Búsqueda y Recuperación de Información en Textos

Víctor Mijangos de la Cruz

III. Información en textos

Teoría de la información

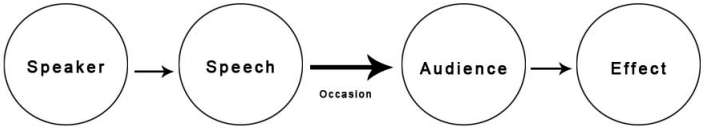
Modelos de comunicación

Un **modelo de comunicación** busca representar, de forma esquemática, el proceso que que se da en el acto comunicativo, cuando hablamos o comunicamos información a otras personas. Filósofos como **Aristóteles** han propuestos modelos de comunicación. El modelo aristotélico consta de (Aristóteles, *Retóricoa*, p. 72):

- quién habla (emisor)
- de qué habla (mensaje)
- para quién habla (receptor)

Modelo aristotélico

El **emisor** es quien produce el **mensaje**. El mensaje es recibido por un **receptor**, en quien produce un **efecto**. Además, se considera una **ocasión** o contexto en que el acto comunicativo se produce.



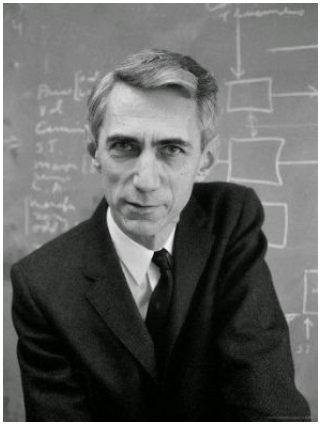
ARISTOTLE'S MODEL OF COMMUNICATION

Ejemplo de acto comunicativo

Un discurso político puede verse como un acto comunicativo:

- Emisor: Político que da un discurso
- Mensaje: El discurso que busca convencer a los votantes
- Receptor: El público (los electores)
- Efecto: Buscar que se vote por el emisor
- Ocasión: Época de elecciones

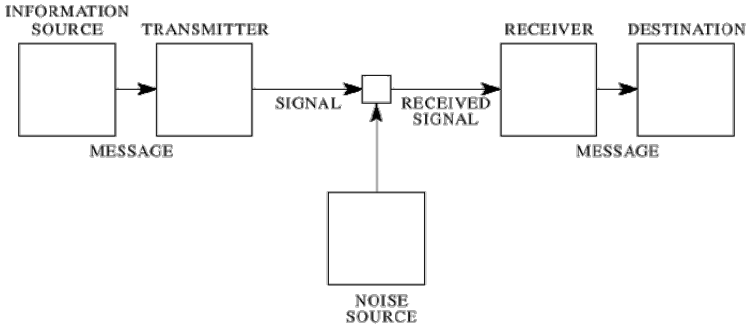
Teoría de la información



- La teoría de información se inaugura con el trabajo pionero de Claude E. Shannon: *A mathematical theory of communication*.
- Surge en el ámbito de las telecomunicaciones. Se plantea el problema de reproducir un mensaje a través de un canal de comunicación.
- Sin embargo, su aplicación se extiende a diferentes áreas del conocimiento (estadística, sistemas complejos, lingüística, aprendizaje de máquina, ...).

Un modelo matemático de la comunicación

En el ámbito de las telecomunicaciones, Claude E. Shannon se preguntó cómo modelar la comunicación **matemáticamente**.



Un modelo matemático de la comunicación

A diferencia del modelo aristotélico, el modelo de Shannon se enfoca en dos cosas:

- **Codificación**; El mensaje μ debe ser codificado por el transmisor para poder transmitirse.

$$x = R(\mu)$$

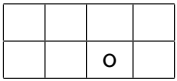
- **Decodificación**: El mensaje debe ser decodificado por el receptor, con la posibilidad de pérdida de información (debido al **ruido**).

$$\hat{\mu} = f(x)$$

Codificación de mensajes

Arieh Ben-Naim (2007, pp.72-74) propone el siguiente juego para ejemplificar el proceso de codificación:

> *Se tiene un tablero con 8 casillas Y en donde se ha escondido una moneda. Sin importar dónde está la moneda, se desea encontrar el mínimo de preguntas (sí o no) necesarias para poder saber con exactitud en que casilla está la moneda.*



Codificación del mensaje

Podemos expresar este juego matemáticamente. En el tablero Y, cada casilla del tablero será una variable aleatoria $y_1, y_2, ..., y_8$ que puede tomar un valor 0 o 1 (0 si no contiene a la moneda, 1 sí la contiente).

El tablero anterior se vería entonces como:

$y_1 = 0$	$y_2 = 0$	$y_3 = 0$	$y_4 = 0$
$y_5 = 0$	$y_6 = 0$	$y_7 = 1$	$y_8 = 0$

Cada variable representa un **bit** (binary unit of information).

Codificación del mensaje

Una mala estrategia consiste en pregunrar:

1. ¿La moneda está en y_1?
2. ¿La moneda está en y_2?
3. ¿La moneda está en y_3?
4. ¿La moneda está en y_4?
5. ¿La moneda está en y_5?
6. ¿La moneda está en y_6?
7. ¿La moneda está en y_7?
8. ¿La moneda está en y_8?

Se requieren 8 bits para codificar el mensaje bajo esta estrategia. ¿Se puede reducri el número de bits?

Codificación del mensaje

La codificación del mensaje equivale a **comprimirlo**; es decir, encontrar el mínimo número de variables aleatories que puedan expresar el mismo mensaje.

En este ejemplo, procedemos por medio de una estrategia discriminativa:

1. ¿La moneda está en la parte superior del tablero?
2. ¿La moenda está en la parte izquierda de las casillas restantes?
3. ¿La moneda está en la casilla izquierda?

Esto reduce el número de bits a sólo 3. Es posible reconstruir cualquier configuración original del tablero con 3 bits.

$$\begin{pmatrix} y_1 & y_2 & \dots & y_8 \end{pmatrix} \mapsto \begin{pmatrix} x_1, x_2, x_3 \end{pmatrix}$$

Codificación del mensaje

¿Qué pasa ahora si tenemos un tablero con 16. 32, 64 o más casillas? ¿Cuál es el número menor de información que se requiere para codificar estos tableros?

Cada variable (bit) puede tomar dos valores (0 ó 1). ¿Cuántas de estas variables abarcan todo el tablero Y? Necesitamos resolver para H la ecuación:

$$2^H = |Y|$$

Esto nos deja con:

$$H = \log_2 |Y| \tag{1}$$

Codificación de distribuciones

Observamos que el tablero tiene **distribución uniforme**, de lo cual podemos ver que:

$$\log_2 |Y| = -\log_2 \frac{1}{|Y|} = -\log_2 p(Y = y_i)$$

Cuando el proceso que estamos observando no es discreto o no es fácil de modelar, pero conocemos su distribución, podemos utilizar esta distribución para determinar su codificación.

