un producto de factores $\Psi_c(v_c)$ donde v_c es el subconjunto de variables aleatorias que constituyen cada factor:

$$\mathbb{P}(x, y) = \prod_{i=1}^{n} \mathbb{P}\left(y_i | y_{i-1}\right) \mathbb{P}\left(x_i | y_i\right) = \prod_j \mathbb{P}(v_j | v_j^p) = \prod_c \Psi_c(v_c)$$

Si por ejemplo la secuencia de observaciones es $x = (x_1, x_2, x_3)$, entonces HMM (Figura 2.2) representa la probabilidad conjunta como el siguiente producto de factores

$$\begin{aligned} \mathbb{P}(x_1, x_2, x_3, y_1, y_2, y_3) &= \mathbb{P}(y_1)\mathbb{P}(x_1|y_1)\mathbb{P}(y_2|y_1)\mathbb{P}(x_2|y_2)\mathbb{P}(y_3|y_2)\mathbb{P}(x_3|y_3) \\ &= \Psi_1(y_1)\Psi_2(x_1, y_1)\Psi_3(y_1, y_2)\Psi_4(x_2, y_2)\Psi_5(y_2, y_3)\Psi_6(x_3, y_3) \end{aligned}$$

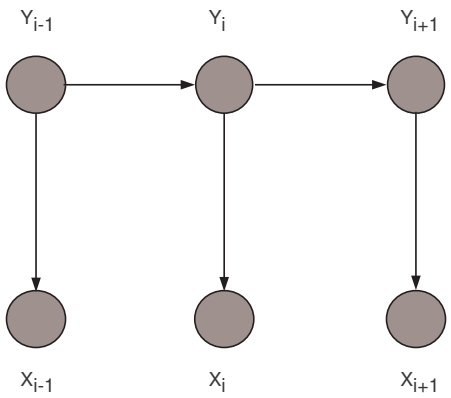

Figura 2.2: Estructura gráfica de un HMM.

Esta representación de un HMM resalta las relaciones de independencia condicional presentes en el modelo. La probabilidad del estado en la posición i sólo depende del estado anterior $i - 1$ y la observación generada en el lugar i sólo depende del estado en la ubicación i.

2.1.4. Elección del modelo

A la hora de elegir el modelo debe determinarse tanto el número de estados y de símbolos, como la estructura de las transiciones de la cadena de Markov (por ejemplo decidiendo si todas las transiciones

son posibles para el problema a modelar).

En el caso del análisis de una secuencia de ADN los símbolos representan las cuatro bases (A, C, T, G), y el número de estados depende del problema que se plantee resolver. Podría considerarse dos estados, por ejemplo gen y región intergénica, si se quiere predecir solamente aquellas regiones del genoma que se corresponden con genes o podría elegirse trabajar más en detalle definiendo además estados que reflejen otras estructuras.

2.2. Inferencia

Conocidos los parámetros del modelo, puede plantearse como objetivo encontrar la mejor secuencia de estados y que genera una secuencia dada de observaciones x de longitud n. Este proceso puede plantearse de varias formas dependiendo de qué se entienda como el mejor camino de estados.

2.2.1. Algoritmo de Viterbi

Si se considera que el mejor camino es aquel que con mayor probabilidad genera la secuencia dada, se utiliza una técnica de programación dinámica llamada *algoritmo de Viterbi*.

El camino de estados y que más probablemente genera una secuencia de observaciones x, se obtiene maximizando la probabilidad condicional de y dada x, esto es maximizando

$$\mathbb{P}(y|x) = \frac{\mathbb{P}(x,y)}{\mathbb{P}(x)}$$

Por lo tanto el camino óptimo es y^* tal que

$$y^* = \arg\max_y \frac{\mathbb{P}(x,y)}{\mathbb{P}(x)}$$

Ahora, como el denominador es constante para una secuencia dada x, lo que se necesita encontrar es el camino y^* que maximiza la probabilidad conjunta $\mathbb{P}(x,y)$.
Para cada lugar i en la secuencia de observaciones y para cada estado k, el algoritmo encuentra el camino de estados $y_1, y_2, \ldots, y_i$ que más probablemente genera la secuencia $x_1, x_2, \ldots, x_i$ y que finaliza en el estado k (o sea $y_i = k$).
Sea $V_i(k)$ la probabilidad del mejor camino hasta el estado k conocidas las primeras i observaciones. Luego para la posición $i+1$ de la secuencia se calcula la probabilidad del mejor camino que,

considerando las observaciones $x_1, x_2, \ldots, x_{i+1}$, termina en el estado l para cada l, es decir

$$V_{i+1}(l) = \max_k \left(V_i(k)\, \mathbb{P}\big(y_{i+1} = l | y_i = k\big) \right) \mathbb{P}\big(x_{i+1} | y_{i+1} = l\big)$$

El algoritmo comienza entonces calculando las probabilidades de que la observación x_1 sea generada por cada uno de lo estados y toma $V_1(k)$ como valor inicial. Luego en forma recursiva se calcula $V_i(l)$ para cada estado l y para cada $i = 2, \ldots, n$.

Como el objetivo es calcular la secuencia de estados que más probablemente genera la secuencia de observaciones dada, es necesario almacenar el argumento que corresponde al camino óptimo en cada paso i y para cada estado l. Se define entonces una nueva variable: $I_i(l)$.

La probabilidad $\mathbb{P}(x, y^*)$ es el máximo de $V_n(k)$ sobre todos los estados k, y la reconstrucción del camino más probable y^* se realiza rastreando hacia atrás (desde n hasta 1) el estado correspondiente $y^*_{i-1} = I_i(y^*_i)$.

Algoritmo de Viterbi

1. **inicialización**

   $V_1(k) = \mathbb{P}(y_1 = k)\mathbb{P}\big(x_1 | y_1 = k\big)$ `para todo` k

   $I_1(k) = 0$ `para todo` k

2. **iteración**

   `para` $i = 2$ `hasta` n`:`

   $V_i(l) = \max_k \left(V_{i-1}(k)\mathbb{P}\big(y_i = l | y_{i-1} = k\big) \right) \mathbb{P}\big(x_i | y_i = l\big)$

   $I_i(l) = \arg\max_k \left(V_{i-1}(k) P\big(y_i = l | y_{i-1} = k\big) \right)$

3. **finalización**

   $\mathbb{P}(x, y^*) = \max_k V_n(k)$

   $y^*_n = \arg\max_k V_n(k)$

4. **reconstrucción del camino de estados (***backtracking***)**

   `para` $i = n$ `hasta` $i = 1$`:`

   $y^*_{i-1} = I_i(y^*_i)$

En cuanto al número de operaciones a realizar por el algoritmo, el costo computacional es de orden $K^2 n$, siendo n la longitud de la secuencia y K el número de estados.

2.2.2. Decodificación a posteriori

El algoritmo de Viterbi encuentra el camino de estados que, con mayor probabilidad, genera una secuencia dada x. Ahora, muchos caminos en un HMM pueden generar la misma secuencia de símbolos. Consideremos, dada una secuencia x de longitud n, la probabilidad de que en la posición i se esté en el estado k, esto es, la probabilidad a posteriori $\mathbb{P}\big(y_i = k|x\big)$. Luego es posible, por ejemplo, encontrar el estado k con mayor probabilidad a posteriori para cada posición i de la secuencia, es decir

$$y_i^* = \arg\max_k \mathbb{P}\big(y_i = k|x\big)$$

a diferencia del algoritmo de Viterbi que calcula el camino más probable para la secuencia completa. Este proceso recibe el nombre de *decodificación a posteriori*.

Consideremos entonces la probabilidad de generar una secuencia donde el i-ésimo símbolo sea emitido por el estado k:

$$\begin{aligned}
\mathbb{P}(x, y_i = k) &= \mathbb{P}(x_1, \ldots, x_i, y_i = k)\mathbb{P}(x_{i+1}, \ldots, x_n|x_1, \ldots, x_i, y_i = k) \\
&= \mathbb{P}(x_1, \ldots, x_i, y_i = k)\mathbb{P}(x_{i+1}, \ldots, x_n|y_i = k) \\
&= \alpha_k(i)\beta_k(i)
\end{aligned}$$

donde $\alpha_k(i)$ es la probabilidad conjunta de generar los primeros i símbolos de la secuencia y terminar en el estado k (*probabilidad forward*) y $\beta_k(i)$ es la probabilidad condicional de emitir los símbolos restantes dado que en el ubicación i se está en el estado k (*probabilidad backward*). Luego

$$\mathbb{P}\big(y_i = k|x\big) = \frac{\mathbb{P}(x, y_i = k)}{\mathbb{P}(x)} = \frac{\alpha_k(i)\beta_k(i)}{\mathbb{P}(x)} \tag{2.1}$$

Las probabilidades *forward* y *backward* se calculan en forma recursiva

$$\begin{aligned}
\alpha_k(i) &= \mathbb{P}(x_1, \ldots, x_i, y_i = k) \\
&= \mathbb{P}\big(x_i|y_i = k\big) \sum_j \alpha_j(i-1)\mathbb{P}\big(y_i = k|y_{i-1} = j\big) \\
\beta_k(i) &= \mathbb{P}(x_{i+1}, \ldots, x_n|y_i = k) \\
&= \sum_l \mathbb{P}\big(x_{i+1}|y_{i+1} = l\big)\beta_l(i+1)\mathbb{P}\big(y_{i+1} = l|y_i = k\big)
\end{aligned}$$

utilizando para ello los llamados algoritmos de avance (o *forward*) y retroceso (o *backward*) respectivamente.

La probabilidad $\mathbb{P}(x)$ del denominador de la fórmula 2.1 se calcula también mediante el algoritmo *forward*:

$$\mathbb{P}(x) = \sum_k \alpha_k(n)$$

2.2.3. Algoritmo forward

El algoritmo *forward* comienza considerando la probabilidad conjunta de que el proceso inicie en el estado k y dicho estado genere la observación x_1, para cada estado k. En el paso i-ésimo de la recursión se calcula $\alpha_k(i)$, que representa la probabilidad conjunta que el proceso genere la sucesión $(x_1, x_2, \ldots, x_i)$ y que el estado y_i sea k, como la probabilidad de que el estado k emita la observación x_i por la suma, en todos los estados posibles j, del producto $\alpha_j(i-1)\mathbb{P}\big(y_i = k|y_{i-1} = j\big)$. El algoritmo finaliza con el cálculo de $\mathbb{P}(x)$ sumando todas las variables $\alpha_k(n)$.
La complejidad del algoritmo de orden K^2n, siendo n la longitud de la secuencia y K el número de estados.

Algoritmo forward

1. **inicialización**
   $\alpha_k(1) = \mathbb{P}\big(x_1, y_1 = k\big) = \mathbb{P}\big(y_1 = k\big)\mathbb{P}\big(x_1|y_1 = k\big)$ `para todo` k
2. **iteración**
   `para` $i = 2$ `hasta` n`:`
   $\alpha_k(i) = \mathbb{P}\big(x_i|y_i = k\big)\sum_j \alpha_j(i-1)\mathbb{P}\big(y_i = k|y_{i-1} = j\big)$
3. **finalización**
   $\mathbb{P}(x) = \sum_k \alpha_k(n)$

2.2.4. Algoritmo backward

En el primer paso del algoritmo *backward* se asigna el valor 1 a las K variables $\beta_k(n)$ (considerando K como el número de estados). En el paso inductivo $\beta_l(i+1)$ es la probabilidad de obtener la secuencia $(x_{i+2}, \ldots, x_n)$ dado que el estado en la ubicación $i+1$ es l. Ésta se multiplica por la probabilidad de pasar del estado k al l y por la probabilidad de que el estado l emita la observación x_{i+1}. Calculando este producto para cada uno de los l estados posibles y sumando se obtiene $\beta_k(i)$.
En el paso final se obtiene $\mathbb{P}(x)$ en la dirección contraria del algoritmo *forward*, multiplicando $\beta_k(1)$ por la probabilidad de comenzar en el estado k y la probabilidad de que el estado k emita la observación x_1.
El costo computacional, al igual que en algoritmo *forward*, es de orden K^2n, siendo n la longitud de

la secuencia y K el número de estados.

Algoritmo backward

1. **inicialización**

   $\beta_k(n) = 1$ `para todo` k

2. **iteración**

   `para` $i = n - 1$ `hasta 1:`

   $\beta_k(i) = \sum_l \mathbb{P}\big(x_{i+1}|y_{i+1} = l\big)\beta_l(i+1)\mathbb{P}\big(y_{i+1} = l|y_i = k\big)$

3. **finalización**

   $\mathbb{P}(x) = \sum_k \mathbb{P}(y_1|y_0 = k)\mathbb{P}\big(x_1|y_1 = k\big)\,\beta_k(1)$

2.2.5. Estabilidad numérica de los algoritmos

Todos los algoritmos presentados hasta el momento involucran cálculos con productos de probabilidades lo que puede dar como resultado valores numéricos muy pequeños e inestabilizar los algoritmos. Para subsanar este problema se puede trabajar con logaritmos de las probabilidades, o bien utilizar coeficientes de escalado.

Transformación logarítmica

En el algoritmo de Viterbi el problema de la inestabilidad puede solucionarse trabajando con logaritmos con lo cual los productos se convierten en sumas y se mejora la eficiencia. Así la relación de recurrencia del algoritmo se convierte en

$$\log V_i(l) = \max_k \left(\log V_{i-1}(k) + \log \mathbb{P}\big(y_i = l|y_{i-1} = k\big) \right) + \log \mathbb{P}\big(x_i|y_i = l\big)$$

La base del logaritmo puede ser cualquiera de las habituales: 2, e o 10.

Para los algoritmos *forward* y *backward* la utilización de logaritmos no es buena idea debido a que los cálculos en éstos involucrarían logaritmos de sumas de probabilidades lo que finalmente resulta costoso desde el punto de vista de la eficiencia computacional.

Coeficientes de escalado

Una alternativa a la utilización de logaritmos consiste en considerar coeficientes de escalado para las probabilidades *forward* y *backward* de forma tal que, en cada iteración, los resultados de las operaciones con estas probabilidades se mantengan dentro de cierto rango numérico. Para ello se definen variables de escalado s_i y las probabilidades en el paso i de los algoritmos *forward* y *backward* se calculan como:

$$\begin{aligned}\tilde{\alpha}_k(i) &= \frac{1}{s_i}\mathbb{P}\big(x_i|y_i=k\big)\sum_j \tilde{\alpha}_j(i-1)\mathbb{P}\big(y_i=k|y_{i-1}=j\big)\\ \tilde{\beta}_k(i) &= \frac{1}{s_i}\sum_l \mathbb{P}\big(x_{i+1}|y_{i+1}=l\big)\tilde{\beta}_l(i+1)\mathbb{P}\big(y_{i+1}=l|y_i=k\big)\end{aligned}$$

siendo los coeficientes s_i tales que $\sum_k \tilde{\alpha}_k(i) = 1$.

2.3. Entrenamiento

El entrenamiento del modelo consiste en, dada una secuencia x, determinar el parámetro θ (que representa las probabilidades de transición y de emisión) que maximiza $\mathbb{P}_\theta(x)$. En otras palabras, entrenar el modelo implica ajustar las probabilidades de transición y emisión del HMM para maximizar la verosimilitud de que la secuencia x sea generada por el modelo (*Principio de máxima verosimilitud*). Para determinar los parámetros $p_{kl} = \mathbb{P}\big(y_{i+1}=l|y_i=k\big)$ y $e_k(b) = \mathbb{P}\big(x_i=b|y_i=k\big)$ se calculan los correspondientes estimadores de máxima verosimilitud:

$$\hat{p}_{kl} = \frac{P_{kl}}{\sum_{l'} P_{kl'}} \qquad\qquad \hat{e}_k(b) = \frac{E_k(b)}{\sum_{b'} E_k(b')}$$

donde P_{kl} es número de transiciones de k a l y $E_k(b)$ el número de veces que el estado k emite el símbolo b en la secuencia de entrenamiento.

Ahora, si no se cuenta con la secuencia completa de estados en el conjunto de entrenamiento, no es posible calcular los estimadores de los parámetros de esta manera. En este caso se utilizan algoritmos, como el algoritmo de *Baum - Welch*, un caso particular de algoritmo EM (*Expectation-Maximization*).

2.3.1. Algoritmo de Baum - Welch

Un algoritmo EM permite encontrar estimadores de máxima verosimilitud de los parámetros en modelos probabilísticos, aun cuando en el conjunto de entrenamiento se tienen datos faltantes (en el caso de un HMM los datos faltantes serían los estados que se encuentran ocultos). EM alterna un paso

de cálculo del valor esperado de la verosimilitud (paso E) con un paso de cálculo de los estimadores de los parámetros del modelo que maximizan la verosimilitud esperada calculada en el paso anterior (paso M). Luego, con los parámetros recalculados en el paso M, se vuelve al paso E repitiendo este proceso hasta que se satisface cierto criterio de parada.

El algoritmo de Baum - Welch [32] comienza asignando valores iniciales a los parámetros, esto es a las probabilidades de transición y de emisión, que pueden o bien sortearse de una distribución uniforme o elegirse si se cuenta con algún conocimiento a priori que lo permita. En cada iteración se calcula el número esperado de veces en que cada transición y emisión es utilizada por el modelo para generar los datos de entrenamiento con los parámetros del paso anterior. Luego se actualizan las probabilidades de transición y emisión de forma tal que se maximice la verosimilitud de los valores esperados.

Intuitivamente el algoritmo de Baum - Welch podría describirse de la siguiente manera. En un comienzo no se conocen los parámetros que mejor ajustan al modelo por lo que se eligen en forma aleatoria, pero se dispone una o más secuencias de entrenamiento x que pueden ser utilizadas para maximizar $\mathbb{P}_\theta(x)$. Con los parámetros escogidos en el paso de inicio, se identifican las transiciones y emisiones más probables para el conjunto de entrenamiento, que suplantan entonces a los valores iniciales de las probabilidades de transición y emisión y como consecuencia se mejora el modelo, es decir se obtiene un modelo con mayor probabilidad de haber generado la secuencia dada. El proceso se repite hasta que no se observan mejoras en el modelo.

Si x es una secuencia de entrenamiento, la probabilidad de que la transición del estado k al estado l sea utilizada en la posición i de x es

$$\begin{aligned}
&\mathbb{P}\left(y_i = k, y_{i+1} = l|x\right) = \frac{\mathbb{P}\left(y_i = k, y_{i+1} = l, x\right)}{\mathbb{P}\left(x\right)} \\
= \;& \frac{\mathbb{P}\left(x_1, \ldots, x_i, y_i = k\right)\mathbb{P}\big(y_{i+1} = l|y_i = k\big)\mathbb{P}\big(x_{i+1}|y_{i+1} = l\big)\,\mathbb{P}\left(x_{i+2}, \ldots, x_n|y_{i+1} = l\right)}{\mathbb{P}\left(x\right)} \\
= \;& \frac{\alpha_k(i)\mathbb{P}\big(y_{i+1} = l|y_i = k\big)\mathbb{P}\big(x_{i+1}|y_{i+1} = l\big)\,\beta_l(i+1)}{\mathbb{P}\left(x\right)}
\end{aligned}$$

donde $\alpha_k(i) = \mathbb{P}\left(x_1, \ldots, x_i, y_i = k\right)$ y $\beta_l(i+1) = \mathbb{P}\left(x_{i+2}, \ldots, x_n|y_{i+1} = l\right)$ son las probabilidades *forward* y *backward* definidas anteriormente y la probabilidad del denominador se calcula mediante el algoritmo *forward*.

Luego, para calcular la proporción esperada de veces que se utiliza la transición del estado k al estado

l en la secuencia x, basta con sumar en todos los lugares i

$$\sum_i \mathbb{P}\left(y_i = k, y_{i+1} = l|x\right) = \sum_i \frac{\alpha_k(i)\mathbb{P}\big(y_{i+1} = l|y_i = k\big)\mathbb{P}\big(x_{i+1}|y_{i+1} = l\big)\,\beta_l(i+1)}{\mathbb{P}\left(x\right)}$$

De manera análoga se calcula la proporción de veces que el símbolo b es generado por el estado k:

$$\frac{\sum_{\{i:x_i=b\}} \alpha_k(i)\beta_k(i)}{\mathbb{P}(x)}$$

Supongamos se cuenta con un número T de secuencias de entrenamiento que llamamos x^t con $t = 1, \ldots, T$.

Por un lado las probabilidades de transición se estiman como el cociente entre el número esperado de transiciones del estado k al l y el número esperado de transiciones desde k en todas las secuencias x^t. Por otra parte las probabilidades de emisión son calculadas como el cociente entre el número esperado de veces que se transita por el estado k y se emite el símbolo b y el número esperado de veces que se pasa por k también en todas las secuencias de entrenamiento.

Luego los estimadores de los parámetros del modelo $\hat{p}_{kl}$ y $\hat{e}_k(b)$ son

$$\hat{p}_{kl} = \frac{P_{kl}}{\sum_{l'} P_{kl'}} \qquad\qquad \hat{e}_k(b) = \frac{E_k(b)}{\sum_{b'} E_k(b')}$$

donde

$$P_{kl} = \sum_t \frac{1}{P(x^t)} \sum_i \alpha_k^t(i) P\big(y_{i+1} = l|y_i = k\big) P\big(x_{i+1}^t|y_{i+1} = l\big)\,\beta_l^t(i+1)$$

$$E_k(b) = \sum_t \frac{1}{P(x^t)} \sum_{\{i:x_i^s=b\}} \alpha_k^t(i)\beta_k^t(i)$$

Ahora, si se suplantan los valores de los parámetros $\hat{\theta}_h$ ($\hat{p}_{kl}$ y $\hat{e}_k(b)$ en el paso h) por los calculados en el paso siguiente entonces, puede probarse que $P_{\hat{\theta}_{h+1}}(x) \geq P_{\hat{\theta}_h}(x)$, es decir, la verosimilitud del modelo aumenta con las sucesivas iteraciones.

El algoritmo llega a su fin o bien cuando la diferencia en las probabilidades de un paso a otro es imperceptible o cuando se alcanza el número de iteraciones fijado (criterio de parada).

El método de Baum - Welch presenta una fuerte dependencia de la elección inicial de los parámetros lo que implica que el algoritmo puede no encontrar el mejor modelo si la inicialización de las probabilidades de transición y emisión no resultan adecuadas.

Algoritmo Baum - Welch

1. **inicialización**

   se elige un valor arbitrario para θ

2. **iteración**

   a) cálculo de $\alpha_k^t(i)$, $\beta_k^t(i)$ para cada secuencia t, P_{kl} y $E_k(b)$

   b) cálculo de $\hat{p}_{kl}$ y $\hat{e}_k(b)$, es decir cálculo de $\hat{\theta}$ (nuevo modelo)

   c) cálculo de $\mathbb{P}_{\hat{\theta}}(x)$

   d) si $\mathbb{P}_{\hat{\theta}}(x) > \mathbb{P}_{\theta}(x)$ se considera $\theta = \hat{\theta}$ y se vuelve al paso a

3. **finalización**

   si $\mathbb{P}_{\hat{\theta}}(x) = \mathbb{P}_{\theta}(x)$ o se alcanza el número de iteraciones pautado

2.3.2. Prueba del algoritmo de Baum - Welch

El proceso iterativo en el algoritmo de Baum - Welch se basa en el aumento de la verosimilitud en cada paso, en esta sección se mostrará por qué la función de verosimilitud es creciente con el número de iteraciones.

Definición 2.3.1. Entropía relativa

La entropía relativa o distancia de Kullback-Leibler de dos distribuciones de probabilidad $\mathbb{P}_1$ *y* $\mathbb{P}_2$ *se define como*

$$d\big(\mathbb{P}_1(x), \mathbb{P}_2(x)\big) = \sum_i \mathbb{P}_1(x_i) \log\left(\frac{\mathbb{P}_1(x_i)}{\mathbb{P}_2(x_i)}\right)$$

La entropía relativa representa una medida no simétrica de la diferencia entre dos distribuciones. Es no negativa (Teorema 2.3.1) y toma el valor cero sólo si $\mathbb{P}_1(x_i) = \mathbb{P}_2(x_i)$ para todo i.

Teorema 2.3.1. Desigualdad de Gibbs

Si $\mathbb{P}_1$ *es la distribución de probabilidad de una variable aleatoria* x *y* $\mathbb{P}_2$ *es otra distribución de probabilidad discreta entonces*

$$d(\mathbb{P}_1(x), \mathbb{P}_2(x)) = \sum_i \mathbb{P}_1(x_i) \log\left(\frac{\mathbb{P}_1(x_i)}{\mathbb{P}_2(x_i)}\right) \geq 0$$

Demostración:

Si φ es una función convexa, la *desigualdad de Jensen* establece que $E\left(\varphi(y)\right) \geq \varphi\left(E(y)\right)$.

Dado que la función $\varphi(y) = -\log(y)$ es convexa, aplicando la *desigualdad de Jensen* a la variable aleatoria $y = \dfrac{\mathbb{P}_2(x)}{\mathbb{P}_1(x)}$ se tiene que

$$\sum_i \mathbb{P}_1(x_i) \log\left(\frac{\mathbb{P}_1(x_i)}{\mathbb{P}_2(x_i)}\right) \geq -\log \sum_i \mathbb{P}_1(x_i) \frac{\mathbb{P}_2(x_i)}{\mathbb{P}_1(x_i)} = -\log \sum_i \mathbb{P}_2(x_i) = 0$$

con lo cual queda demostrado que la entropía relativa es no negativa.

Teorema 2.3.2. La función Q

Sean x e y variables aleatorias y la función $Q(\theta, \theta') = \sum_y \mathbb{P}_\theta(y|x) \log \mathbb{P}_{\theta'}(x, y)$, donde θ y θ' representan los parámetros de dos modelos distintos. Entonces

$$Q(\theta, \theta') \geq Q(\theta, \theta) \;\Rightarrow\; \mathbb{P}_{\theta'}(x) \geq \mathbb{P}_\theta(x)$$

Demostración:

Como $\mathbb{P}_\theta(x) = \dfrac{\mathbb{P}_\theta\left(x, y\right)}{\mathbb{P}_\theta\left(y|x\right)}$, tomando logaritmos se tiene que

$$\log \mathbb{P}_\theta(x) = \log \frac{\mathbb{P}_\theta\left(x, y\right)}{\mathbb{P}_\theta\left(y|x\right)} = \log \mathbb{P}_\theta\left(x, y\right) - \log \mathbb{P}_\theta\left(y|x\right)$$

Multiplicando por $\mathbb{P}_\theta\left(y|x\right)$ y sumando en y:

$$\begin{aligned} \log \mathbb{P}_\theta(x) &= \sum_y \mathbb{P}_\theta\left(y|x\right) \log \mathbb{P}_\theta\left(x, y\right) - \sum_y \mathbb{P}_\theta\left(y|x\right) \log \mathbb{P}_\theta\left(y|x\right) \\ &= Q(\theta, \theta) - \sum_y \mathbb{P}_\theta\left(y|x\right) \log \mathbb{P}_\theta\left(y|x\right) \end{aligned}$$

Luego

$$\begin{aligned} \log \mathbb{P}_{\theta'}(x) - \log \mathbb{P}_\theta(x) &= Q(\theta, \theta') - \sum_y \mathbb{P}_\theta\left(y|x\right) \log \mathbb{P}_{\theta'}\left(y|x\right) - Q(\theta, \theta) + \sum_y \mathbb{P}_\theta\left(y|x\right) \log \mathbb{P}_\theta\left(y|x\right) \\ &= Q(\theta, \theta') - Q(\theta, \theta) + \sum_y \mathbb{P}_\theta\left(y|x\right) \log \frac{\mathbb{P}_\theta\left(y|x\right)}{\mathbb{P}_{\theta'}\left(y|x\right)} \end{aligned}$$

El último sumando resulta no negativo debido a la *desigualdad de Gibbs* (Teorema 2.3.1) por lo que $\log \mathbb{P}_{\theta'}(x) - \log \mathbb{P}_\theta(x) \geq Q(\theta, \theta') - Q(\theta, \theta)$, de donde se concluye que si $Q(\theta, \theta') \geq Q(\theta, \theta)$ entonces $\log \mathbb{P}_{\theta'}(x) \geq \log \mathbb{P}_\theta(x)$, es decir $\mathbb{P}_{\theta'}(x) \geq \mathbb{P}_\theta(x)$.

El algoritmo de Baum - Welch se fundamenta en el teorema anterior, puesto que, si se encuentra un valor del parámetro θ' para el cual se satisface la primera desigualdad en el Teorema 2.3.2, los datos de entrenamiento x serán más probables bajo el modelo de parámetro θ' que bajo el modelo de parámetro θ, es decir aumenta la verosimilitud.

Se observa que la igualdad en el Teorema 2.3.2 se alcanza sólo si $\theta = \theta'$ o si $\mathbb{P}_\theta(y|x) = \mathbb{P}_{\theta'}(y|x)$ para algún $\theta \neq \theta'$. Además si se elige el parámetro en el paso $h+1$ como $\theta^{h+1} = \arg\max_{\theta^h} Q(\theta, \theta^h)$ la verosimilitud del nuevo modelo resulta mayor o igual que la del modelo del paso anterior h.

El algoritmo EM comienza asignando un valor arbitrario a θ y repite los pasos E y M de manera alternada:

paso E: se toma el valor esperado de la variable aleatoria $\log \mathbb{P}_{\theta^h}(x, y)$ con respecto a la distribución $\mathbb{P}_\theta(y|x)$, esto es, se considera $Q(\theta, \theta^h) = \sum_y \mathbb{P}_\theta(y|x) \log \mathbb{P}_{\theta^h}(x, y)$.

paso M: se maximiza Q como función del argumento θ^h

A continuación se muestran los cálculos que permiten maximizar la función Q considerando, para simplificar y dado que la prueba es análoga en el caso general, que $\theta = p_{kl}$, es decir no se toma en cuenta la estimación de las probabilidades de emisión.

Teorema 2.3.3. Maximización de la función Q

El máximo de la función Q se alcanza en

$$p_{kl}^h = \frac{1}{\lambda_k \mathbb{P}_\theta(x)} \sum_i \alpha_k(i) \mathbb{P}_\theta(y_{i+1} = l | y_i = k) \mathbb{P}_\theta(x_{i+1} | y_{i+1} = l) \beta_l(i+1)$$

Demostración:

Si p_{kl}^h es la probabilidad de transición del estado k al l bajo el modelo de parámetro θ^h, para reestimar las probabilidades de transición debemos maximizar la función $Q(\theta, \theta^h)$ respecto de p_{kl}^h con la restricción $\sum_l p_{kl}^h = 1$.