Información (bits)

La información (en bits) de un evento $X = x$ con probabilidad $p(X = x)$ es el número de bits necesarios para codificarlo. Esto es:

$$I(x) = -\log_2 p(X = x)$$

Codificación Shannon-Fano

Un evento puede codificarse en bits, o bien si contamos con un alfabeto Σ de longitud $N = |\Sigma|$ podemos encontrar un código óptimo bajo la lógica anterior.

Codificación Shannon-Fano

Dado un alfabeto Σ con longitud $N = |\Sigma|$, y dado una variable X con eventos $x_1, ..., x_n$, la codificación Sannon-Fano para esta variable determina la longitud l_i para el evento x_i como:

$$l_i = \lceil -\log_N p(X = x_i) \rceil$$

Los eventos más probables requieren de menos símbolos. Mientras que eventos raros requieren de una longitud de código mayor.

Ejemplo de información

Digamos que lanzar una moneda corresponde a una variable X. Pueden suceder los eventos:

- $X = \textit{cara}$
- $X = \textit{cruz}$

La información del evento $X = \textit{cara}$ se puede calcular, sabiendo que su probabilidad es $\frac{1}{2}$:

$$I(\textit{cara}) = -\log_2 \frac{1}{2} = 1 \textit{bit}$$

De igual forma, vemos que la información de $X = \textit{cruz}$ es de 1 bit. Cada evento requiere un bit para codificarse (ubn código para este evento podría ser $C = \{0, 1\}$).

Ejemplo de información

Se tiene un saco donde se han vertido 32 frijoles y 2 lentejas. Tenemos dos eventos: $X = \textit{frijoles}$ y $X = \textit{lentejas}$ con probabilidades $\frac{16}{17}$ y $\frac{1}{17}$ respectivamente.

La información para frijoles es:

$$I(\textit{frijoles}) = -\log_2 \frac{16}{17} \approx 0.087$$

Y para lentejas es:

$$I(\textit{lentejas}) = -\log_2 \frac{1}{17} \approx 4.087$$

Tipos de códigos

Codificar los mensajes en un alfabeto Σ se puede hacer de diversas maneras, reconocemos **tipos de códigos** según sus propiedades.

Podemos definir una **función de codificación**

$$C : \Omega \to \Sigma^*$$

que va del espacio de los mensajes (eventos) hacia las cadenas del alfabeto.

Por ejemplo, para $\Omega = \{blanco, amarillo, gris, negro\}$ podemos tener el código:

$$\begin{aligned} C(blanco) &= 0 \\ C(amarillo) &= 0 \\ C(gris) &= 10 \\ C(negro) &= 0 \end{aligned}$$

Códigos singulares

Códigos singulares

Las codificaciones singulares son códigos en los que dos o más mensajes pueden tener el mismo código. Es decir, C **no** es inyectiva.

Códigos no singulares

Una codificación singular asigna a cada mensaje un código diferente. Es decir C es inyectiva.

Del ejemplo anterior podemos definir el siguiente código singular:

$$C(\textit{blanco}) = 0$$
$$C(\textit{amarillo}) = 010$$
$$C(\textit{gris}) = 00$$
$$C(\textit{negro}) = 10$$

Codificación de cadenas

En algunos casos se pueden enviar cadenas de mensajes como por ejemplo:

blanco negro

Bajo la codificación que hemos definido, el código de esta cadena sería 010, que se confunde con el código de 'amarillo'.

El problema que surge es que el código de 'blanco' es un **prefijo** del de 'amarillo'.

Prefijo

Un prefijo es una cadena $x \in \Sigma^*$ tal que existe otra cadena y tal que $w = x \cdot y$ es parte del código (o lenguaje).

Códigos unívocamente decodificables

Para lidiar con el envío de cadenas de mensajes se propone una codificación que no tenga ambigüedades:

Códigos unívocamente decodificables

Una codificación unívocamente decodificable es una codificación no singular en donde cada código y cadena de códigos tienen una codificación distinta.

Del ejemplo anterior podemos definir la siguiente codificaicón unívocamente decodificable:

$$C(\textit{blanco}) = 10$$
$$C(\textit{amarillo}) = 01$$
$$C(\textit{gris}) = 11$$
$$C(\textit{negro}) = 110$$

Códigos instantáneos

Una restricción más fuerte es impedir que los códigos sean prefijos entre sí.

Códigos instantáneos

Una codificación instantánea asigna a cada mensaje un código que no puede aparecer como prefijo en ningún otro código.

Del ejemplo anterior podemos definir la siguiente codificaicón instantánea:

$$
\begin{aligned}
C(\textit{blanco}) &= 0 \\
C(\textit{amarillo}) &= 10 \\
C(\textit{gris}) &= 110 \\
C(\textit{negro}) &= 111
\end{aligned}
$$

Tipos de códigos

Además de los tipos de códigos que podemos utilizar, una codificación **óptima** se fija en la longitud de código (por ejemplo con Shannon-Fano).

Un resumen de los tipos de codificación se presenta a continuación:

Mensaje	**Singular**	**No singular**	**UD**	**Instantáneo**	**Shannon-Fano (unif.)**
blanco	0	0	10	0	00
amarillo	0	010	00	10	01
gris	10	01	11	110	10
negro	0	10	110	111	11

Longitud esperada

En teoría de la información los métodos de codificación buscan minimizar la **longitud esperada** L. Si X es una v.a. con n salidas (mensajes), $x_1, ..., x_n$ y C es la función de codificación tal que $|C(x_i)| = l_i$, la longitud esperada es:

$$L = \mathbb{E}_{X \sim p}[|R(X)|] = \sum_{i}^{n} p(x_i) l_i \tag{2}$$

La longitud esperada es una forma de medir la **eficiencia** del código en base a su distribución.

Desigualdad de Kraft

Teorema (Desigualdad de Kraft)

Sea N la longitud del alfabeto Σ para un código. Sean $l_1, l_2, ..., l_n$ las longitudes del código. Un código es instantáneo si y sólo si se cumple la desigualdad:

$$\sum_i N^{-l_i} \leq 1$$

Si se crea un árbol N-ario con ramas que sean los códigos observamos: (1) Si $l_{max} = \max\{l_i : i = 1, .., n\}$, entonces el número de nodos del árbol es menor o igual a l_{max}; (2) Como son códigos instantáneos, ningún código puede ser padre de otro código; (3) un código a nivel l_i tiene $\frac{N^{l_{max}}}{N^{l_i}} = N^{l_{max}-l_i}$ descendientes; (4) y el número de nodos debe ser $\leq N^{l_{max}}$. Así $\sum_i N^{l_{max}-l_i} \leq N^{l_{max}}$ de donde $\sum_i N^{-l_i} \leq 1$.