La función de Lagrange asociada al problema es

$$L_\lambda(\theta^h) = Q(\theta, \theta^h) - \sum_k \lambda_k \left(\sum_l p_{kl}^h - 1 \right)$$

derivando con respecto a p^t_{kl} se obtiene

$$\frac{\partial}{\partial p^h_{kl}} L_\lambda(\theta^h) = \frac{\partial}{\partial p^h_{kl}} \left(Q(\theta, \theta^h) - \sum_k \lambda_k \left(\sum_l p^h_{kl} - 1 \right) \right) = \sum_y P_\theta(y|x) \frac{\frac{\partial}{\partial p^h_{kl}} P_{\theta^h}(x,y)}{P_{\theta^h}(x,y)} - \lambda_k$$

Dado que $P_{\theta^h}(x,y) = \prod_{i=1}^{n} \mathbb{P}_{\theta^h}(x_i|y_i)\, \mathbb{P}_{\theta^h}(y_i|y_{i-1}) = \prod_{i=1}^{n} \mathbb{P}_{\theta^h}(x_i|y_i)\, p^h_{i-1,\,i}$ la derivada con respecto a p^h_{kl} es

$$\begin{aligned}
\frac{\partial}{\partial p^h_{kl}} P_{\theta^h}(x,y) &= \frac{\partial}{\partial p^h_{kl}} \left(\prod_{i=1}^{n} \mathbb{P}_{\theta^h}(x_i|y_i) \prod_{(i-1,\,i)\neq(k,l)} p^h_{i-1,\,i} \prod_{(i-1,\,i)=(k,l)} p^h_{i-1,\,i} \right) \\
&= \prod_{i=1}^{n} \mathbb{P}_{\theta^h}(x_i|y_i) \prod_{(i-1,\,i)\neq(k,l)} p^h_{i-1,\,i} \frac{\partial}{\partial p^h_{kl}} \left(\left(p^h_{kl} \right)^{c_{kl}} \right) \\
&= \prod_{i=1}^{n} \mathbb{P}_{\theta^h}(x_i|y_i) \prod_{(i-1,\,i)\neq(k,l)} p^h_{i-1,\,i}\, c_{kl} \left(\left(p^h_{kl} \right)^{c_{kl}-1} \right) \\
&= \prod_{i=1}^{n} \mathbb{P}_{\theta^h}(x_i|y_i) \prod_{(i-1,\,i)\neq(k,l)} p^h_{i-1,\,i} \frac{c_{kl}}{p^h_{kl}} \left(\left(p^h_{kl} \right)^{c_{kl}} \right) \\
&= \prod_{i=1}^{n} \mathbb{P}_{\theta^h}(x_i|y_i)\, \frac{c_{kl}}{p^h_{kl}} \prod_{i=1}^{n} p^h_{i-1,\,i} \\
&= \frac{c_{kl}}{p^h_{kl}} \mathbb{P}_{\theta^h}(x,y)
\end{aligned}$$

siendo c_{kl} el número de veces que la transición del estado k al l se da en la secuencia de estados y. Entonces

$$\frac{\frac{\partial}{\partial p^h_{kl}} \mathbb{P}_{\theta^h}(x,y)}{\mathbb{P}_{\theta^h}(x,y)} = \frac{c_{kl}}{p^h_{kl}}$$

además el punto estacionario de la función de Lagrange es un punto donde la función alcanza un máximo pues

$$\frac{\frac{\partial^2}{\partial (p^h_{kl})^2} \mathbb{P}_{\theta^h}(x,y)}{\mathbb{P}_{\theta^h}(x,y)} = -\frac{c_{kl}}{(p^h_{kl})^2} < 0$$

Así

$$\frac{\partial}{\partial p^h_{kl}} L_\lambda(\theta^h) = 0 \;\Leftrightarrow\; \sum_y \mathbb{P}_\theta(y|x) \frac{c_{kl}}{p^h_{kl}} = \lambda_k \;\Leftrightarrow\; p^h_{kl} = \frac{1}{\lambda_k} \sum_y \mathbb{P}_\theta(y|x) c_{kl}$$

Si escribimos $\mathbb{P}_\theta(y|x) = \dfrac{\mathbb{P}_\theta(x,y)}{\mathbb{P}_\theta(x)}$ entonces $p^h_{kl} = \dfrac{1}{\lambda_k \mathbb{P}_\theta(x)} \displaystyle\sum_y \mathbb{P}_\theta(x,y) c_{kl}$.

Reescribiendo $c_{kl} = \sum_i 1_{\{(i,i+1)=(k,l)\}}$ y suplantando

$$\begin{aligned} p_{kl}^t &= \frac{1}{\lambda_k \mathbb{P}_\theta(x)} \sum_i \sum_y \mathbb{P}_\theta(x,y) 1_{\{(i,i+1)=(k,l)\}} \\ &= \frac{1}{\lambda_k \mathbb{P}_\theta(x)} \sum_i \mathbb{P}_\theta(x_1,\ldots,x_i,y_i=k)\mathbb{P}_\theta(y_{i+1}=l|y_i=k)\mathbb{P}_\theta(x_{i+1}|y_{i+1}=l)\mathbb{P}_\theta(x_{i+1},\ldots,x_n|y_{i+1}=l) \\ &= \frac{1}{\lambda_k \mathbb{P}_\theta(x)} \sum_i \alpha_k(i)\mathbb{P}_\theta(y_{i+1}=l|y_i=k)\mathbb{P}_\theta(x_{i+1}|y_{i+1}=l)\beta_l(i+1) \end{aligned}$$

se obtiene el valor de p_{kl}^h que maximiza $Q(\theta,\theta^h)$ sujeta a la restricción $\sum_l p_{kl}^h = 1$, esto es, se tiene la reestimación de la probabilidad de transición del estado k al l.

Una vez reestimados los parámetros θ^h, éstos se utilizan en la siguiente iteración, aumentando en cada paso el valor de la función $Q(\theta,\theta^h)$ y por lo tanto la verosimilitud $\mathbb{P}_{\theta^h}(x)$ resulta mayor que $\mathbb{P}_\theta(x)$ (Teorema 2.3.2).

2.3.3. Una alternativa al algoritmo de Baum - Welch

Una alternativa al algoritmo de Baum - Welch para entrenar el modelo es el algoritmo de entrenamiento de Viterbi. Éste se basa en la aplicación del algoritmo de Viterbi para el cálculo de los caminos de estados más probables para las secuencias de entrenamiento que luego son utilizados para la estimación de los parámetros del modelo [13].
El número de operaciones involucradas en algoritmo de Baum - Welch es mayor que en el algoritmo de entrenamiento de Viterbi, lo que redunda en un costo computacional más alto, sin embargo Baum - Welch brinda mejores resultados por lo que resulta más adecuado.

Capítulo 3

Modelos de Markov de máxima entropía

3.1. Introducción

Entre los métodos más utilizados para modelar secuencias de observaciones se encuentran los modelos de Markov ocultos (HMM), pero un HMM resulta a menudo inadecuado para describir situaciones reales. La propiedad de Markov de primer orden establece que la probabilidad de un estado sólo depende de la probabilidad del estado anterior

$$\mathbb{P}\big(y_{i+1}|y_1, y_2, \ldots, y_i\big) = \mathbb{P}\big(y_{i+1}|y_i\big)$$

lo que implica que un modelo basado en este supuesto no tiene en cuenta las relaciones entre dos estados no consecutivos y por lo tanto no modela muchas de las situaciones que se plantean en los problemas reales.
Otra desventaja es que en un HMM cada observación x_i de la secuencia x se genera en forma independiente, dependiendo sólo del estado y_i, no representando entonces posibles interacciones entre las observaciones. Se asume por lo tanto, que las observaciones $x_1, x_2, \ldots, x_i$ son condicionalmente independientes.

Cuando se plantea un modelo matemático para predecir o explicar determinado problema en cualquier área de la ciencia, como por ejemplo etiquetar una secuencia de observaciones, se busca que éste refleje las interacciones entre las observaciones sin asumir independencias condicionales no deseadas, pero también es necesario considerar un modelo que sea tratable desde el punto de vista de la

inferencia estadística a partir del cual pueda comprenderse o interpretarse el fenómeno bajo estudio. Los HMM se construyen sobre determinados supuestos para ser modelos manejables estadísticamente, pero presentan, en algunos casos, debilidades.

Una forma de sortear las desventajas que plantea un HMM consiste en considerar un *modelo de probabilidad condicional*, esto es, un modelo que calcule $\mathbb{P}(y|x)$, la probabilidad condicional de la secuencia de etiquetas y dada una secuencia de observaciones x, en lugar de la probabilidad conjunta $\mathbb{P}(x, y)$. Dado que los modelos de probabilidad condicional no representan la forma en que se generan las observaciones x, no es necesario imponer condiciones sobre las secuencias de observaciones.

En 2000 McCallum et al [23] proponen los modelos de Markov de máxima entropía, *Maximum Entropy Markov Models* (MEMM), para extracción de información y segmentación de textos.
Los MEMM son modelos probabilísticos condicionales no generativos basados en el *Principio de Máxima Entropía* (Jaynes, 1957) [18], en los que las probabilidades de transición y emisión de los HMM son reemplazadas por una única función $\mathbb{P}(y_i|y_{i-1}, x_i)$, que representa la probabilidad del estado y_i dados el estado anterior y_{i-1} y la observación x_i (Figure 3.1).
Las observaciones están dadas, por lo cual no debemos preocuparnos por calcular su probabilidad. Además en un MEMM, la observación en el lugar i de la secuencia depende no sólo del estado y_i sino también del estado anterior y_{i-1}, por lo que a diferencia de los HMM las observaciones están asociadas con las probabilidades de transición más que con los propios estados.
El modelo es entonces

$$\mathbb{P}(y|x) = \mathbb{P}(y_1|x_1) \prod_{i=2}^{n} \mathbb{P}(y_i|y_{i-1}, x_i)$$

o también

$$\mathbb{P}(y|x) = \prod_{i=1}^{n} \mathbb{P}(y_i|y_{i-1}, x_i)$$

considerando $\mathbb{P}(y_1|y_0, x_1) = \mathbb{P}(y_1|x_1)$

Las probabilidades condicionales de un estado dada una observación y el estado anterior, se representan por modelos exponenciales de acuerdo con los modelos de máxima entropía, como se detalla en las secciones que siguen.

Los problemas que tratan los HMM se resuelven también utilizando los MEMM para lo cual se consideran variantes de los algoritmos *forward-backward*, *Viterbi* y *Baum-Welch*.

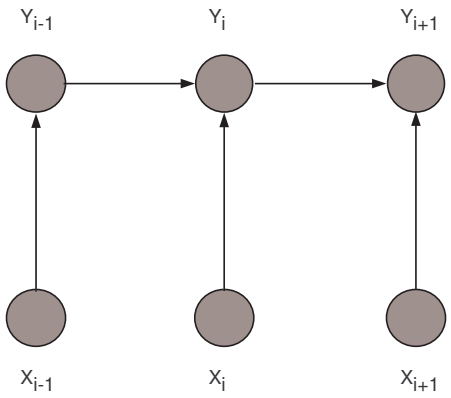

Figura 3.1: Estructura gráfica de un MEMM.

3.2. Principio de máxima entropía

3.2.1. El Principio de máxima entropía

El principio de máxima entropía es una herramienta para estimar la distribución de probabilidad de un conjunto de datos. Se basa en la idea de que el mejor modelo para los datos es aquel que asume el menor número de supuestos posibles, pero es consistente con las limitaciones o restricciones que presenta el conjuntos de las observaciones. Una distribución de probabilidad que considera un número reducido de supuestos es una distribución cercana a la uniforme, que es la distribución de máxima entropía. Por esta razón se busca estimar aquella distribución de probabilidad que maximiza la entropía y refleja además las características de los datos de entrenamiento.

El principio fue expuesto por primera vez por Edwin T. Jaynes en 1957 [18], quien en un trabajo publicado en 1990 [3], [19], explica el concepto de la distribución de máxima entropía de la siguiente manera:

> *... the fact that a certain probability distribution maximizes entropy subject to certain constraints representing our incomplete information, is the fundamental property which justifies use of that distribution for inference; it agrees with everything that is known, but carefully avoids assuming anything that is not known. It is a transcription into mathematics of an ancient principle of wisdom...*

3.2.2. Distribuciones de probabilidad de entropía máxima

Definición 3.2.1. Entropía

Si X es una variable aleatoria discreta con valores $x_1, x_2, \ldots, x_n$ y distribución de probabilidad $\mathbb{P}(X = x_i) = p_i \quad \forall i = 1, \ldots, n$, se define la entropía de Shannon como

$$H(p) = -\sum_{i=1}^{n} p_i \ln(p_i)$$

considerando $0 \log 0 = 0$.

La entropía $H(p)$ (o también $H(X)$) es una medida de uniformidad o de incertidumbre de una distribución de probabilidad, así, cuando la entropía es alta, resulta difícil predecir los valores de la variable X debido a que estos tienen probabilidades similares, mientras que si existen valores de X que son mucho más probables que otros la entropía es baja.
Por ejemplo, para una variable X con una distribución Bernoulli de parámetro p, se tiene

$$H(p) = -(1-p)\ln(1-p) - p\ln(p)$$

Si $p = 0$ o $p = 1$ entonces $H(p) = 0$. Para $0 < p < 1$ se tiene que

$$H'(P) = \ln(1-p) - \ln(p) = \ln\left(\frac{1-p}{p}\right) = 0 \;\Leftrightarrow\; \frac{1-p}{p} = 1 \;\Leftrightarrow\; p = \frac{1}{2}$$

por lo que, dado que $H''(p) < 0$ para p tal que $0 < p < 1$, se tiene que $H(p)$ alcanza su máximo en $p = \frac{1}{2}$; lo que se corresponde con la noción intuitiva de que la mayor incertidumbre se presenta cuando los dos posibles valores de la variable tienen la misma probabilidad de ocurrir.

Teorema 3.2.1. Distribución de entropía máxima

Sea X una variable aleatoria discreta con valores $x_1, x_2, \ldots, x_n$ y distribución de probabilidad $\mathbb{P}(X = x_i) = p_i \quad \forall i = 1, \ldots, n$. Entonces la función de probabilidad que maximiza la entropía es aquella tal que $p_i = \frac{1}{n} \quad \forall\, i = 1, \ldots, n$.

Demostración:
Se quiere maximizar la función $H(p) = -\sum_{i=1}^{n} p_i \ln(p_i)$ sujeta a la restricción $\sum_{i=1}^{n} p_i = 1$.
La función de Lagrange asociada al problema es

$$L_\lambda(p) = -\sum_{i=1}^{n} p_i \ln(p_i) + \lambda\left(\sum_{i=1}^{n} p_i - 1\right)$$

cuya derivada respecto de p_i para cada $i = 1, \ldots, n$ es

$$\frac{\partial}{\partial p_i} L_\lambda(p) = -\left(\ln(p_i) + 1\right) + \lambda = 0 \;\Leftrightarrow\; p_i = e^{-1+\lambda}$$

Luego, sustituyendo en la restricción se obtiene

$$\sum_{i=1}^{n} p_i = n e^{-1+\lambda} = 1 \;\Leftrightarrow\; p_i = \frac{1}{n}$$

Como la función es cóncava dado que $\dfrac{\partial^2}{\partial p_i^2} L_\lambda(p) = -\dfrac{1}{p_i} < 0$, se deduce que en el punto obtenido se presenta un máximo.

Si se conoce además el valor esperado de una función de las observaciones f, es decir, si se introduce la condición $E\left(f(X)\right) = \sum_{i=1}^{n} f(x_i) p_i = \gamma$, el problema se resuelve agregando un multiplicador de Lagrange para la nueva restricción y derivando como en el Teorema 3.2.1.

La nueva función de Lagrange es

$$L_{\lambda_0,\lambda_1}(p) = -\sum_{i=1}^{n} p_i \ln(p_i) + (\lambda_0 + 1)\left(\sum_{i=1}^{n} p_i - 1\right) + \lambda_1\left(\sum_{i=1}^{n} f(x_i) p_i - \gamma\right)$$

de donde

$$\frac{\partial}{\partial p_i} L_{\lambda_0,\lambda_1}(p) = -\left(\ln(p_i) + 1\right) + \lambda_0 + 1 + \lambda_1 f(x_i) = 0 \;\Leftrightarrow\; p_i = \exp\left(\lambda_0 + \lambda_1 f(x_i)\right)$$

Para calcular los multiplicadores se sustituye p_i en las restricciones obteniendo

$$\sum_{i=1}^{n} \exp\left(\lambda_0 + \lambda_1 f(x_i)\right) = 1$$

$$\sum_{i=1}^{n} f(x_i) \exp\left(\lambda_0 + \lambda_1 f(x_i)\right) = \gamma$$

Finalmente

$$\lambda_0 = -\ln\left(\sum_{i=1}^{n} \exp(\lambda_1 f(x_i))\right) = -\ln\left(Z_{\lambda_1}(X)\right)$$

$$\gamma = \frac{\sum_{i=1}^{n} f(x_i) \exp\left(\lambda_1 f(x_i)\right)}{\sum_{i=1}^{n} \exp\left(\lambda_1 f(x_i)\right)} = \frac{\sum_{i=1}^{n} f(x_i) \exp\left(\lambda_1 f(x_i)\right)}{Z_{\lambda_1}(X)}$$

donde $Z_{\lambda_1}(X) = \sum_{i=1}^{n} \exp(\lambda_1 f(x_i))$ se denomina *función partición* [18]. Entonces

$$p_i = \exp\left(\lambda_0 + \lambda_1 f(x_i)\right) = \exp(\lambda_0)\exp(\lambda_1 f(x_i)) = \frac{1}{Z_{\lambda_1}(X)} \exp(\lambda_1 f(x_i))$$

Este resultado se generaliza de manera análoga cuando se cuenta con un número m de funciones f de las cuales se conoce el valor esperado.