Códigos óptimos

Una forma de establecer las longitudes $l_1, l_2, ..., l_n$ de un código óptimo es plantearse un problema de optimización con restricciones:

$$\mathcal{J} = \sum_{i=1}^{n} p(x_i) l_i + \lambda \sum_{i} N^{-l_i}$$

Derivando, obtenemos que:

$$\frac{\partial \mathcal{J}}{\partial l_i} = p(x_i) + \lambda N^{-l_i} \log N$$

De donde obtenemos $N^{-l_i} = \frac{p(x_i)}{\lambda \log N}$ y $\lambda = \frac{1}{\log N}$, que nos deja con $N^{-l_i} = p(x_i)$. Por tanto:

$$l_i^* = -\log_N p(x_i)$$

Entropía

A partir de minimizar la longitud de código esperado restringido a que se cumpla la desigualdad de Kraft, obtenemos la cota inferior de la longitud esperada:

$$-\sum_{i=1}^{n} p(x_i) \log_N p(x_i) \leq \sum_{i=1}^{n} p(x_i) l_i$$

Esta cota inferior define un concepto primordial para la teoría de la información:

Entropía

Dada una variable aleatoria X con función de probabilidad $p(x_i)$, la entropía (medida en bits) de esta variable se define como:

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log_2 p(x_i)$$

Propiedades de la entropía

A partir de la definición de la entropía, podemos determinar las siguientes propiedades:

- La entropía determina la **longitud del código (binario)** más pequeño que describe los eventos de una variable aleatoria.
- Es la **esperanza** de la información $H(X) = \mathbb{E}\big[I(X)\big]$.
- Es **positiva**: $H(X) \leq 0$.
- La base determina la longitud del alfabeto. Se puede **cambiar la base**: $H_N(X) = (\log_N 2) H(X)$.
- Cuando se usa la **base natural** se habla de que la netropía mide los **nats** (*natural units*).

Entropía: ejemplo

La entropía de **lanzar una moneda** es:

$$H(Y) = -\frac{1}{2}1 - \frac{1}{2}1 = 1$$

Supóngase que $p(X = 0) = \frac{16}{17}$ y $p(X = 1) = \frac{1}{17}$, la entropía es:

$$\begin{aligned} H(Y) &= -\frac{16}{17}\log_2\frac{16}{17} - \frac{1}{17}\log_2\frac{1}{17} \\ &= \frac{16}{17}0.087 - \frac{1}{17}4.087 = 0.322 \end{aligned}$$

$$H(Y) = -\frac{16}{17}0.087 - \frac{1}{17}4.087 = 0.322$$

Un evento que peude predecirse con mayor facilidad presentará menos entropía; es decir, menos incertidumbre.

Entropía conjunta

Entropía conjunta

Dadas dos variables aleatorias X, Y con función de probabilidad conjunta $p(x, y)$, la entropía conjunta (en bits) de las variables está dada por:

$$H(X, Y) = \sum_x \sum_y p(x, y) \log_2 p(x, y) = -\mathbb{E}_p \log_2 p(X, Y)$$

Podemos extender el concepto de entropía conjunta:

Entropía conjunta

Dada una familia de variables $X_1, X_2, ..., X_n$ con función de probabilidad conjunta $p(x_1, x_2, ..., x_n)$ la entropía conjunta es:

$$H(X_1, ..., X_n) = -\sum_{x_1} \cdots \sum_{x_n} p(x_1, ..., x_n) \log_2 p(x_1, ..., x_n)$$

Entropía condicional

Entropía condicionada a un evento

Dada una variable aleatoria X condicionada a un evento $Y = y$ con probabilidad $p(x|Y = y)$, la entropía está determinada como:

$$H(X|Y = y) = -\mathbb{E}_p \log_2 p(X|Y = y) = -\sum_x p(x|Y = y) \log_2 p(x|Y = y)$$

Entropía condicional

Sean X y Y dos variables aleatorias con distribuciones $p(x)$ y $p(y)$ respectivamente; y distribución condicional $p(x|Y = y)$. La entropía condicional de X dado Y se determina como:

$$H(X|Y) = \mathbb{E}_y H(X|Y = y) = -\sum_y \sum_x p(x, y) \log_2 p(x|Y = y)$$

Información mutua

Una función importante de la teoría de la información es relacionar la **cantidad de información que comparten** dos variables.

Información mutua

Sean X y Y dos variables aleatorias con funciones de probabilidad $p(x)$, $p(y)$, respectivamente, y distribución conjunta $p(x, y)$. La dependencia entre ambas se determina de la siguiente forma:

$$\begin{aligned} MI(X;Y) &= H(X) - H(X|Y) \\ &= H(Y) - H(Y|X) \\ &= \sum_{x,y} p(x,y) \cdot \log_2 \frac{p(x,y)}{p(x)p(y)} \end{aligned}$$

Información mutua

Las **propiedades** de la informaicón mutua son las siguientes:

1. $MI(X;Y) = MI(Y;X)$
2. $MI(X;Y) = 0$ sii X, Y son independientes.
3. $MI(X;X) = H(X)$

También se utiliza la información mutua para medir la capacidad de un canal de comunicación. Esta **capacidad de canal** se define como:

$$C = \max MI(X;Y)$$

donde X es la señal de entrada y Y representa la señal de salida.

Usos de la teoría de la información

La **teoría de la información** tiene una gran cantidad de aplicaciones. En particular, dentro de la **recuperación de información**, podemos enumarar las siguientes:

- Extracción de términos
- Representación de términos en espacios vectoriales
- Representación de documentos en espacios vectoriales

Extracción de términos

Palabras clave

Palabra clave

Una palabra clave es una secuencia de una o más palabras que provee una representación completa del contenido del documento.

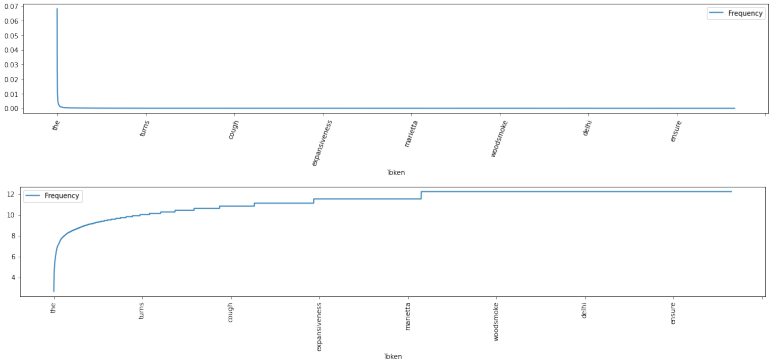
Las palabras claves son relevantes en tanto nos **proveen de queries** para sistemas de RI.

Podemos encontrar dos aproximaciones a la extracción de palabras clave:

- **Formales**: Basadas en la estructura de los términos, generalmente definida por etiquetas POS.
- **Estadísticos**: Basada en la información estadística de los términos dentro de las colecciones.

Extracción de términos por información

La **probabilidad** y la **información** están inversamente correlacionadas. Los términos menos probables son los más informativos.



Sin embargo, los términos con poca probabilidad no suelen ser buenos candidatos a palabras clave.

Distribución de palabras claves

Los hapax o palabras con frecuencia 1 no representan el corpus ni los documentos de manera precisa, por tanto no son buenos candidatos a palabras clave.

Las palabras clave de un documento cumplen dos propiedades importantes en su distribución:

- Dentro del documento la **frecuencia** del token es alta.
- El token **no** se distribuye uniformemente en los documentos.

Podemos determinar estadísticas que tomen en cuenta estos factores para poder hacer búsquedas de información en documentos particulares.

Inverse document frequency

Para que, dentro de un documento, un token sea **candidato a palabra clave**, debe ser **propio de ese documento**; es decir, no aparecer en otros muchos documentos.

Inverse Document Frequency (IDF)

Dado un corpus de documentos $D = \{d_1, ..., d_N\}$, la frecuencia inversa en documentos o IDF de un término w_i se define como:

$$idf_i = -\log \frac{|\{d : w_i \in d\}|}{|D|}$$
$$= \log \frac{|D|}{|\{d : w_i \in d\}|}$$

Term frequency

El IDF indica la distribución que un término tiene en el corpus, pero todavía no indica si puede ser un candidato a palabra clave.

El segundo punto que tomamos es la **frecuencia de término** dentro de un documento, la cuál se puede calcular de diversas formas:

- **Frecuencia absoluta**: La frecuencia absoluta del término i en el documento j: $tf_{i,j} = f_{i,j}$.
- **Frecuencia relativa**: La frecuencia relativa del término i a través del corpus:

$$tf_{i,j} = \frac{f_{i,j}}{\sum_{i,j} f_{i,j}}$$

- **Frecuencia booleana**: 1 si el término está en el documento y 0 en otro caso.
- **Frecuencia normalizada**: Normaliza con respecto al término más frecuente en el documento:

$$tf_{i,j} = \frac{f_{i,j}}{\max\{f_{k,j} : w_k \in d_j\}}$$

Frecuencia doblemente normalizada

La forma más común de determinar la frecuencia del término dentro de un documento es a partir de la **frecuencia doblemente normalizada**, que se define como:

$$tf_{i,j} = \alpha + (1 - \alpha)\frac{f_{i,j}}{\max\{f_{k,j} : w_k \in d_j\}}$$

Donde $0 \leq \alpha \leq 1$ es un parámetro, generalmente con valor $\alpha = 0.5$ (**frecuencia aumentada**).

Otros tipos de frecuencia surgen de esta fórmula: si $\alpha = 0$ tenemos la frecuencia normalizada; si $\alpha = 1$ tenemos la frecuencia booleana.

TF-IDF

La medida de TF-IDF de un término $w_i i$ en el documento d_j se determina entonces como:

$$tfidf_{i,j} = tf_{i,j} \cdot idf_i$$

El valor del TF-IDF puede servir para **estimar las palabras clave** de un documento. Asimismo, se puede usar como un *score* para recuperar documentos:

TF-IDF score

Dada una query compuesta por los términos $q = \{w_1, ..., w_T\}$, el TF-IDF score se determina como:

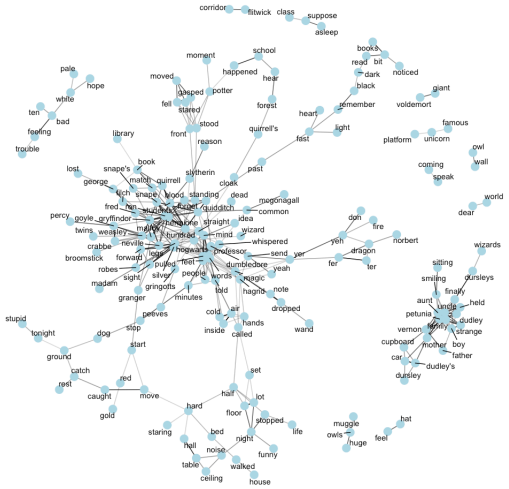
$$score(q, d_j) = \sum_{i=1}^{T} tfidf_{i,j}$$

Extracción términos por RAKE

Otro método para extracción de palabras clave es **RAKE (Rapid Automatic Keyword Extraction)** que se basa en los siguientes supuestos:

- Los términos se componen de diferentes palabras.
- No contienen signos de puntación ni otros elementos ortográficos.
- Los candidatos a términos serán aquellos que se relacionen mayormente con otros términos.
- Los candidatos a términos no serán los términos más frecuentes; son inversamente proporcionales a su frecuencia.

Formalizando relaciones de términos



Para conformar las relaciones que se dan entre los términos, el método de RAKE propone:

- Obtener los contextos (frases u oraciones) de un documento.
- Dos palabras están relacionadas si aparecen en la misma oración.
- El peso de su relación está dado por las veces que aparecen juntas.

Esto conforma una **gráfica** de relaciones entre términos.

Gráfica de relación de términos

En la **gráfica de relaciones** cada términos w_i es un nodo i conectado a otro nodo j si y sólo si aparece en el mismo contexto con w_j.

Gráfica de relación de términos

Una gráfica pesada $G = (V, E, \phi)$ relaciona los términos si cada nodo $v_i \in V$ representa a un término w_j, si las aristas $e_{i,j} \in E$ relaciona los términos w_i y w_j y si la función de peso es: $\phi(e_{i,j}) = coocurrence(w_i, w_j)$.

Asociada a esta gráfica tenemos una matriz de adyacencia $A \in \mathbb{R}^{n \times n}$ definido como:

$$A = (a_{i,j}) = \phi(e_{i,j})$$

Índice RAKE

Para determinar cuántas relaciones están determinadas por un término, se utiliza el **grado**, calculado a partir de la matriz de adyacencia como:

$$deg(w_i) = \sum_j A_{i,j}$$

RAKE

El score o índice de RAKE para un término está determinado por el grado del término entre su frecuencia; es decir, se define como:

$$score(w_i) = \frac{deg(w_i)}{f_i}$$

Las palabras con un score más alto son candidatos a términos.