Teorema 3.2.2. Modelo exponencial de entropía máxima

Sea X una variable aleatoria discreta con valores $x_1, x_2, \ldots, x_n$ y distribución de probabilidad $\mathbb{P}(X = x_i) = p_i \quad \forall i = 1, \ldots, n$. Si f_j es tal que $E(f_j(X)) = \gamma_j \quad \forall j = 1, \ldots, m$ entonces, la función de probabilidad que maximiza la entropía es aquella definida por

$$p_i = \frac{1}{Z_\lambda(X)} \exp\Big(\lambda_1 f_1(x_i) + \ldots + \lambda_m f_m(x_i)\Big) = \frac{1}{Z_\lambda(X)} \exp\Big(\sum_{j=1}^{m} \lambda_j f_j(x_i)\Big)$$

donde $Z_\lambda(X) = \sum_{i=1}^{n} \exp\Big(\sum_{j=1}^{m} \lambda_j f_j(x_i)\Big)$ y las constantes se determinan de las ecuaciones

$$\begin{aligned} \gamma_j &= \frac{1}{Z_\lambda(X)} \sum_{i=1}^{n} f_j(x_i) \exp\Big(\sum_{j=1}^{m} \lambda_j f_j(x_i)\Big) \\ \lambda_0 &= -\ln(Z_\lambda(X)) \end{aligned}$$

Si lo que se quiere maximizar es la entropía condicional $H(y|x)$, definida como

$$H(y|x) = -\sum_{(x,y)} \mathbb{P}(x, y) \log \mathbb{P}(y|x)$$

el modelo resulta

$$\mathbb{P}(y|x) = \frac{1}{Z_\lambda(x)} \exp\Big(\sum_{j=1}^{m} \lambda_j f_j(x, y)\Big)$$

donde

$$Z_\lambda(x) = \sum_{y} \exp\Big(\sum_{j=1}^{m} \lambda_j f_j(x, y)\Big)$$

3.3. El modelo

Los HMM utilizan por un lado probabilidades de transición de estados y por otro probabilidades condicionales de las observaciones dados los estados, mientras que, en los MEMM, sólo se plantea la transición del estado en la posición i dados el estado en la ubicación anterior $i - 1$ y la observación x_i. Esto permite modelar las transiciones en términos de características independientes de las observaciones, lo cual se lleva a cabo ajustando modelos exponenciales que maximizan la entropía.

Para construir un modelo que incluya de forma precisa la información contenida en el conjunto de entrenamiento se necesita representar de alguna forma ese conocimiento sobre los datos, lo que puede hacerse incluyendo restricciones en el modelo que expresen características de éstos. McCallum et al [23], consideran restricciones basadas en *funciones características* binarias que dependen tanto de la observación x_i como del estado y_i y son de la forma

$$f_{\langle g,k\rangle}(x_i, y_i) = \begin{cases} 1 & \text{si } g(x_i) \text{ es verdadero ; } y_i = k \\ 0 & \text{en otro caso} \end{cases}$$

donde g es una función que toma el valor 1 cuando al evaluarla en x_i resulta verdadera y 0 cuando resulta falsa.
Así cada función f_j indica cuándo una característica binaria g es verdadera para la observación x_i y el estado en la posición i es k.

Por ejemplo en caso de modelar secuencias de texto, podría considerarse $g(x_i) = 1$ si la palabra x_i está escrita en mayúsculas y 0 en otro caso. Así una posible función característica es aquella que toma el valor 1 si la observación en la posición i de la cadena está escrita en letras mayúsculas y el estado correspondiente es nombre propio:

$$f(x_i, y_i) = \begin{cases} 1 & \text{si } g(x_i) \text{ es verdadera , } y_i = \text{nombre propio} \\ 0 & \text{en otro caso} \end{cases}$$

De esta manera, considerando un conjunto de funciones características de este tipo, se resume la información relevante contenida en el conjunto $\mathcal{T}$ de datos de entrenamiento.

Ahora, para que el modelo refleje la información contenida en los datos, el valor esperado de cada función característica f_j en la distribución estimada

$$E(f_j) = \sum_{(x,y)} \mathbb{P}(x, y) f_j(x, y)$$

debe coincidir con su promedio en el conjunto de entrenamiento $\mathcal{T}$, es decir $E(f_j) = \tilde{E}(f_j)$ siendo

$$\tilde{E}(f_j) = \sum_{(x,y)} \tilde{\mathbb{P}}(x, y) f_j(x, y) = \frac{1}{T} \sum_{(x,y)\in\mathcal{T}} f_j(x, y) \tag{3.1}$$

La distribución empírica $\tilde{\mathbb{P}}$ se calcula como el cociente entre el número de veces que ocurre (x, y) en el conjunto de entrenamiento $\mathcal{T}$ y el cardinal de éste, T. Como $\tilde{\mathbb{P}}(x, y) = 0$ si (x, y) no pertenece a $\mathcal{T}$,

se tiene la segunda igualdad en la ecuación 3.1.

Si se aproxima $P(x)$ por la empírica $\tilde{\mathbb{P}}(x)$ se obtiene $\mathbb{P}(x,y) = \mathbb{P}(x)\mathbb{P}(y|x) \sim \tilde{\mathbb{P}}(x)\mathbb{P}(y|x)$ de donde

$$E(f_j) \sim \frac{1}{T} \sum_{x \in \mathcal{T}} \sum_{y} \mathbb{P}(y|x) f_j(x,y)$$

Las condiciones $E(f_j) = \tilde{E}(f_j)$, junto con $\mathbb{P}(y|x) \geq 0$ y $\sum_y \mathbb{P}(y|x) = 1 \; \forall \, x$, conforman las restricciones del problema y son de la misma forma que las planteadas en el Teorema 3.2.2. Luego, la distribución de máxima entropía para estas condiciones es

$$\mathbb{P}\left(y_i|y_{i-1}, x_i\right) = \frac{1}{Z_\lambda(x_i, y_{i-1})} \exp\left(\sum_j \lambda_j f_j(x_i, y_i)\right)$$

donde los λ_j son parámetros a estimar y Z_λ es una constante de normalización que se calcula como $Z_\lambda(x_i, y_{i-1})) = \sum_{y_i} \exp\left(\sum_j \lambda_j f_j(x_i, y_i)\right)$

Para estimar los parámetros de cada modelo exponencial se utiliza una técnica denominada *Generalized Iterative Scaling*, [23].

Al igual que los HMM, los MEMM se utilizan para etiquetar nuevos datos identificando la secuencia de etiquetas que mejor describe la nueva secuencia de observaciones, para lo cual se busca el camino de estados más probable para la secuencia de observaciones dada. McCallum et al [23], presentan para ello una variante del algoritmo de Viterbi.

3.4. Desventajas de los modelos de máxima entropía

Los modelos de Markov de entropía máxima presentan una debilidad que Lafferty et al [22], llaman *label bias problem* y exponen mediante el siguiente ejemplo.

Supongamos se quiere analizar las secuencias (`r,i,b`) y (`r,o,b`), para lo cual se plantea el modelo MEMM dado por

$$\mathbb{P}\left(y|x\right) = \prod_{i=1}^{3} \mathbb{P}\left(y_i|y_{i-1}, x_i\right)$$

donde

$$\mathbb{P}\left(y_i|y_{i-1}, x_i\right) = \frac{1}{Z_\lambda(x_i, y_{i-1})} \exp\left(\sum_j \lambda_j f_j(x_i, y_i)\right)$$

siendo Z_λ es una constante de normalización y se busca la secuencia de estados y que maximiza la probabilidad de la secuencia dada x.

En la Figura 3.2 (Lafferty et al, [22]) se visualizan las probabilidades de transición del modelo, en cada nodo del grafo las aristas dirigidas indican una posible transición al nodo siguiente. La suma de las probabilidades de salida en cada nodo debe ser igual a 1.

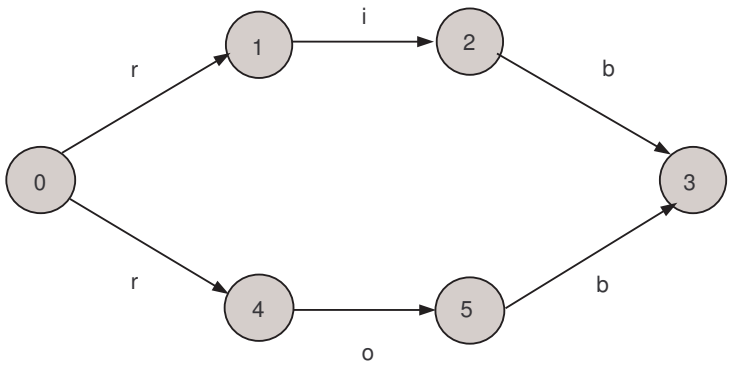

Figura 3.2: Ejemplo de *label bias problem*.

Supongamos que se quiere encontrar el camino de estados más probable para la secuencia de observaciones $x = (\texttt{r},\texttt{i},\texttt{b})$. En un primer paso observamos `r`, dado que desde el estado inicial 0 es posible pasar al estado 1 o al estado 4 al condicionar en `r`, se tiene que la probabilidad se divide en dos partes $\mathbb{P}(1|0,\texttt{r})$ y $\mathbb{P}(4|0,\texttt{r})$ que verifican $\mathbb{P}(1|0,\texttt{r}) + \mathbb{P}(4|0,\texttt{r}) = 1$ pues la suma desde todos los estados alcanzables desde 0 (en este caso 1 y 4) debe sumar 1.
En un segundo paso observamos `i`. Como las probabilidades de todos los estados alcanzables desde 1 (solamente el estado 2) y desde 4 (sólo el estado 5) deben sumar 1, tenemos que $\mathbb{P}(2|1,\texttt{i}) = 1$ y $\mathbb{P}(5|4,\texttt{i}) = 1$. Obviamente la probabilidad de un evento que no ocurre en el conjunto de entrenamiento debe ser casi nula, en este caso los datos de entrenamiento no contienen ninguna transición del estado 4 al 5 en la observación `i` y, sin embargo, $\mathbb{P}(5|4,\texttt{i}) = 1$.
Por lo tanto los estados con sólo una transición posible ignoran la observación a la cual están condicionados. En el ejemplo, los caminos de estados (0,1,2,3) y (0,4,5,3) resultan igualmente probables independientemente de la secuencia de observaciones dada.

La exigencia de que la suma de las probabilidades de los estados alcanzables desde uno dado sea igual a 1, implica que las observaciones sólo influyen en la elección del estado siguiente, pero no en la probabilidad de la transición.
Esto da como resultado la existencia de un sesgo hacia los estados con transición de baja entropía,

modelo	error	error fdv
HMM	5.69 %	45.99 %
MEMM	6.37 %	54.61 %

Cuadro 3.1: Tasa de error por palabra y tasa de error por palabra fuera de vocabulario para POS usando HMM y MEMM.

por lo que aquellos estados que se suceden de estados de baja entropía no toman en cuenta las observaciones y, en el caso extremo de los estados con una única transición de salida, la observación es directamente omitida.

Si bien en muchos casos se obtienen mejores resultados al aplicar un MEMM que al utilizar un HMM [23], es posible que la performance de los MEMM sea peor que las de los HMM cuando ocurre el problema *label bias*. Lafferty et al [22] realizaron varios experimentos para comparar estos modelos en aplicaciones de análisis de textos (*Part Of Speech (POS)*) donde a cada palabra en una frase se le debe asignar una de 45 etiquetas posibles. Los resultados obtenidos con HMM son mejores que aquellos logrados con MEMM, como consecuencia del problema *label bias* y se visualizan en el Cuadro 3.1.

Capítulo 4

Campos aleatorios condicionados

4.1. Introducción

Los campos aleatorios condicionados, *Conditional Random Fields* (CRF), presentados en 2001 por Lafferty et al [22], son modelos probabilísticos utilizados para segmentar y etiquetar secuencias de observaciones. Presentan ventajas sobre los modelos de Markov ocultos (HMM) puesto que relajan las fuertes hipótesis de independencia condicional que éstos asumen y también sobre los modelos de Markov de máxima entropía (MEMM) ya que superan el problema denominado *label bias problem*.

Los HMM son modelos probabilísticos generativos que modelan la probabilidad conjunta de las observaciones y la secuencia de etiquetas asumiendo, para hacer el modelo tratable desde el punto de vista del entrenamiento y de la inferencia, hipótesis como la independencia condicional de las observaciones.

Los MEMM son modelos probabilísticos condicionales que modelan la probabilidad condicional de la secuencia de etiquetas dada la secuencia de observaciones, utilizando para ello un modelo exponencial para cada estado de acuerdo con el principio de máxima entropía. En los MEMM las observaciones sólo influyen en la elección del estado siguiente, pero no en la probabilidad de la transición, lo que da como resultado la existencia de un sesgo hacia los estados con transición de baja entropía.

Los CRF son también modelos condicionales que definen la probabilidad condicional $\mathbb{P}(y|x)$ de la secuencia de etiquetas y dada una secuencia de observaciones x. Como en todos los modelos condicionales, en un CRF no se modela la distribución de las observaciones y por lo tanto no es necesario

imponer condiciones sobre éstas, teniendo entonces libertad de incluir en el modelo características arbitrarias de las mismas. Los CRF se representan mediante grafos no dirigidos (o Campos aleatorios de Markov), en los cuales no se necesita definir para cada estado la distribución de los estados siguientes dado el actual, sino que se define la distribución de la secuencia de etiquetas en su totalidad dada la secuencia de observaciones.

4.2. Campos aleatorios condicionados para secuencias

Dentro de la familia de modelos CRF, los más utilizados son aquellos con una estructura de cadena lineal y que se denominan campos aleatorios condicionados para secuencias o campos aleatorios condicionados con estructura de cadena lineal, *Linear Chain Conditional Random Field* (LCCRF).

4.2.1. El modelo

Un HMM modela la probabilidad conjunta de la secuencia de observaciones $x = (x_1, x_2, \ldots, x_n)$ y de etiquetas $y = (y_1, y_2, \ldots, y_n)$, con la convención $\mathbb{P}(y_1|y_0) = \mathbb{P}(y_1)$, como

$$\mathbb{P}(x, y) = \prod_{i=1}^{n} \mathbb{P}(y_i|y_{i-1})\mathbb{P}(x_i|y_i)$$

Si como en Sutton y McCallum (2007) [27] se definen los parámetros $\lambda_{kl} = \ln \mathbb{P}(y_i = l|y_{i-1} = k)$ y $\mu_{bl} = \ln \mathbb{P}(x_i = b|y_i = l)$, la probabilidad anterior resulta

$$\mathbb{P}(x, y) = \frac{1}{Z} \exp\left(\sum_i \sum_{k,l} \lambda_{kl} \mathbf{1}_{\{y_i=l\}} \mathbf{1}_{\{y_{i-1}=k\}} + \sum_i \sum_l \sum_b \mu_{bl} \mathbf{1}_{\{y_i=l\}} \mathbf{1}_{\{x_i=b\}} \right)$$

siendo Z una constante de normalización.

Si se introducen, al igual que en los modelos de máxima entropía, *funciones características* de la forma $f_j(y_{i-1}, y_i, x_i)$, donde se consideran para cada transición (k, l) y para cada par estado-observación (l, b) las funciones

$$f_{kl}(y_i, y_{i-1}, x_i) = \mathbf{1}_{\{y_{i-1}=k\}} \mathbf{1}_{\{y_i=l\}} = \begin{cases} 1 & \text{si } y_i = l, y_{i-1} = k \\ 0 & \text{en otro caso} \end{cases}$$

$$f_{lb}(y_i, y_{i-1}, x_i) = \mathbf{1}_{\{y_i=l\}} \mathbf{1}_{\{x_i=b\}} = \begin{cases} 1 & \text{si } y_i = l, x_i = b \\ 0 & \text{en otro caso} \end{cases}$$

entonces la probabilidad conjunta en un HMM es

$$\mathbb{P}(x,y) = \frac{1}{Z}\exp\left(\sum_i \sum_j \lambda_j f_j(y_i, y_{i-1}, x_i)\right)$$

Luego

$$\mathbb{P}(y|x) = \frac{\mathbb{P}(x,y)}{\sum_{y'}\mathbb{P}(x,y')} = \frac{\exp\left(\sum_i \sum_j \lambda_j f_j(y_i, y_{i-1}, x_i)\right)}{\sum_{y'}\exp\left(\sum_i \sum_j \lambda_j f_j(y'_i, y'_{i-1}, x_i)\right)}$$

Esta distribución condicional es un caso particular de LCCRF, en el cual sólo se incluye la información de la posición i de la secuencia de observaciones (x_i), pero un LCCRF puede utilizar información más completa de la secuencia de observaciones x permitiendo que las funciones características f_j sean más generales.