Extracción de términos multi-palabra

Para la extracción de términos **multipalabra** se puede usar una aproximación **formal** que se fija en la **estructura** de los términos.

La estructura de los términos se puede determinar por sus **etiquetas POS**.

En español se puede proponer algunas estructuras de términos multipalabra:

- Artículo + Sustantivo
- Artículo + Sustantivo + Adjetivo
- Artículo + Sustantivo + Preposición + Sustantivo
- Sustantivo + Preposición + Sustantivo

La extracción de términos se lleva a cabo al encontrar **coincidencias** en el documento con estos patrones.

Problemas de la perspectiva formal

Sin embargo, extraer términos únicamente basado en los patrones no implica que se extraigan términos del documento. Aquí cuentan dos factores más:

- Candidatos que cumplen con la estructura pero que no tienen mucha **frecuencia** dentro del documento.
- La **longitud** del término (número de palabras) puede ser un factor que influencie la extracción.
- Pueden extraerse términos contenidos dentro de otros (**anidados**); ejemplo, 'Artículo + Sustantivo' está contenido en 'Artículo + Sustantivo + Adjetivo'.

C-Value

Para lidiar con estos problemas, se propone incorporar una aproximación **estadística**:

C-Value

El C-Value de un candidato a término (multipalabra) w está determinado como:

$$CValue(w) = \begin{cases} \log_2 |w| \cdot f_w & \text{si no está anidado} \\ \log_2 |w| \left(f_w - \frac{1}{|T_w|} \sum_{v \in T_w} f_v \right) & \text{si está anidado} \end{cases}$$

donde f_w es la frecuencia del término w, y T_w son los candidatos que contienen a w.

El C-Value pondera los términos más largos, siendo estos mejores candidatos a términos. De esta forma, los términos anidados tenderán a tener un valor más bajo.

Ejemplo: C-Value

Supóngase que se tienen los siguientes candidatos a términos multipalabra:

- 'Universidad Nacional', frecuencia 10
- 'Universidad de Toluca' frecuencia 20
- 'La Universidad de Toluca' frecuencia 15

El candidato 'Universidad Nacional' tiene C-Value $\log_2 2 \cdot 10 = 10$. Mientras el candidato 'La Universidad de Toluca' tiene $\log_2 4 \cdot 15 = 2 \cdot 15 = 30$.

El candidato 'Universidad de Toluca' está anidado (contenido) en 'La Universidad de Toluca', por lo que su C-Value es:

$$\log_2 3\big(20 - 15\big) \approx 1.58 \cdot 5 = 7.9$$

NC-Value

Además de el C-Value, se puede incorporar información contextual de los términos para poder mejorar los pesos de estos. En particular:

NC-Value

El NC-Value de un candidato a término (multipalabra), w, está dado por:

$$NCValue(w) = \lambda CValue(w) + (1 - \lambda) \sum_{t \in C_w} f_{t|w} weight(t)$$

donde C_w son los contextos de w, $f_{t|w}$ es la coocurrencia de t con w, $weight(t)$ es una función de peso asignado al término t, y λ es un parámetro.

El valor del parámetro es comúnmente $\lambda = 0.8$, y la función de peso se determina como:

$$weight(t) = \frac{|\{w : t \text{ coocurre con } w\}|}{\sum_i f_i}$$

Vinculación de términos

En muchos casos, para la extracción de información a través de queries es importante saber qué términos están relacionados entre sí y con que grado.

Vinculación de términos

La vinculación de términos determina un valor numérico que relaciona dos términos w y v.

la vinculación de términos nos puede decir que tan similar es una query con respecto a los términos de un índice.

Por ejemplo, la query 'casa' tendrá una alta vinculación con 'hogar' o 'choza'.

Vinculación por PMI

Una primera aproximación para determinar la vinculación entre dos términos en un documento es a partir de observar la **información** que comparten.

Pointwise Mutual Information

La Pointwise Mutual Information (PMI) o información punto a punto determina la información compartida entre dos términos w_i y w_j en base a su distribución como:

$$PMI(w_i; w_j) = \log_2 \frac{p(w_i, w_j)}{p(w_i)p(w_j)}$$

Tal que $p(w_i, w_j)$ es la probabilidad de coocurrencia en un contexto y $p(w_i)$ son las probabilidades de los términos.

PMI

La función de PMI cumple propiedades que nos ayudan a determinar la vinculación de dos términos:

- Si w_i y w_j son eventos independientes, entonces $PMI(w_i; w_j) = 0$.
- Es simétrica: $PMI(w_i; w_j) = PMI(w_j; w_i)$.
- No está acotada: $-\infty < PMI(w_i; w_j) < \infty$

Este último punto indica que puede haber valores negativos. Para que el PMI sea estrictamente positivo se utiliza la llamada **Positive Pointwise Mutual Information**:

$$PPMI(w_i; w_j) = \max\{0, \log_2 \frac{p(w_i, w_j)}{p(w_i)p(w_j)}\}$$

PMI de rango k

Una forma común de trabajar con el PMI es tomar aquellos valores que superen sólo cierto rango, mientras que si son menor a un rango se llevan a 0, para esto se utiliza la siguiente función:

PMI de rango k

La información mútua punto a punto (PMI) de rango k está determinado por un valor $k \in \mathbb{R}^*$ y se define como:

$$PPMI(w_i; w_j) = \max\{0, \log_2 \frac{p(w_i, w_j)}{p(w_i)p(w_j)} - \log_2 k\}$$

Los valores menores a $.log_2 k$ serán llevados a 0. Generalmente, se escogen valores $k = 5, 10, 15$.

Vinculación de términos con PMI

El valor de PMI entre dos términos puede considerarse un índice de su vinculación. Pero en la práctica suelen tomarse mejores medidas de vinculación entre términos.

Vinculación por PMI

Dados dos términos w_i y w_j, la vinculación entre ambos se puede determinar como:

$$Vinc(w_i, w_j) = \sum_n PMI(w_i, w_n) PMI(w_j, w_n)$$

Dos términos están más vinculados entre sí, entre más contextos comparten: entre **más palabras similares los acompañan**.

Vinculación por rasgos

Otra forma de vincular los términos es a partir de observar cuántos **rasgos comparten** entre sí.

Rasgo de un término

Un rasgo de un término hace referencia a un elemento (generalmente una palabra) que describe la semántica de dicho término.

El rasgo de un término puede ser sus relaciones con otras palabras, o las palabras que se usan para definirla.

De esta forma, podemos denotar a un término w conformado por sus rasgos como:

$$w = \begin{pmatrix} \phi_1(w) & \phi_2(w) & \cdots \phi_n(w) \end{pmatrix}$$

Vinculación de Lin

A partir de los rasgos, podemos determinar si dos términos están vinculados entre sí. Existen diferentes estrategias:

Vinculación de Lin

Si w y u son dos términos. Su nivel de vinculación está determinado por:

$$LIN(w,u) = \frac{common(w,u)}{union(w,u)}$$

En este caso, la función *common* hace referencia a la intersección de rasgos:

$$common(w,u) = |w \cap u|$$

Y *union* a los rasgos diferenciales, o la unión de los rasgos: $union(w,u) = |w \cup u|$

Vinculación de Lin

Una variante de la vinculación de Lin toma en cuenta una escala logarítmica:

Vinculación de Lin

Si w y u son dos términos. Su nivel de vinculación está determinado por:

$$LIN(w,u) = \frac{\log_2 common(w,u)}{\log_2 difference(w,u)}$$

La función *common* es igual que la anterior, pero la función *difference* hace referencia a los rasgos diferenciales:

$$difference(w,u) = |\{\phi_i : \phi_i \in w, \phi_i \notin u\}|$$

Vinculación dirigida

En muchos caso se busca determinar una **relación jerárquica** entre los términos. Dos tipos esenciales de relaciones que se encuentran son:

- **Hiperónimo**: Un término es hiperónimo de otro cuando este término puede funcionar como un rasgo del otro.
- **Hipónimo**: Un término es hipónimo de otro cuando contiene a este otro término entre sus rasgos.

Esta relación es: **Hiperónimo** $\rightarrow$ **Hipónimo**.

Generalmente la relación se determina por **is_a**: gato IS_A animal.

Precisión media

Para establecer una relación jerárquica entre palabras se propone el uso de una métrica de precisión media. Para esta se definen dos elementos:

- Un factor que determina cuáles de los rasgos se pueden encontrar en los términos:

$$P(r; w, u) = \frac{|\{\phi_i : w_i \cap u_i, i = 1, ..., r\}|}{r}$$

- Una variable que determina si el rasgo está presente en el término:

$$rel(\phi) = \begin{cases} 1 & \text{si } \phi \in w \\ 0 & \text{en otro caso} \end{cases}$$

Vinculación por precisión media

Precisión media

La vinculación por precisión media (average precision) vincula dos términos de forma jerárquica $w \rightarrow u$ y está definida como:

$$AP(w \rightarrow u) = \frac{1}{|u|} \sum_{r=1}^{|w|} P(r; w, u) \cdot rel(\phi_r(w))$$

Algunos problemas que se apuntan con esta métrica son:

- Dividir sobre $|u|$ no captura bien la idea de inclusión de rasgos.
- Todas los rasgos tienen la misma relevancia.

Vinculación por precisión media

Para corregir esto se propone una medida de relevancia:

$$rel'(\phi) = \begin{cases} 1 - \frac{rank(\phi, w)}{|w|+1} & \text{si } \phi \in w \\ 0 & \text{en otro caso} \end{cases}$$

Precisión media con ranking

La vinculación por precisión media (average precision) vincula dos términos de forma jerárquica $w \to u$ y está definida como:

$$APinc(w \to u) = \frac{1}{|w|} \sum_{r=1}^{|w|} P(r; w, u) \cdot rel'(\phi_r(w))$$

Vinculación dirigida

Corrigiendo los valores de Precisión media se puede obtener una medida dirigida para la vinculación de dos términos.

Vinculación dirigida

La vinculación dirigida vincula dos términos de forma jerárquica $w \rightarrow u$ y está definida como:

$$balAPinc(w \rightarrow u) = \sqrt{LIN(w,u) \cdot APinc(w \rightarrow u)}$$

Ejemplo: vinculación

Consideremos los dos términos con los siguientes conjuntos de rasgos:

- husky = animal, doméstico, cánido, perro
- gato = animal, doméstico, felino

Tenemos que:

$$LIN(husky, gato) = \frac{2}{7} \approx 0.28$$

O bien:

$$LIN_{log}(husky, gato) = \frac{\log_2 2}{\log_2 3} \approx 0.63$$

Ejemplo: vinculación

Podemos calcular la vinculación de 'gato $\rightarrow$ husky' de tal forma que necesitamos ponderar los rasgos:

$$\begin{aligned} rel'(animal) &= 1 - 3/4 = 0.25 \\ rel'(domestico) &= 1 - 2/4 = 0.5 \\ rel'(felino) &= 1 - 3/4 = 0.75 \end{aligned}$$

Para 'cánido' y 'perro' el valor es 0. De tal forma que obtenemos:

$$APinc(gato \rightarrow husky) = \frac{1}{3}\Big(1 \cdot 0.25 + 2 \cdot 0.5 + 0 \cdot 0.75\Big) = \frac{1.25}{41} \approx 0.41$$

Y de aquí:

$$balAPinc(gato \rightarrow husky) = \sqrt{0.28 \cdot 0.41} \approx 0.33$$

Vinculación por aparición en documentos

En algunos casos, no contaremos con un conjunto de rasgos que describan a los términos que queremos vincular, pues este es un recurso léxico costoso.

Se pueden representar los rasgos de un término como los **documentos que son recuperados** cuando este término es query.

Sus **rasgos** pueden ser las listas de posting o los vectores binarios de la representación booleana.

El **ranking** de la relevancia puede ser el ranking del documento en la búsqueda.

Extracción de conocimiento con etiquetado automático

Conocimiento lingüístico en documentos

Los documentos que conforman las colecciones son tratados como conjuntos de términos. Pero en ellos existe **conocimiento** no explícito que puede inferirse por diversos métodos.

Parte del conocimiento que se encuentra en los documentos es:

- El **lenguaje** al que pertenecen los documentos.
- Los **temas** o tópicos que los documentos tratan.
- La estructura del lenguaje que se da: como el **tipo de palabras** al que pertenecen los términos.
- Nombres de personas, instituciones, fechas e **información sobre entidades** que refiere el documento.

Aprendizaje supervisado

Una forma de obtener conocimiento a partir de colecciones es **enseñar** a la máquina aquello quiere obtener por medio de **ejemplos**:

Aprendizaje supervisado

Se llama aprendizaje supervisado a la familia de algoritmos que determinan una solución a un problema a partir de la observación de un conjunto de ejemplos solucionados (bajo una supervisión).

El **conjunto de ejemplos** se define como un conjunto (dataset) supervisado de la siguiente forma:

$$\mathcal{S} = \{(x, y) : x \in \mathbb{R}^d, y \in Y\}$$

x es un vector que describe a los objetos del problema con d rasgos; y es la supervisión (solución).

Clasificación

Clasificación

La clasificación es un proceso de aprendizaje supervisado en donde se asigna una clase o categoría a un objeto o vector de entrada.