Un LCCRF modela la probabilidad condicional $\mathbb{P}(y|x)$ como

$$\mathbb{P}(y|x) = \frac{1}{Z_\lambda(x)}\exp\left(\sum_{i=1}^{n}\sum_{j=1}^{m} \lambda_j f_j(y_{i-1}, y_i, x, i)\right)$$

donde i indica la posición en la secuencia de observaciones x, λ_j son parámetros reales, f_j funciones características y Z_λ es una constante de normalización

$$Z_\lambda(x) = \sum_y \exp\left(\sum_{i=1}^{n}\sum_{j=1}^{m} \lambda_j f_j(y_{i-1}, y_i, x, i)\right)$$

Luego la forma general de un LCCRF es:

$$\mathbb{P}(y|x) = \frac{1}{Z_\lambda(x)}\prod_{i=1}^{n}\exp\left(\sum_{j=1}^{m} \lambda_j f_j(y_{i-1}, y_i, x, i)\right)$$

Las funciones características $f_j(y_{i-1}, y_i, x, i)$ son funciones a valores reales cualesquiera, por ejemplo el valor de una función f_j en la posición i puede depender del valor de las observaciones x en una ubicación distante de i, lo cual permite al modelo captar interacciones de largo alcance.

Para calcular la constante de normalización $Z_\lambda(x)$ se debe sumar en todos las posibles secuencias de estados para lo cual se utiliza un algoritmo *forward-backward*.

En ocasiones se escribe

$$\mathbb{P}(y|x) = \frac{1}{Z_\lambda(x)}\prod_{i=1}^{n}\exp\left(\sum_j \lambda_j f_j(y_{i-1}, y_i, x, i) + \sum_k \mu_k g_k(y_i, x, i)\right)$$

donde cada $f_j(y_{i-1}, y_i, x, i)$ contiene alguna información sobre las etiquetas en las posiciones i e $i-1$ y sobre la secuencia de observaciones completa y cada $g_k(y_i, x, i)$ es una función de la etiqueta en la posición i y la secuencia de observaciones x.

4.2.2. Representación gráfica

Un modelo probabilístico puede representarse de forma gráfica lo que permite visualizar cómo la distribución conjunta de las variables aleatorias (representadas como los nodos o vértices del grafo) puede descomponerse en un producto de factores donde cada factor depende sólo de un subconjunto de las variables.

En un HMM la probabilidad conjunta $\mathbb{P}(x, y)$ puede ser escrita como producto de factores $\Psi_c(v_c)$ donde v_c es el subconjunto de variables aleatorias que componen cada factor:

$$\mathbb{P}(x, y) = \prod_{i=1}^{n} \mathbb{P}(y_i|y_{i-1})\, \mathbb{P}(x_i|y_i) = \prod_{c} \Psi_c(v_c)$$

Las funciones Ψ_c, llamadas *funciones potenciales*, no necesariamente deben ser probabilidades como en un HMM, sino que una distribución de probabilidad puede escribirse como

$$\mathbb{P}(v) = \frac{1}{Z} \prod_{c} \Psi_c(v_c)$$

donde los factores $\Psi_c(v_c) \geq 0$ son funciones de las variables aleatorias v_c que se corresponden con el subconjunto c de los nodos o vértices del grafo que verifican que todo par de vértices en c están conectados por una arista (*clique*).

En un modelo probabilístico generativo como los HMM, el grafo subyacente es dirigido y las funciones Ψ_c son probabilidades, mientras que en un modelo gráfico no dirigido las funciones potenciales son sólo funciones no negativas, lo que hace necesario introducir un factor de normalización $Z(v) = \sum_v \prod_c \Psi_c(v_c)$ para tener una medida de probabilidad.

Un LCCRF se escribe como

$$\mathbb{P}(y|x) = \frac{1}{Z(x)} \prod_{i=1}^{n} \Psi_i(x, y)$$

donde la constante de normalización es $Z(x) = \sum_{y'} \prod_{i=1}^{n} \Psi_i(x, y')$ y los factores Ψ_i son de la forma

$$\Psi_i(x, y) = \exp\left(\sum_{j=1}^{m} \lambda_j f_j(y_{i-1}, y_i, x, i) \right)$$

y se representa mediante un grafo no dirigido (Figura 4.1).

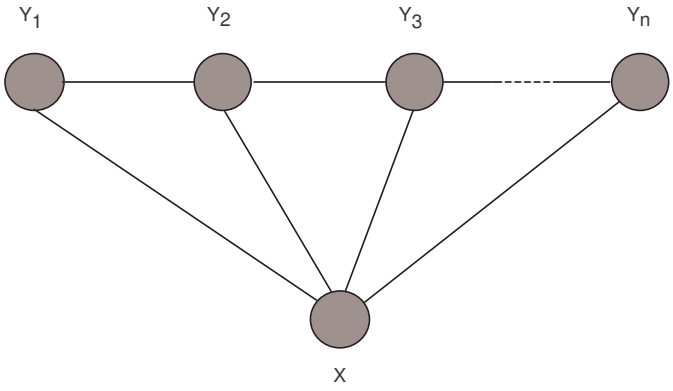

Figura 4.1: Estructura gráfica de un LCCRF.

4.2.3. Entrenamiento del modelo

Entrenar el modelo implica estimar los parámetros λ_j en

$$\mathbb{P}(y|x) = \frac{1}{Z_\lambda(x)} \exp\left(\sum_{i=1}^{n}\sum_{j=1}^{m} \lambda_j f_j(y_{i-1}, y_i, x, i)\right)$$

para $j = 1, \ldots, m$. Para ello se buscan los parámetros de forma tal que se maximice la log-verosimilitud $\mathcal{L}(\lambda)$ de los datos del conjunto de entrenamiento $\mathcal{T}$. Supongamos dado $\mathcal{T}$ con T secuencias de entrenamiento (x^t, y^t) donde $x^t = (x_1^t, x_2^t, \ldots, x_n^t)$, $y^t = (y_1^t, y_2^t, \ldots, y_n^t)$ son secuencias de observaciones y de etiquetas. Se buscan los $\lambda_j,\ j = 1, \ldots, m$ que hacen máxima

$$\mathcal{L}(\lambda) = \sum_{(x,y)\in\mathcal{T}} \ln \mathbb{P}(y|x) = \sum_{(x,y)\in\mathcal{T}} \ln\left[\frac{\exp\left(\sum_{i=1}^{n}\sum_{j=1}^{m} \lambda_j f_j(y_{i-1}, y_i, x, i)\right)}{\sum_{y'} \exp\left(\sum_{i=1}^{n}\sum_{j=1}^{m} \lambda_j f_j(y'_{i-1}, y'_i, x, i)\right)}\right]$$

La suma del denominador es sobre todas las secuencias de estados y' posibles y no sólo en los estados de las secuencias de entrenamiento.
Para evitar el sobreajuste del modelo se introduce un factor de penalización: $-\sum_j \frac{\lambda_j^2}{2\sigma^2}$ [27].

Reescribiendo la log-verosimilitud como en Klinger y Tomanek (2007) [20] se obtiene

$$\mathcal{L}(\lambda) = \sum_{(x,y)\in\mathcal{T}} \ln \mathbb{P}(y|x) = \sum_{(x,y)\in\mathcal{T}} \ln \left[\frac{\exp\left(\sum_{i=1}^{n} \sum_{j=1}^{m} \lambda_j f_j(y_{i-1}, y_i, x, i) \right)}{\sum_{y'} \exp\left(\sum_{i=1}^{n} \sum_{j=1}^{m} \lambda_j f_j(y'_{i-1}, y'_i, x, i) \right)} \right] - \sum_j \frac{\lambda_j^2}{2\sigma^2}$$

$$= \sum_{(x,y)\in\mathcal{T}} \sum_{i=1}^{n} \sum_{j=1}^{m} \lambda_j f_j(y_{i-1}, y_i, x, i)$$

$$- \sum_{(x,y)\in\mathcal{T}} \ln \left(\sum_{y'} \exp\left(\sum_{i=1}^{n} \sum_{j=1}^{m} \lambda_j f_j(y'_{i-1}, y'_i, x, i) \right) \right) - \sum_j \frac{\lambda_j^2}{2\sigma^2}$$

Las derivadas respecto de λ_k cada sumando son

$$\frac{\partial}{\partial \lambda_k} \sum_{(x,y)\in\mathcal{T}} \ln \left(\sum_{y'} \exp\left(\sum_{i=1}^{n} \sum_{j=1}^{m} \lambda_j f_j(y'_{i-1}, y'_i, x, i) \right) \right) = \frac{\partial}{\partial \lambda_k} \sum_{(x,y)\in\mathcal{T}} \ln Z_\lambda(x)$$

$$= \sum_{(x,y)\in\mathcal{T}} \frac{1}{Z_\lambda(x)} \frac{\partial Z_\lambda(x)}{\partial \lambda_k}$$

$$= \sum_{(x,y)\in\mathcal{T}} \frac{1}{Z_\lambda(x)} \sum_{y'} \exp\left(\sum_{i=1}^{n} \sum_{j=1}^{m} \lambda_j f_j(y'_{i-1}, y'_i, x, i) \right) \sum_{i=1}^{n} f_k(y'_{i-1}, y'_i, x, i)$$

$$= \sum_{(x,y)\in\mathcal{T}} \sum_{y'} \frac{1}{Z_\lambda(x)} \exp\left(\sum_{i=1}^{n} \sum_{j=1}^{m} \lambda_j f_j(y'_{i-1}, y'_i, x, i) \right) \sum_{i=1}^{n} f_k(y'_{i-1}, y'_i, x, i)$$

$$= \sum_{(x,y)\in\mathcal{T}} \sum_{y'} P(y'|x) \sum_{i=1}^{n} f_k(y'_{i-1}, y'_i, x, i)$$

$$\frac{\partial}{\partial \lambda_k} \sum_{(x,y)\in\mathcal{T}} \sum_{i=1}^{n} \sum_{j=1}^{m} \lambda_j f_j(y_{i-1}, y_i, x, i) = \sum_{(x,y)\in\mathcal{T}} \sum_{i=1}^{n} f_k(y_{i-1}, y_i, x, i)$$

$$\frac{\partial}{\partial \lambda_k} \left(- \sum_j \frac{\lambda_j^2}{2\sigma^2} \right) = -\frac{2\lambda_k}{2\sigma^2} = -\frac{\lambda_k}{\sigma^2}$$

por lo que

$$\frac{\partial \mathcal{L}(\lambda)}{\partial \lambda_k} = \sum_{(x,y)\in\mathcal{T}} \sum_{i=1}^{n} f_k(y_{i-1}, y_i, x, i) - \sum_{(x,y)\in\mathcal{T}} \sum_{y'} \mathbb{P}(y'|x) \sum_{i=1}^{n} f_k(y'_{i-1}, y'_i, x, i) - \frac{\lambda_k}{\sigma^2}$$

La función $\mathcal{L}(\lambda)$ es cóncava ya que es una suma de funciones cóncavas: el primer sumando es lineal, el segundo sumando $- \sum_{(x,y)\in\mathcal{T}} \ln \sum_{y'} \exp\left(\sum_{i=1}^{n} \sum_{j=1}^{m} \lambda_j f_j(y'_{i-1}, y'_i, x, i) \right)$ es cóncavo ya que la función $\ln \sum_i \exp(x_i)$ es convexa y el tercero también lo es dado que la derivada segunda respecto de λ_k

resulta negativa. Así basta con anular las derivadas parciales de la log verosimilitud para encontrar su máximo global.

La derivada parcial del primer sumando de $\mathcal{L}(\lambda)$ es el valor esperado $\tilde{E}(f_k)$ de la función característica f_k respecto de la distribución empírica $\tilde{\mathbb{P}}(x,y) = \frac{1}{T}\sum_{t=1}^{T} \mathbf{1}_{\{y=y^t\}}\mathbf{1}_{\{x=x^t\}}$.

La derivada del segundo término es el valor esperado de f_k bajo la distribución del modelo, $E(f_k)$, por lo que anular la derivada parcial de $\mathcal{L}(\lambda)$ respecto de λ_k equivale a anular $\tilde{E}(f_k) - E(f_k) - \frac{\lambda_k}{\sigma^2}$.

Para calcular $\tilde{E}(f_k)$ basta con contar cuántas veces ocurre cada característica en los datos de entrenamiento, pero calcular $E(f_k)$ sólo es posible en forma numérica utilizando una variante del algoritmo *forward-backward* descrito para HMM (McDonald y Pereira [24]).

4.2.4. Inferencia

Una vez estimados los parámetros del modelo, el objetivo de la inferencia consiste en encontrar la secuencia de etiquetas y más probable para la secuencia de observaciones dada, esto es $y^* = arg\,\max_y \mathbb{P}(y|x)$. Para ello se utiliza una variante del algoritmo de Viterbi.

4.3. Campos aleatorios condicionados generales

Lafferty et al [22] introducen los CRF en su forma general, esto es, no necesariamente con una estructura de cadena lineal en la que cada función característica sólo involucra dos estados consecutivos.

La generalización de un LCCRF a un CRF implica no sólo un cambio en la estructura del grafo sino la utilización de algoritmos más generales para el entrenamiento y la inferencia del modelo.

La definición 4.1 es la de un campo aleatorio condicionado general (Lafferty et al [22]).

Definición 4.1. *Si X e Y son variables aleatorias que representan secuencias de observaciones y de etiquetas respectivamente y $G = (V, E)$ es un grafo con vértices V y aristas E tal que $Y = (Y_v)_{v \in V}$, entonces (X, Y) es un CRF si, al condicionar en X, las variables aleatorias Y_v obedecen la propiedad de Markov con respecto al grafo:*

$$P(Y_v|X, Y_w, w \neq v) = P(Y_v|X, Y_w, w \sim v)$$

donde $w \sim v$ significa que w y v son vecinos en el grafo G.

4.4. Aplicación de CRF en Bioinformática

La gran diferencia entre los predictores de genes basados en HMM y aquellos que se basan en CRF, es que los primeros utilizan como dato solamente la secuencia a anotar (secuencia objetivo) y los últimos incorporan además información de otro tipo mediante las funciones características. Ésto trae como consecuencia una mejora en la performance de los modelos CRF respecto de los HMM, pero tiene como desventaja el incremento de la complejidad del modelo.
Los CRF si bien fueron diseñados para análisis de textos [22], han sido adaptados a los problemas de búsqueda de genes, existiendo en la actualidad varios modelos creados para la identificación de regiones codificantes en genomas particulares. En éstos la forma en la cual se introduce en el modelo la información está especialmente adaptada al genoma bajo estudio, razón por la cual resulta imprescindible adaptar el programa para cada caso que se desee analizar. Debe tenerse en cuenta además que el costo computacional en este tipo de programas es muy elevado.

Flicek [15] resume los predictores de genes que han sido mayormente utilizados en los últimos años. Entre los programas basados en GHMM, GENSCAN presentado en 1997, es el más conocido y el que demostró mayor precisión entre los predictores de genes hasta ese momento, aún prediciendo de manera correcta sólo un pequeño porcentaje de los genes en conjuntos de datos reales. En 2002 se publica la secuencia de genes del ratón que puede ser utilizada en genómica comparativa para descifrar por ejemplo, el genoma humano, lo que trae como consecuencia el desarrollo de varios programas con el objetivo de incorporar esta nueva fuente de datos. El programa más exitoso en este campo fue TWINSCAN y más adelante su extensión N-SCAN.
En 2007 se presentan varios programas predictores de genes que utilizan CRF. Algunos basados en semi-Markov CRF como CRAIG desarrollado por Bernal et al [4] y CONRAD creado por DeCaprio

et al [8]. Otros se basan en CRF combinado con otros clasificadores y utilizando secuencias genómicas informantes como CONTRAST [16].