En este caso, $y \in Y$ es una clase o categoría que describe los datos. Por ejemplo, si tenemos el vector de entrada d-dimensional:

$$x = \begin{pmatrix} x_1 & x_2 & \cdots & x_d \end{pmatrix}$$

Un **problema de clasificación** buscará asignar una clase $\hat{y}$ determinada como:

$$\hat{y} = \arg\max_y p(Y = y|x)$$

Clasificación de lenguas

Un primer problema que se puede tratar como **clasificación** es determinar **cuál lengua** pertenece al texto.

Para realizar la clasificación necesitamos:

- Un **corpus multilingüe** en donde cada documento esté asociado a la lengua a la que pertenece.
- Un método para conformar los **rasgos** que tomará el algoritmo de clasificación.
- Un **algoritmo de clasificación** que nos permita aprender a clasificar a través de los ejemplos.

Obtención de rasgos para identificar lenguas

Las palabras, en general, no tienen una frecuencia tan alta para ser buenos indicadores del idioma.

Las **stopwords** pueden ser útiles, pero podemos incluir más rasgos para la clasificación.

Considerar secuencias de caracteres o **n-gramas** de caracteres nos permite capturar rasgos para clasificar las lenguas.

Rango	Español	Inglés	Náhuatl
1	de	th	tl
2	en	he	an
3	es	in	ca
4	la	er	ui
5	os	an	hu

Clasificador Bayes ingenuo

Una aproximación sencilla y útil para la clasificación de lengua es utilizar el clasificador de Bayes ingenuo.

Este algoritmo se basa en obtener una **probabilidad conjunta** en lugar de una condicional:

$$\arg\max p(y|x_{1,2}, ..., x_d) = \arg\max p(y, x_{1,2}, ..., x_d)$$

De tal forma que se puede factorizar la probabilidad conjunta (asumiendo independencia de X_i's) de manera eficiente como:

$$p(y, x_{1,2}, ..., x_d) = \prod_{i=1}^{d} p(x_i|y)p(y)$$

Este tipo de modelos se conocen como **modelos generativos**.

Clasificador de Bayes ingenuo

Bayes ingenuo

El clasificador de Bayes ingenuo es un modelo gráfico dirigido y generativo, donde determina la clase a partir de maximizar la función objetivo:

$$\arg\max_y p(y, x_{1,2}, ..., x_d) = \arg\max_y \prod_{i=1}^{d} p(x_i|y)p(y)$$

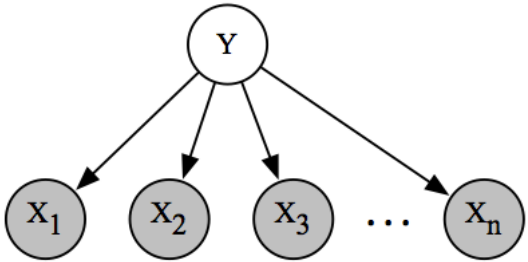
El algoritmo de Bayes ingenuo se divide en dos fases:

Fase de entrenamiento: Estima las probabilidades $p(x_i|y)$ y $p(y)$ para todos los rasgos y todas las clases a partir de los ejemplos.

Fase de evaluación: Predice las clases $\hat{y}$ en base a la función objetivo para nuevos datos. Nos permite determinar su capacidad de generalización.

Bayes como modelo gráfico

- El modelo de Bayes ingenueo es un **modelo gráfico dirigido**, representa las probabilidades conjuntas como una gráfica.
- Es **generativo**: asume que la clase Y genera a los datos visibles $X_1, ..., X_d$.
- En este sentido, asume que la clase Y es la **causa** y los observables $X_1, ..., X_d$ los efectos.



Ejemplo: Identificación de lenguas

Consideremos que tenemos la siguiente colección anotada:

- **Inglés**: 'The cat is sleeping'
- **Inglés**: 'The dog is eating'
- **Español**: 'El gato duerme'
- **Español**: 'El perro duerme'

Obtenemos los **tri-gramas** de caracteres par obtener los rasgos:

- (the, cat, is, sle, lee, epi, pin, ing, **Inglés**)
- (the, dog, is, eat, ati, tin, ing, **Inglés**)
- (el, gat, ato, due, uer, erm, rme, **Español**)
- (el, per, err, rro, due, uer, erm, rme, **Español**)

Ejemplo: Identificación de lenguas

Construimos una tabla de probabilidades para los rasgos $\mathbf{p}(\mathbf{X_i}|\mathbf{Y})$:

Trigrama	Inglés	Español
the	$\frac{2}{15}$	0
cat	$\frac{1}{15}$	0
dog	$\frac{2}{15}$	0
is	$\frac{2}{15}$	0
ing	$\frac{2}{15}$	0
el	0	$\frac{2}{15}$
gat	0	$\frac{1}{15}$
ato	0	$\frac{1}{15}$
per	0	$\frac{1}{15}$
due	0	$\frac{2}{15}$
$\vdots$	$\vdots$	$\vdots$

Asimismo obtenemos las probabilidades **a prior** $\mathbf{p}(\mathbf{Y})$:

Español	Inglés
$\frac{1}{2}$	$\frac{1}{2}$

Estas dos tablas de probabilidades conforman el **modelo de aprendizaje**.

Al estimarlas hemos concluido la fase de entrenamiento.

Ejemplo: Bayes ingenuo

Podemos probar el modelo **prediciendo** la lengua en un documento no observado:

'the cat is eating'

Del cual podemos obtener los rasgos: x = (the, cat, is, eat, ati, tin, ing) y estimar las probabilidades:

$$p(Y = ingles|x) = \frac{2}{15}\frac{1}{15}\frac{2}{15}(4 \cdot \frac{1}{15}) = \frac{16}{170859375}$$
$$p(Y = espanol|x) = 0$$

Consideraciones de Bayes ingenuo

- En muchos casos, para evitar la presencia de 0's se utiliza un estimador de probabilidad **add one** (F es el conjunto de rasgos):

$$p(x_i|y_j) = \frac{f_{i,j} + 1}{\sum_{i=1}^{|F|} f_{i,j} + |F|}$$

- Si existe un rasgo que no ha sido visto en el entrenamiento, se le asigna una probabilidad 1 o bien $\frac{1}{|F|}$.
- Para evaluar la capacidad de generalización se predice las clases para datos que no observó en el entrenamiento. Se pueden usar distintas métricas:

$$Acc = \frac{\#\text{ejemplos bien clasificados}}{\#\text{total de ejemplos}}$$

Partes de la oración

Otra forma relevante de conocimiento en los textos es su **estructura**. Al nivel básico esta estructura está dada por las llamadas partes de la oración o **Parts Of Speech (POS)**:

Partes de la oración

Las partes de la oración son categorías que determinan qué tipo de palabra y cuál es su funcionamiento dentro de una oración.