4.4.1. LCCRF para predicción de genes

Dado que en el problema de la búsqueda de genes la salida del modelo consiste en secuencias de etiquetas, por ejemplo (`exon`, `exon`, `exon`, `intron` , `intron`), corresponde entonces, la utilización de CRF para secuencias, esto es LCCRF.
Un LCCRF modela la probabilidad condicional de la secuencia oculta de estados y dada la secuencia observada de nucleótidos x como

$$P(y|x) = \frac{1}{Z_\lambda(x)} \prod_{i=1}^{n} \exp\left(\sum_{j=1}^{m} \lambda_j f_j(y_{i-1}, y_i, x, i) \right)$$

donde i indica la posición en la secuencia de observaciones x, λ_j son parámetros reales, f_j funciones características y Z_λ es una constante de normalización. La parte fundamental en el diseño de un CRF es la elección de las funciones características f_j.
Las aplicaciones $f(y_{i-1}, y_i, x, i)$ y $g(y_i, x, i)$ son ejemplos de funciones características.

$$f(y_{i-1}, y_i, x, i) = \begin{cases} 1 & \text{si } y_{i-1} = \texttt{intron}, y_i = \texttt{exon} \\ 0 & \text{en otro caso} \end{cases}$$

$$g(y_i, x, i) = \begin{cases} 1 & \text{si } y_{i-1} = \texttt{intron}, x_i = \texttt{'G'} \\ 0 & \text{en otro caso} \end{cases}$$

4.4.2. SMCRF para predicción de genes

En predicción de genes, los estados ocultos y tienen una estructura lineal, esto es, la salida del modelo es un vector $y_1, y_2, \ldots, y_n$ donde cada y_i, que puede ser por ejemplo `exon` o `intron`, se corresponde con cada nucleótido de la secuencia x a anotar.
O también, la secuencia y puede estar formada por etiquetas que se correspondan con la división de la secuencia de nucleótidos en un número variable r de intervalos (t_i, u_i, v_i) para $i = 1, 2, \ldots, r$, donde t_i indica el inicio del intervalo, u_i el final y v_i la etiqueta asignada a éste:

$$t_1 = 1;\ u_i \geq t_i;\ u_r = n;\ v_{i-1} \neq v_i$$

$$y_{t_i} = y_{t_{i+1}} = y_{t_{i+2}} = \ldots = y_{u_i} = v_i$$

Por ejemplo, la secuencia $(x_1, x_2, x_3, x_4, x_5)$ puede etiquetarse como (`exon`, `exon`, `exon`, `intron`, `intron`) o también segmentarse en $r = 2$ intervalos, uno con el rótulo `exon` y otro con `intron`, esto es $(t_1, u_1, v_1) = (1, 3, \texttt{exon})$ y $(t_2, u_2, v_2) = (4, 5, \texttt{intron})$.

Esta forma de asignar las etiquetas permite modelar con mayor comodidad algunas de las restricciones que se presentan en las transiciones de una etiqueta a otra, por ejemplo, la transición de una región intergénica (intergénica) a un exon (exon) sólo puede darse cuando el codon de inicio es ATG, esto es si $v_{i-1} =$ intergénica entonces $(t_i, u_i, v_i) = (\texttt{A}, \texttt{G}, \texttt{exon})$. Al igual que en los HMM o GHMM se impone la condición de que cada intervalo (t_i, u_i, v_i) interactúe solamente con sus vecinos, luego las funciones características son de la forma $f_j(v_{i-1}, t_i, u_i, v_i, x, i)$ y el modelo es

$$P(y|x) = \frac{1}{Z_\lambda(x)} \prod_{i=1}^{n} \exp\left(\sum_{j=1}^{m} \lambda_j f_j(v_{i-1}, t_i, u_i, v_i, x, i) \right)$$

donde i indica la posición en la secuencia de observaciones x, λ_j son parámetros reales, f_j funciones características y Z_λ es una constante de normalización

$$Z_\lambda(x) = \sum_{y} \exp\left(\sum_{i=1}^{m} \sum_{j=1}^{n} \lambda_j f_j(v_{i-1}, t_i, u_i, v_i, x, i) \right)$$

Entonces, para anotar una secuencia de observaciones x dada, el modelo encuentra la secuencia segmentada de etiquetas y que hace máxima la probabilidad $P(y|x)$. Este tipo de modelo CRF recibe el nombre de *Semi-Markov Conditional Random Fields* (SMCRF).

Los parámetros λ_j en un CRF se entrenan maximizando la función de log-verosimilitud de y dado x utilizando para ello el conjunto de secuencias de entrenamiento. Esta función, que resulta cóncava tiene máximo global que se calcula usando el método del gradiente descendente. Para un SMCRF las derivadas parciales de la función log-verosimilitud se calculan usando un algoritmo de programación dinámica del tipo *forward-backward*. Para la inferencia se utiliza una variante del algoritmo de Viterbi.

4.4.3. CONRAD y CONTRAST

Los CRF pueden incorporar información al modelo en forma de funciones características. Esta información puede provenir por ejemplo de secuencias alineadas de otros genomas o de EST [1].

Entre los modelos que utilizan CRF para la búsqueda de genes disponibles, se describen brevemente a continuación CONRAD y CONTRAST.

[1] Las EST son subsecuencias cortas de una secuencia de nucleótidos ya transcrita.

CONRAD, CONditional RAnDom Fields

CONRAD, presentado por DeCaprio et al [8] es un predictor de genes basado en *Semi-Markov Linear Chain Conditional Random Fields* (SMLCCRF). CONRAD incorpora información adicional como las EST, que codifica en forma de funciones características.

En procesamiento de textos se definen con frecuencia muchas funciones características binarias, que valen 0 la mayor parte del tiempo y 1 sólo en algunas situaciones especiales. En predicción de genes se toma como punto de partida en la construcción de las funciones características los modelos probabilísticos ya existentes. En el enfoque propuesto por DeCaprio et al [8] se utilizan modelos probabilísiticos para las funciones características siempre que esto sea posible y modelos no probabilísticos sólo cuando es necesario.

El modelo se aplica para predecir genes en *Cryptococcus neoformans* y *Aspergillus nidulans*. Los resultados son comparados con los obtenidos en programas como TWINSCAN para *Cryptococcus neoformans* y FGENESH para el caso de *Aspergillus nidulans* [8].

A la hora de construir un modelo CRF además de decidir sobre el número de estados a incluir, debe tenerse especial cuidado en el diseño de las funciones caracterísiticas f_j y en la selección de los pesos λ_j. Las funciones características captan las propiedades relevantes de las observaciones x y las utilizan para asignar un valor a cada etiqueta de cada intervalo posible, no requiriendo independencia ni interpretación probabilística.

CONRAD cuenta 13 estados que modelan las regiones codificantes en los genes y 29 funciones características.

Para calcular los pesos, esto es para entrenar el modelo, se utilizan algoritmos como *Conditional Maximum-Likelihood* (CML) y *Maximum Expected Accuracy* (MEA) [8].

Si bien CONRAD muestra mejoras respecto de los principales programas de predicción en los genomas de hongos, su aplicación a los genomas de mamíferos es computacionalmente prohibitiva [15].

CONTRAST (CONditionally TRAined Search for Transcripts)

CONTRAST, desarrollado por Gross et al [16] incorpora datos de genomas informantes combinando además CRF con una técnica de aprendizaje automático llamada *Support Vector Machine* (SVM). El programa utiliza clasificadores SVM para el reconocimiento de las fronteras de las regiones codificantes (sitios de empalme, codones de inicio y finalización) y CRF para modelar la estructura del gen.

Integra los resultados obtenidos por SVM con datos provenientes de secuencias alineadas de otros genomas llamados informantes, efectuando un alineamiento múltiple basado en algunos supuestos sobre los respectivos procesos evolutivos. La similitud entre los genomas de dos especies está correlacionada con su distancia evolutiva, esto es, con los cambios genéticos producidos a partir de la ramificación de las especies de un ancestro común. Entonces si se consideran dos especies con una distancia evolutiva apropiada, el alineamiento de sus secuencias puede usarse para identificar regiones del genoma que se hayan conservado, en particular regiones que codifican en proteínas.

El programa fue testeado con el genoma humano encontrando correctamente casi el 60 % de los genes, utilizando para ello 11 especies informantes. Para evaluar la precisión se compararon los resultados obtenidos con CONTRAST con las predicciones calculadas con otro software, N-SCAN, que predice correctamente casi el 35 % de los genes humanos.

Algunos de los resultados obtenidos en el trabajo de Gross et al, dados en porcentaje de sensibilidad y especificidad utilizando N-SCAN y CONTRAST con el genoma de ratón como único informante y CONTRAST con 11 informantes, se muestran en el Cuadro 4.1 (Gross et al [16]).

	N-SCAN (ratón)	CONTRAST (ratón)	CONTRAST (11 informantes)
Sensibilidad	35.6	50.8	58.6
Especificidad	25.1	29.3	35.5
Genes predichos	22596	27614	26260

Cuadro 4.1: Porcentajes de sensibilidad y especificidad en genes del genoma humano.

Capítulo 5

Predicción de genes VSG en *Trypanosoma brucei*

5.1. *Trypanosoma brucei*

La tripanosomiasis africana o enfermedad del sueño, es causada por un parásito: el *Trypanosoma brucei*[1].

Las infecciones se limitan al África subsahariana, donde los insectos que resultan ser vectores transmisores de la enfermedad (moscas tse-tse) son endémicos. Se estima que en la actualidad entre 50000 y 70000 personas están infectadas de tripanosomiasis africana según datos de la Organización Mundial de la Salud [33].

La enfermedad es transmitida mediante la picadura de una mosca tse-tse infectada (Figura 5.1)[2] y en el caso de la tripanosomiasis africana humana se presenta en dos formas dependiendo del parásito involucrado: *Trypanosoma brucei gambiense*, que representa el 90 % de los casos de la enfermedad del sueño y causa una infección crónica en la cual una persona puede estar infectada durante largo tiempo sin presentar síntomas (éstos se manifiestan cuando la persona se encuentra en una etapa avanzada de la enfermedad); y *Trypanosoma brucei rhodesiens* que ocasiona una infección aguda afectando el sistema nervioso central en cuestión de semanas.

La enfermedad también se manifiesta en otros animales particularmente en el ganado (en este caso

[1]Los tripanosomas son protozoos que integran el Reino Protista. Éste está formado por organismos con células eucariotas que no se clasifican como hongos, animales o plantas, como es el caso de las algas y los protozoos.

[2]Centers for Disease Control and Prevention (CDC), disponible en
http://www.dpd.cdc.gov/dpdx/HTML/TrypanosomiasisAfrican.htm

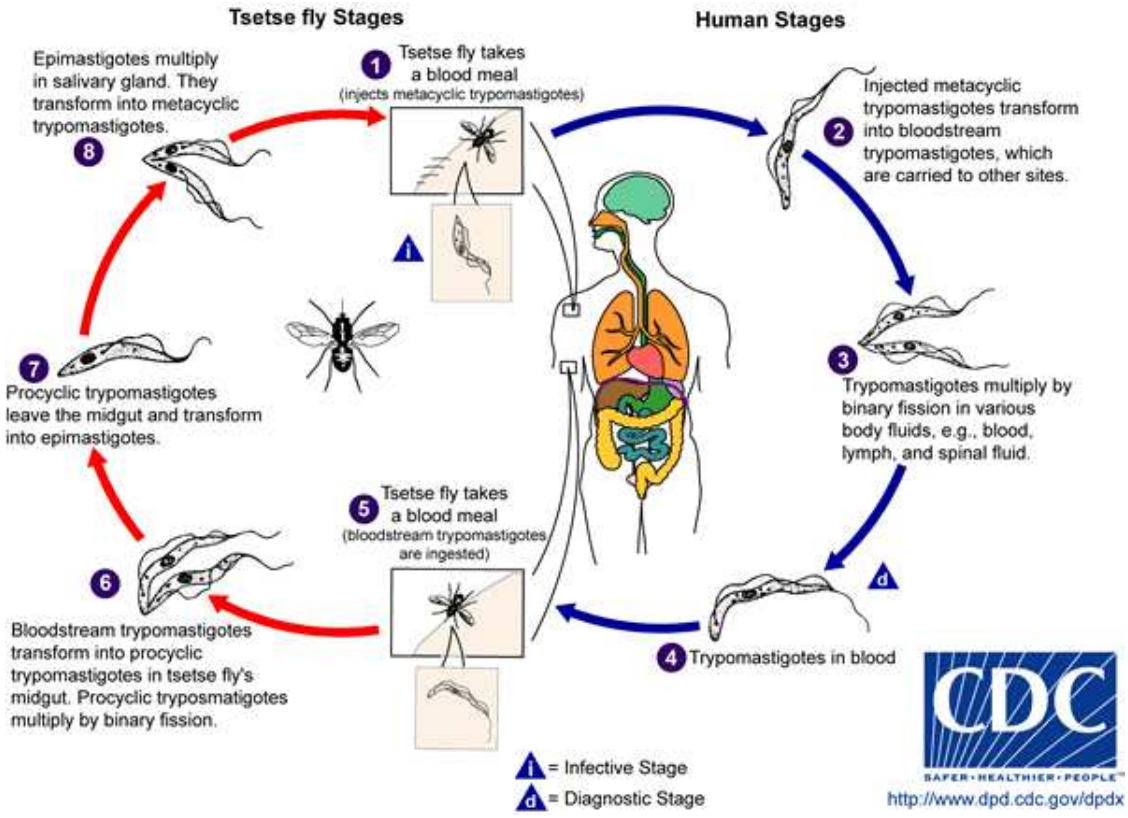

Figura 5.1: Ciclo de la tripanosomiasis africana.

se le llama Nagana[3]), incidiendo en el desarrollo económico de las áreas rurales afectadas.

En América del Sur y América Central se da otro tipo de tripanosomiasis que se denomina tripanosomiasis americana o enfermedad de Chagas, siendo en este caso el *Trypanosoma cruzi* el agente infeccioso.

La superficie de la célula de los tripanosomas se compone de una capa uniforme de glicoproteína variante de superficie (VSG) que funciona como una barrera que bloquea el reconocimiento del tripanosoma por el sistema inmunitario del mamífero huésped.

Una vez que el individuo resulta infectado, el tripanosoma expresa una VSG en particular y el sistema inmune del huésped reacciona generando una respuesta a esta capa de proteínas lo que produce una disminución en el número de tripanosomas. Pero algunos tripanosomas eluden este reconocimiento expresando una VSG alternativa para la cual aún no se han desarrollado anticuerpos, aumentando nuevamente la población de agentes infecciosos. Así se suceden distintas infecciones cada una de las cuales es ocasionada por células con diferentes expresiones de cubiertas VSG.

[3]N'gana una palabra zulú que significa estar deprimido.

Los genes que codifican VSG (que llamaremos genes VSG) pueden buscarse dentro del genoma del *Trypanosoma brucei* utilizando técnicas basadas en homología de secuencias, esto es comparando la secuencia de ADN dada con secuencias almacenadas en bases de datos. Pero, como las secuencias codificantes de VSG presentan una gran variabilidad, la búsqueda de genes VSG basada en la homología puede no arrojar buenos resultados.
Cabe preguntarse entonces si estos genes VSG presentan características particulares que determinen la posibilidad de localizarlos mediante otros métodos, por ejemplo utilizando modelos matemáticos. Los modelos de Markov ocultos (HMM) son modelos probabilísticos diseñados para el análisis de secuencias de datos, por lo que pueden utilizarse para determinar si es posible distinguir en el genoma del *Trypanosoma brucei* zonas homogéneas (esto es, regiones correspondientes a un mismo estado) que se identifiquen con genes VSG.

5.2. HMM para predicción de genes VSG

5.2.1. Datos y elección del modelo

Se cuenta con nueve secuencias del cromosoma 9 del genoma del *Trypanosoma Brucei* donde se localizan clusters de genes VSG (Cuadro 5.1) y se elige para modelar el problema de predicción de genes VSG un modelo de Markov oculto (HMM). Para definir el modelo se debe establecer el número de estados y el de símbolos, es decir debe elegirse el número de parámetros. Como los datos en este caso son secuencias de ADN, el alfabeto de símbolos cuenta con cuatro letras que se corresponden con las cuatro bases: adenina (A), guanina (G), citosina (C) y timina (T). En cuanto al número de estados se comienza ajustando un HMM con dos estados, que representarían las zonas VSG y no VSG.

Para estimar las matrices de probabilidades de transición de estados (estT) y emisión de símbolos (estE) del modelo de Markov oculto, se divide en forma aleatoria el conjunto de las nueve secuencias dadas en dos grupos.
El primero de estos grupos (conjunto de entrenamiento) se utiliza para estimar los parámetros del modelo por medio del algoritmo de Baum - Welch. Se ingresan como datos sólo las secuencias de nucleótidos, esto es no se tiene en cuenta el conocimiento existente sobre la anotación de los genes VSG. El algoritmo de Baum - Welch finaliza cuando el cambio, de un paso de iteración a otro, en la función de log-verosimilitud (de que la secuencia dada sea generada por los valores estimados de las

secuencias	longitud (pb)	número de genes VSG anotados
secuencia 1	75462	16
secuencia 2	67530	21
secuencia 3	19745	7
secuencia 4	89317	10
secuencia 5	102962	3
secuencia 6	17110	5
secuencia 7	88923	8
secuencia 8	24879	0
secuencia 9	65933	7

Cuadro 5.1: Longitud medida en pares de bases (pb) y número de genes VSG identificados en 9 secuencias del cromosoma 9 del genoma de *Trypanosoma brucei* que contienen clusters de genes VSG.

probabilidades de transición y emisión en ese paso) y la norma[4] de la matrices de transición y emisión, sean todas cantidades menores que un valor determinado denominado tolerancia. En todos los análisis se consideró una tolerancia de 0.00001 y se colocó además como condición de parada adicional que el número máximo de iteraciones a ejecutar sea 500.
El otro grupo de secuencias (conjunto de testeo) es utilizado para validar el modelo.

Luego, mediante el algoritmo de Viterbi se estima, para cada secuencia de testeo, el camino de estados más probable según la estimación de parámetros obtenida por Baum - Welch, determinando cuáles regiones de las secuencias de testeo son predichas como genes VSG. Estos resultados se comparan con los genes VSG anotados para estas secuencias.

Finalmente se calcula la precisión de las predicciones obtenidas con el modelo elegido. Al comparar las predicciones del modelo con los genes VSG observados pueden darse varios tipos de error.
Por ejemplo el modelo puede predecir un VSG que no figura como tal en la secuencia anotada o puede ser que un gen VSG anotado no sea predicho por el modelo (Figura 5.2).
También puede ocurrir que aquellos genes VSG encontrados correctamente por el modelo no coincidan exactamente en su ubicación dentro de la secuencia con los genes VSG anotados (Figura 5.3).

[4]La norma de una matriz es una medida de la magnitud de los elementos de la matriz. Si bien pueden definirse varias normas de matrices, el algoritmo utilizado se basa en la norma definida como el mayor valor singular de la matriz.

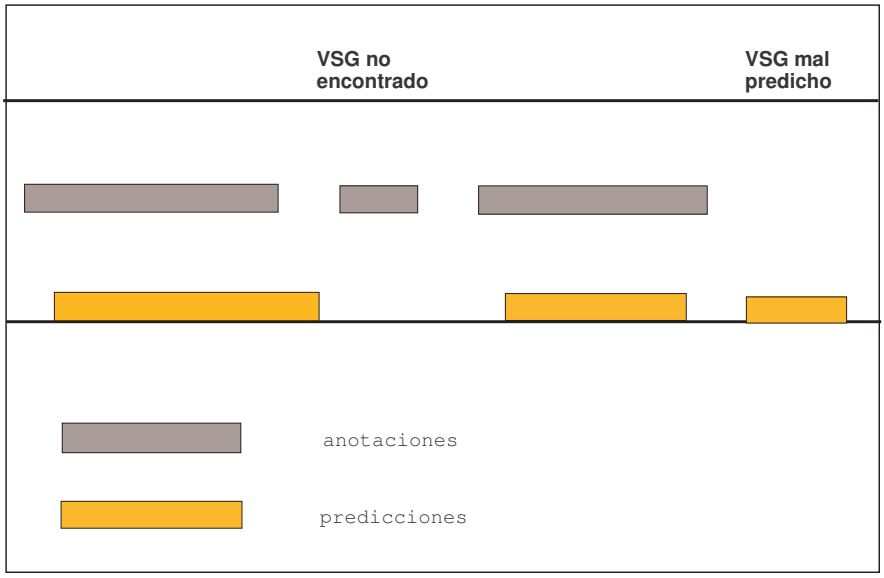

Figura 5.2: Precisión de las predicciones de VSG.

Los aciertos en predecir genes VSG se cuentan como verdaderos positivos (VP). Las diferencias cuando el modelo predice que determinados nucleótidos forman parte de un gen VSG cuando esto no es cierto se contabilizan como falsos positivos (FP) y, como falsos negativos (FN), se cuentan aquellos nucleótidos que el modelo no predice como parte de VSG y sí forman parte.

predicciones ⁄ anotaciones	positivo	negativo
positivo	verdadero positivo (VP)	falso positivo (FP)
negativo	falso negativo (FN)	verdadero negativo (VN)

Cuadro 5.2: Medidas de precisión.

Es frecuente analizar la precisión del modelo calculando la sensibilidad y especificidad tomando en cuenta el número de nucleótidos predichos en acierto y en error.

La sensibilidad se define como el cociente entre el número de nucleótidos predichos correctamente sobre el total de nucleótidos que forman parte de los VSG, es decir da la proporción de nucleótidos predichos como parte de genes VSG entre todos los posibles nucleótidos de genes VSG.

La especificidad se calcula como el cociente entre las observaciones predichas correctamente como VSG sobre el total de observaciones predichas como VSG, esto es da la proporción de nucleótidos predichos como parte de genes VSG entre todos los genes VSG predichos.

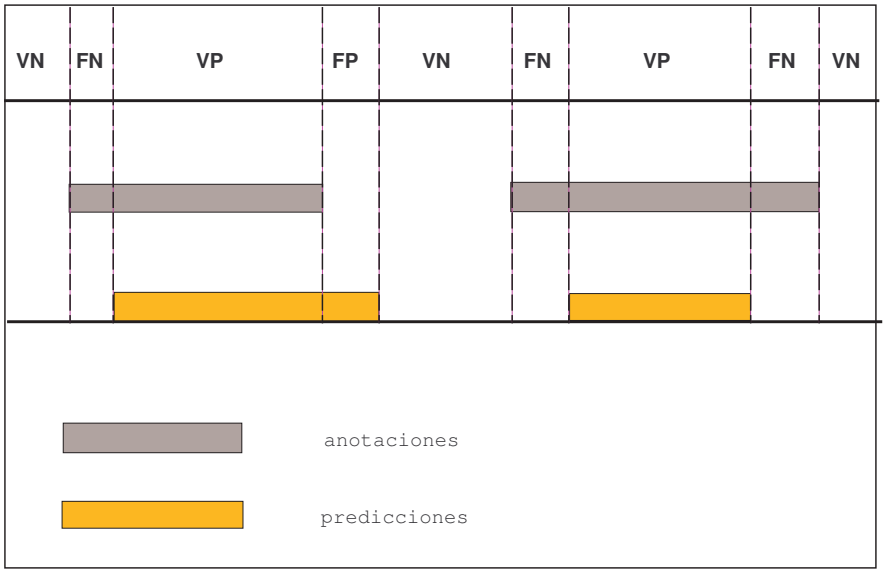

Figura 5.3: Precisión de las predicciones en nucleótidos.

$$\text{sensibilidad}=\frac{VP}{VP+FN}$$

$$\text{especificidad}=\frac{VP}{VP+FP}$$

5.2.2. Modelo con dos estados

Se escoge en primera instancia como modelo para la búsqueda de los genes VSG un HMM con dos estados.

Para estimar los parámetros del modelo se divide el conjunto de secuencias en dos grupos: seis secuencias (1, 2, 4, 5, 7, 8 que forman el grupo de llamaremos A) para el entrenamiento y las tres restantes (3, 6, 9) para la validación. Se considera además otra partición de los datos con las secuencias 1, 2, 4, 3, 6, 9 (grupo B) para la estimación de parámetros. El número de iteraciones necesarias para la convergencia del algoritmo de Baum-Welch en los dos casos no alcanza las 160.

Para evaluar el modelo se calcula el camino de estados más probable para las secuencias que no participaron en el entrenamiento del HMM y se comparan las sucesiones de estados predichas con los correspondientes de la secuencias anotadas.

En ambas pruebas se observa que el modelo predice todos los genes VSG, pero se cometen errores tanto en la localización exacta de los genes VSG predichos como en la predicción de genes VSG que no lo son, razón por la cual la sensibilidad es alta (superior al 94 % en todos los casos) y la especificidad baja (inferior al 50 % en la mayoría de los casos) (Cuadro 5.3).

	grupo A			grupo B		
	secuencia 3	secuencia 6	secuencia 9	secuencia 5	secuencia 7	secuencia 8
Sensitividad	0.9709	0.9634	0.9838	0.9724	0.9485	-
Especificidad	0.5214	0.4652	0.2781	0.2457	0.4775	-
VSG anotados	7	5	7	3	8	0
VSG predichos	10	5	23	18	15	1
VSG no encontrados	0	0	0	0	0	0

Cuadro 5.3: Sensibilidad, especificidad y VSG encontrados para el modelo con dos estados.

5.2.3. Modelo con tres estados

Se realizaron pruebas con un HMM considerando tres estados para investigar la posibilidad de la existencia de un tercer grupo de regiones reconocibles por el modelo. Se consideran los mismos conjuntos de entrenamiento que los utilizados para el modelo con dos estados a modo de comparación. Es decir, los conjuntos de entrenamiento los conforman las secuencias 1, 2, 4, 5, 7, 8 (grupo A) y 1, 2, 4, 3, 6, 9 (grupo B). El algoritmo de Baum - Welch converge luego de 323 (Figura 5.4) y 262 iteraciones en el primer y segundo grupo respectivamente.

Las matrices de probabilidades de transición y emisión estimadas para el grupo A de secuencias indican que las probabilidades de permanencia en cada estado, ubicadas en la diagonal de la matriz estT, son altas. Además, dado que los símbolos 1, 2, 3 y 4 se corresponden con las bases A, C, G y T respectivamente, en la matriz estE se visualizan las probabilidades de que cada estado emita el símbolo correspondiente.

$$\text{estT}=\begin{pmatrix} 0{,}9956 & 0{,}0028 & 0{,}0016 \\ 0{,}0021 & 0{,}9976 & 0{,}0003 \\ 0{,}0017 & 0{,}0003 & 0{,}9980 \end{pmatrix} \qquad \text{estE}=\begin{pmatrix} 0{,}3610 & 0{,}1455 & 0{,}1388 & 0{,}3547 \\ 0{,}1712 & 0{,}2346 & 0{,}2267 & 0{,}3675 \\ 0{,}3610 & 0{,}2330 & 0{,}2369 & 0{,}1691 \end{pmatrix}$$

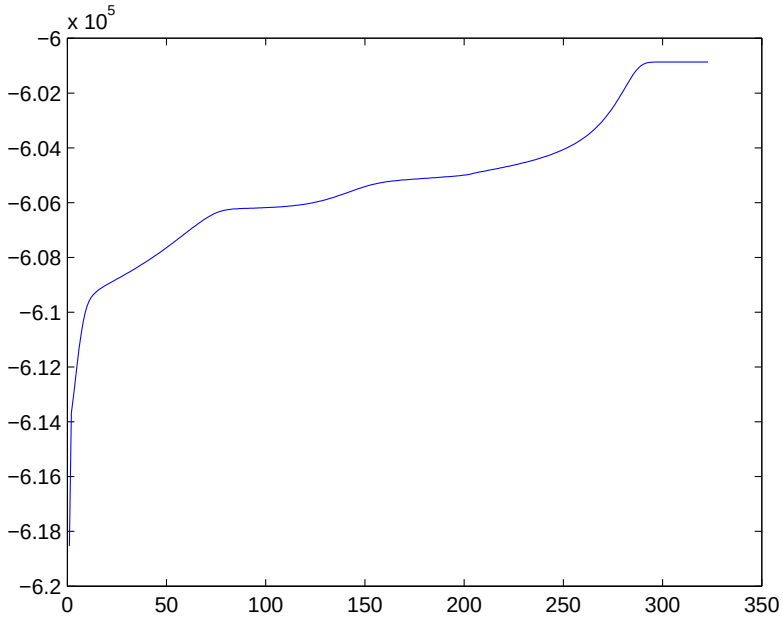

Figura 5.4: Log-verosimilitud del HMM con tres estados para el grupo A de secuencias de entrenamiento en función del número de iteraciones del algoritmo de Baum-Welch.

En primer lugar es claro que el modelo de Markov oculto con tres estados ajusta mejor que aquel con dos estados, ya que en todos los casos si bien la sensibilidad disminuye levemente (no se obtienen valores por debajo del 94 %), la especificidad aumenta de manera significativa (Cuadro 5.4).

	grupo A			grupo B		
	secuencia 3	secuencia 6	secuencia 9	secuencia 5	secuencia 7	secuencia 8
Sensitividad	0.9472	0.9540	0.9559	0.9535	0.9405	-
Especificidad	0.6712	0.7080	0.4022	0.3985	0.6629	-
VSG anotados	7	5	7	3	8	0
VSG predichos	14	11	13	13	12	2
VSG no encontrados	0	0	0	0	0	0

Cuadro 5.4: Sensibilidad, especificidad y VSG encontrados para el modelo con tres estados.

Consideremos, como ejemplo para analizar con detalle, la secuencia 9, para la cual el HMM con tres estados encuentra los siete genes VSG y predice equivocadamente seis genes VSG.

coordenadas genes VSG	coordenadas genes VSG predichos
1001 - 2502	658 - 2427
	3056 - 3169
4223 - 5692	3423 - 5640
6846 - 7375	6429 - 7544
8875 - 10387	8626 - 10315
	10829 - 11339
12360 - 13819	11555 - 13748
	14192 - 14845
16342 - 17943	16184 - 17849
	18148 - 18398
	18712 - 18918
20343 - 21634	19746 - 21585
	21676 - 27276
	50143 - 52575

Cuadro 5.5: Localización de los genes VSG anotados y predichos en la secuencia 9 con el modelo HMM con tres estados.

La longitud de los genes VSG predichos correctamente en la secuencia 9 varía entre un mínimo de 1115 pb a un máximo de 2217 pb, mientras que los genes VSG predichos equivocadamente por el modelo tienen longitudes de 113, 510, 653, 250, 206, 5600, 2432 pb respectivamente; es decir todas cantidades fuera del rango 1115 - 2217. Esta situación respecto de las longitudes de los VSG predichos en forma errónea, se da para un gran número de los genes mal predichos en las secuencias de validación, lo cual indica que si se considera la longitud de las predicciones correctas es posible ajustar las predicciones brindadas por el modelo descartando aquellas que estén fuera del intervalo determinado por las longitudes mínima y máxima de los genes VSG predichos en forma acertada.

Los genes VSG en la secuencia en 9 se localizan entre los primeros 22000 nucleótidos (Cuadro 5.5), región en la cual el camino de estados más probable determinado por el algoritmo de Viterbi muestra, que salvo entre las coordenadas 6277 - 6429 en la cual se estima el estado 2, se alternan transiciones sólo entre los estados 1 y 3 (este último es el que se corresponde con los genes VSG). A partir de

la ubicación 27560 sólo se alternan cambios entre los estados 1 y 2 (con solamente la excepción del estado 3 entre los lugares 50143 y 52575). Esto podría estar indicando la posibilidad de que el modelo esté detectando, además de los genes VSG, otro tipo de regiones (correspondientes al estado 2) con características particulares que las hacen distinguibles de aquellas predichas como estados 1 y 3.
Si se observan las coordenadas de los genes VSG localizados en la secuencia complementaria y se comparan éstas con las correspondientes a las predicciones del estado 2 (Cuadro 5.6) es posible concluir que este estado se corresponde entonces con los genes VSG de la secuencia complementaria.

Para visualizar Los genes VSG anotados y predichos para la secuencia 9 se utilizó el programa Artemis[5]. Las anotaciones se indican con la etiqueta `vsg` (color oscuro) y las predicciones se rotulan `pred` (color claro) (Figura 5.5).

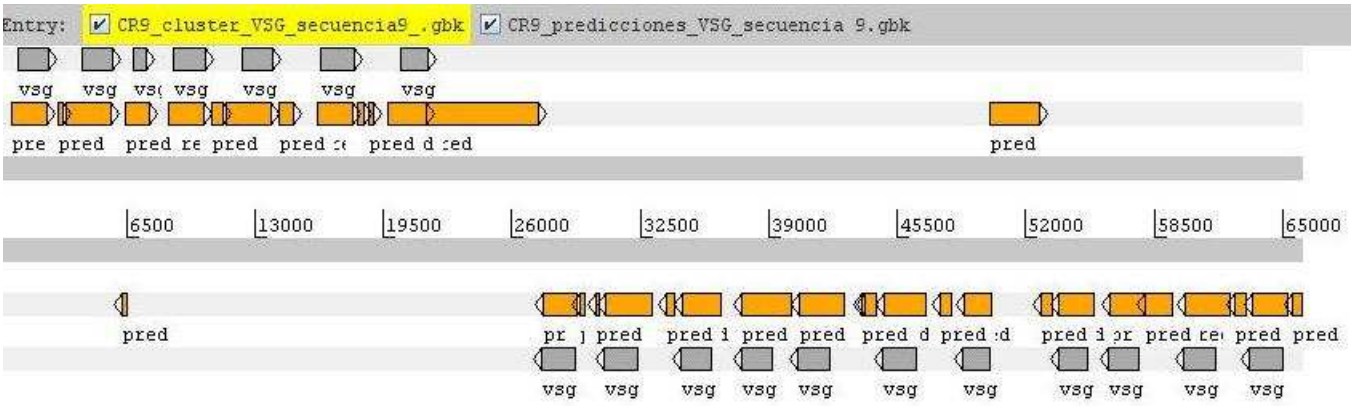

Figura 5.5: Genes VSG anotados y predichos por HMM con tres estados para la secuencia 9.

Resultados similares se observan en las restantes secuencias de validación, esto es, el HMM con tres estados identifica como tercera zona con características homogéneas en el genoma de *Trypanosoma brucei* aquellas que se corresponden con los genes VSG ubicados en las secuencias complementarias (Cuadro 5.7).
Finalmente se incorporan los resultados obtenidos al entrenar el modelo con el tercer grupo de secuencias formados por la 3, 6, 9, 5, 7, 8 (grupo C) (Cuadro 5.7).

[5]Artemis, software disponible en http://www.sanger.ac.uk/Software/Artemis

5.3. Independencia de las secuencias

Un aspecto de gran importancia en los datos de referencia, tanto aquellos que se utilizan para entrenar como para testear, es que éstos presenten independencia, es decir que sean realizaciones mutuamente independientes de un mismo fenómeno.

Un problema corriente al utilizar datos de tipo genómico para estudios de reconocimiento de patrones, es que muchos genes o segmentos genómicos no son independientes, sino que derivan de un ancestro común. Las secuencias de datos pueden estar altamente correlacionadas dado que cuando comenzó el proceso de divergencia, éstas eran idénticas. La correlación entre secuencias se le llama inercia filogenética e indica la similitud que presentan dos secuencias debido a la ancestría común. Debe tenerse en cuenta sin embargo, que el proceso de divergencia evolutiva borra progresivamente la huella filogenética, de hecho en dos secuencias que hayan divergido lo suficiente, los vestigios de un antepasado común pueden haber desaparecido completamente.

Debido a que la ancestría resulta en una gran similitud no sólo en las propiedades estadísticas de los segmentos de ADN o proteínas en cuestión, sino también similitud en la secuencias (ordenamiento lineal de los componentes de la molécula), es posible determinar el grado de independencia de dos segmentos de ADN mediante la comparación del ordenamiento de sus nucleótidos (comparación de secuencias). Por ejemplo, dos secuencias aleatorias con la misma composición de bases (de 1/4 para cada nucleótido) tienen una similitud del 25 % (la similitud esperada por azar lógicamente depende de la frecuencia de los distintos nucleótidos). Por otra parte, dos secuencias que hayan surgido de un ancestro común (y por tanto idénticas a tiempo 0) tienen una similitud esperada luego de un tiempo infinito de divergencia también de un 25 %, es decir igual al que presentan dos secuencias aleatorias.

Esto último quiere decir que el porcentaje de identidad de secuencias puede ser usado como indicador de independencia estadística de las mismas. Así, si se quiere construir un conjunto de secuencias estadísticamente independiente, el procedimiento consiste en comparar todas las secuencias contra todas y retener a aquel conjunto que no incluya a ningún par de secuencias cuya identidad es mayor a lo esperado por azar. Este procedimiento en la jerga bioinformática se le llama producir un set no redundante.

A pesar de que los genes VSG constituyen una superfamilia génica, y por lo tanto son derivados de un ancestro común, los mismos presentan una identidad aminoacídica extremadamente baja. Esto implica que los genes VSG pueden considerarse un set básicamente no redundante y por ende independiente. (Álvarez, Fernando. 2010. Comunicación personal).

5.4. Conclusiones

Los resultados obtenidos indican que los genes VSG presentan características que los hacen reconocibles como zonas homogéneas, con características particulares, dentro del genoma de *Trypanosoma brucei*.
Los análisis realizados permiten concluir además que el modelo de Markov oculto utilizado resultó ser eficiente en la búsqueda de genes VSG, en todas las pruebas la sensibilidad alcanzada por el modelo supera el 90 %, prediciéndose todos los genes objetivo. Sin embargo, debido a que también se predicen como genes VSG regiones del genoma que no resultan serlo, la especificidad del HMM es baja. Ahora, dado que un gran número de los genes VSG mal predichos por el modelo tienen una longitud inferior o superior a las longitudes mínima y máxima de los VSG predichos correctamente, una opción para refinar los resultados podría ser eliminar aquellas predicciones cuya longitud esté fuera de este rango.

La pregunta que podría plantearse es si los genes VSG mal predichos por el modelo corresponden a algún tipo de estructura biológica determinada. Los datos con los cuales se cuenta en este trabajo no permiten la posibilidad de plantear un modelo con cuatro o más estados y reconocer una nueva zona, debido a que sólo se tiene información sobre tres regiones dentro de cada secuencia: gen VSG, región intergénica y gen VSG en la secuencia complementaria.

Para intentar responder esta interrogante se consideraron algunos de los genes mal predichos por el modelo (de la secuencia 9 tomada como ejemplo) y se compararon las secuencia de aminoácidos correspondientes utilizando BLAST, *Basic Local Alignment Search Tool* [6]. BLAST es un programa de alineamiento local de secuencias ADN o de proteínas. Compara una secuencia objetivo con secuencias que se encuentran en una gran base de datos, encontrando aquellas secuencias que tienen mayor parecido a la secuencia objetivo. Las consultas realizadas indican que en algunos casos los genes VSG mal predichos podrían ser genes no VSG, en otros casos podrían ser genes VSG no anotados hasta el momento y en otras consultas no se encuentra la correspondencia de la secuencia objetivo con ninguna región particular.

Finalmente, se concluye que al escoger como modelo para la búsqueda de genes VSG en el cromosoma 9 del genoma del *Trypanosoma brucei* un HMM sólo se elige el número de parámetros (estados

[6] BLAST disponible en http://www.ncbi.nlm.nih.gov/BLAST/

y símbolos). Se ingresa como datos para el funcionamiento del HMM solamente secuencias de ADN sin introducir ningún tipo de información biológica sobre el problema. Aún contando con sólo esos datos de inicio, HMM es capaz que encontrar los genes objetivo tanto en la secuencia como en la complementaria.

La inexactitud en las predicciones indica que el modelo no es del todo eficiente, por lo que para ajustar las predicciones resulta razonable la combinación de este modelo con otras técnicas.

coordenadas genes VSG secuencia complementaria	coordenadas genes VSG predichos secuencia complementaria
27500 - 29086	27560 - 29217
	29404 - 29594
	30216 - 30348
30672 - 32290	30752 - 32972
	33791 - 34145
34509 - 35973	34619 - 36483
37592 - 39046	37648 - 40083
40462 - 41986	40528 - 42688
	43715 - 43783
	43853 - 44335
44756 - 46342	44816 - 46881
	47692 - 48183
48819 - 50113	48910 - 50143
	52782 - 53268
53634 - 55059	53685 - 55339
56172 - 57665	56223 - 57969
	58034 - 59358
59975 - 61544	60042 - 62274
	62542 - 63076
63341 - 64935	63466 - 65166
	65468 - 65933

Cuadro 5.6: Localización de los genes VSG reales y predichos en la secuencia complementaria de la secuencia 9 con el modelo HMM con tres estados.

	grupo A			grupo B			grupo C		
	secuencia 3	secuencia 6	secuencia 9	secuencia 5	secuencia 7	secuencia 8	secuencia 1	secuencia 2	secuencia 4
Sensitividad	0.9472	0.9540	0.9559	0.9535	0.9405	-	0.9559	0.9280	0.9403
Especificidad	0.6712	0.7080	0.4022	0.3985	0.6629	-	0.5027	0.5435	0.4557
VSG anotados	7	5	7	3	8	0	16	21	6
VSG predichos	14	11	13	13	12	2	30	39	19
VSG no encontrados	0	0	0	0	0	0	0	0	0
	secuencia 3 compl.	secuencia 6 compl.	secuencia 9 compl.	secuencia 5 compl.	secuencia 7 compl.	secuencia 8 compl.	secuencia 1 compl.	secuencia 2 compl.	secuencia 4 compl.
Sensitividad	-	-	0.9508	0.9485	0.9568	0.9540	0.9226	-	0.9385
Especificidad	-	-	0.6157	0.5568	0.4833	0.5422	0.5233	-	0.6005
VSG anotados	0	0	11	31	19	7	5	0	16
VSG predichos	2	0	22	52	46	17	17	0	31
VSG no encontrados	-	-	0	0	0	0	0	-	0

Cuadro 5.7: Sensibilidad, especificidad y VSG encontrados para el modelo de tres estados con todas las secuencias.

Bibliografía

[1] Baum, L.E., Eagon, J.A. (1967). An inequality with applications to statistical estimation for probabilistic functions of Markov processes and to a model for ecology. Bulletin of the American Mathematical Society 73 (3), 360-363.

[2] Baum, L.E., Petrie, T., Soules, G., Weiss, N. (1970). A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains. The Annals of Mathematical Statistics 41 (1), 164-171.

[3] Berger, A.L., Della Pietra, S.D., Della Pietra, V.J.D. (1996). A maximum entropy approach to natural language processing. Computational Linguistics 22 (1), 39-71.

[4] Bernal, A., Crammer, K., Hatzigeorgiou, A., Pereira, F. (2007) Global Discriminative Learning for Higher-Accuracy Computational Gene Prediction. PLoS Comput Biol 3 (3): e54.

[5] Bishop, C.M. (2006). Patter Recognition and Machine Learning. Springer.

[6] Crick, F. (1970). Central Dogma of Molecular Biology. Nature 227, 561-563.

[7] Culotta, A., Kulp, D., McCallum, A. (2005). Gene prediction with conditional random fields. Technical Report UMCS2005028, University of Massachusetts, Amherst.

[8] Decaprio, D., Vinson, J.P., Pearson, M.D., Montgomery, P., Doherty, M., Galagan, J.E. (2007). Conrad: Gene prediction using conditional random fields. Genome Res. 17 (9), 1389-1398.

[9] Della Pietra, S., Della Pietra, V.J., Lafferty, J.D. (1997). Inducing features of random fields. IEEE Transactions on Pattern Analysis and Machine Intelligence 19 (4), 380-393.

[10] Dempster, A.P., Laird, N.M., Rubin, D.B. (1977). Maximum likelihood from incomplete data via the em algorithm. Journal of the Royal Statistical Society. Series B (Methodological) 39 (1), 1-38.

[11] Dietterich, T.G. (2002). Machine learning for sequential data: A review. In Proceedings of the Joint IAPR International Workshop on Structural, Syntactic, and Statistical Pattern Recognition. Springer-Verlag, London, UK, 15-30.

[12] Doherty, M.K. (2007) Gene Prediction with Conditional Random Fields. Master Thesis, Massachusetts Institute of Technology.

[13] Durbin, R., Eddy, S.R., Krogh, A., Mitchison, G. (1999). Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids. Cambridge University Press.

[14] Ewens, W., Grant, G. (2001). Statistical Methods in Bioinformatics: An introduction. Springer.

[15] Flicek, P. (2007). Gene prediction: compare and contrast. Genome Biology 8 (12): R233.

[16] Gross, S.S., Do C.B., Sirota M., Batzoglou S. (2007). CONTRAST: A discriminative, phylogeny-free approach to multiple informant de novo gene prediction. Genome Biology 8 (12): R269.

[17] Henderson, J., Salzberg, S., Fasman, K. (1996) Finding Genes in Human DNA with a Hidden Markov Model. Journal of Computational Biology 4 (2), 127-141.

[18] Jaynes, E.T. (1957). Information theory and statistical mechanics. The Physical Review 106 (4), 620-630.

[19] Jaynes, E.T. (1990). Notes on present status and future prospects. Maximum Entropy and Bayesain Methods. Grandy, W.T., Schick, L.H. Kluwer Academic Press, 1-13.

[20] Klinger, R., Tomanek, K. (2007) Classical probabilistic models and conditional random fields. Algorithm Engineering Report TR07-2-013, Department of Computer Science, Dortmund University of Technology.
Disponible en
http://www.scai.fraunhofer.de/fileadmin/images/bio/datamining/paper/crfklingertomanek.pdf.

[21] Kulp, D., Haussler, D., Reese, M.G., Eeckman, F.H. (1996). A generalized hidden Markov model for the recognition of human genes in DNA. In: Proceedings of the Fourth International Conference on Intelligent Systems for Molecular Biology, 134-142.

[22] Lafferty, J., McCallum, A., Pereira, F. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Procedings of the 18th International Conference on Machine Learning. Morgan Kaufmann, San Francisco, CA, 282-289.

[23] McCallum, A., Freitag, D., Pereira, F. (2000). Maximum entropy markov models for information extraction and segmentation. In: Proceedings of the 17th International Conference on Machine Learning. Morgan Kaufmann, San Francisco, CA, 591-598.

[24] McDonald, R., Pereira, F. (2005). Identifying gene and protein mentions in text using conditional random fields. BMC Bioinformatics S6.

[25] Mendel, J.G. (1866). Versuche über Pflanzenhybriden. Verhandlungen des naturforschenden Vereines in Brünn, Bd. IV für das Jahr, 1865 Abhandlungen, 3-47. Traducción al inglés: Druery, C.T, Bateson, W. (1901). Experiments in plant hybridization. Journal of the Royal Horticultural Society 26, 1-32.
Disponible en http://www.esp.org/foundations/genetics/classical/gm-65.pdf.

[26] Rabiner, L.R. (1989). A tutorial on hidden markov models and selected applications in speech recognition. Proceedings of the IEEE 77 (2), 257-286.

[27] Sutton, C., McCallum, A. (2007). An Introduction to Conditional Random Fields for Relational Learning. L. Getoor, B. Taskar, eds. MIT Press, chap. 4.

[28] Wallach, H.M. (2002). Efficient training of conditional random fields. Master's thesis, University of Edinburgh.

[29] Wallach, H.M. (2004). Conditional random fields: An introduction. Technical Report MS-CIS-04-21, University of Pennsylvania.

[30] Waterman, M.S. (2000). Introduction to Computational Biology: Maps, sequences and genomes. Chapman & Hall/CRC.

[31] Watson, J.D., Crick, F.H. (1953). Molecular structure of nucleic acids; a structure for deoxyribose nucleic acid. Nature 171, 737-738.

[32] Welch, L.R. (2003). Hidden Markov Models and the BaumWelch Algorithm. IEEE Information Theory Society Newsletter, 53 (4).

[33] World Health Organization (2006). Fact sheet 259.
Disponible en http://www.who.int/mediacentre/factsheets/fs259/en/.