Algunas de las POS más básicas son:

POS	Función	Etiqueta
Sustantivo	Determina entidades del mundo	N
Verbo	Define acciones	V
Adjetivo	Atribuye características o rasgos a los sustantivos	A

Etiquetas EAGLES

El etiquetado POS debe responder a un **estándar**. Si bien existen varios estándares de etiquetado, uno de los más conocidos es el etiquetado EAGLES.

El etiquetado EAGLES codifica la información de las etiquetas de acuerdo a 4 criterios: **posición**, **atributo**, **valor** y **código**.

Este tipo de etiquetado indica valores **morfosintácticos**, cuando un valor no se presenta o no se determina se utilizan 0s.

Más información sobre el etiquetado en español puede consultarse en:

https://www.cs.upc.edu/~nlp/tools/parole-sp.html

Etiquetas de sustantivo

Pos.	Atributo	Valor	Código
1	Categoría	Nombre	N
2	Tipo	Común	C
		Propio	P
3	Género	Masculino	M
		Femenino	F
		Común	C
4	Número	Singular	S
		Plural	P
		Invariable	N
5	Caso	-	0
6	Género semántico	-	0
7	Grado	Apreciativo	A

Por ejemplo, la palabra:

hermanitas

Recibe la etiqueta:

NCFP00A

La palabra:

Pedro

Recibe la etiqueta:

NPMN000

Etiquetas de artículo

Pos.	Atributo	Valor	Código
1	Categoría	Artículo	T
2	Tipo	Definido	D
3	Género	Masculino	M
		Femenino	F
		Común	C
4	Número	Singular	S
		Plural	P
		Invariable	N
5	Caso	-	0

Por ejemplo 'las' recibiría la etiqueta TDFP0

Etiquetas de adjetivos

Pos.	Atributo	Valor	Código
1	Categoría	Adjetivo	A
2	Tipo	Calificativo	Q
3	Grado	Apreciativo	A
4	Género	Masculino	M
		Femenino	F
		Común	C
5	Número	Singular	S
		Plural	P
		Invariable	N
5	Caso	-	0
6	Función	Participio	P

Por ejemplo, la palabra:

rojo

Recibe la etiqueta:

AQ0MS00

La palabra:

cansaditas

Recibe la etiqueta:

AQAFP0P

Etiquetas de verbo

Pos.	Atributo	Valor	Código
1	Categoría	Verbo	V
2	Tipo	Principal	M
		Auxiliar	A
3	Modo	Indicativo	I
		Subjuntivo	S
		Imperativo	M
		Condicional	C
		Infinitivo	N
		Gerundio	G
		Participio	P
4	Tiempo	Presente	P
		Imperfecto	I
		Futuro	F
		Pasado	S

Pos.	Atributo	Valor	Código
5	Persona	Primera	1
		Segunda	2
		Tercera	3
6	Número	Singular	S
		Plural	p
7	Género	Masculino	M
		Femenino	F

Por ejemplo, la palabra 'anduvieran' recibe la etiqueta: VMSS3P0

Etiqeutado POS

El etiquetado POS consiste en asignar a cada palabra en una oración sus etiquetas correspondientes:

- (Juan, NPMN000) (se, PP3CN000) (ha, VAIP3S0) (cansado, VMP000SM)
- (el, TDMS0) (árbol, NCMS000) (cansado, AQ0MS0P)

Sin embargo, las etiquetas **dependen del contexto** en que aparecen. Por tanto, un método de clasificación usual no funciona.

Debemos utilizar una clasificación de cadenas; es decir, un método que tome en cuenta el contexto de los tokens en las cadenas.

Predicción de cadenas

Los modelos ocultos de Markov son un modelo **generativo** que busca estimar la probabilidad conjunta de dos cadenas:

$$p(y_1, y_2, ..., y_n, x_1, x_2, ..., x_n) \propto p(y_1, y_2, ..., y_n | x_1, x_2, ..., x_n)$$

Esta probabilidad puede factorizarse en una probabilidad condicional como:

$$\begin{aligned} p(y_1, y_2, ..., y_n, x_1, x_2, ..., x_n) &= p(x_1, ..., x_n | y_1, ..., y_n) p(y_1, ..., y_n) \\ &= \prod_{i=1}^{n} p(x_i|y_i) \prod_{i=2}^{n} p(y_i|y_{i-1}) p(y_1) \\ &= \prod_{i=1}^{n} p(x_i|y_i) p(y_i|y_{i-1}) \end{aligned}$$

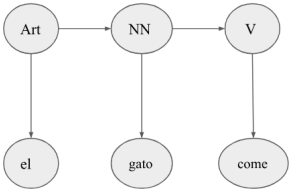
Estimación de probabilidad conjunta

Probabilidad conjunta de cadenas

Un modelo gráfico dirigido y generativo estima la probabilidad de una cadena $y_1, ..., y_n$ a partir de una cadena de entrada $x_1, ..., x_n$ (posibles vectores), a partir de maximizar la función objetivo:

$$p(y_1, y_2, ..., y_n, x_1, x_2, ..., x_n) = \prod_{i=1}^{n} p(x_i|y_i)p(y_i|y_{i-1})$$

En particular, usaremos este método de estimación de probabilidad conjunta para estimar las probabilidades de etiquetas POS.



Observaciones y emisiones

Para realizar el **modelo secuencial** de predicción de etiquetas contamos con dos lenguajes:

- **Símbolos de emisión** o etiquetas $S = \{s_1, ..., s_N\}$.
- **Observaciones** de entrada $\Sigma = \{w_1, ..., w_M\}$

Asociamos las variables de salida Y con los símbolos de emisión S y las variables de entrada X con las observaciones.

Nuestro objetivo es determinar una función $\phi : \Sigma^n \rightarrow S^n$, de tal forma que una cadena de entrada (tokens) se transforem en emisiones (etiquetas):

$$w^{(1)}...w^{(n)} \mapsto s^{(1)}...s^{(n)}$$

Factorización de probabilidades

Para estimar el modelo en base a la factorización propuesta, debemos calcular 3 tablas de probabilidades:

Probabilidades iniciales: Probabilidad de que las emisiones inicien en un símbolo s, $p(s)$ para todo $s \in S$.

Probabilidades de transición: La probabilidad de una emisión dada la emisión anterior: $p(s^{(t)}|s^{(t-1)})$.

Probabilidad de observaciones: La probabilidad de que una observación haya sido generada por una emisión en un estado t: $p(w^{(t)}|s^{(t)})$.

Podemos almacenar esta información en: un **vector de probabilidades iniciales**, una **matriz de transición** y una **matriz de probabilidades de observaciones**.

Modelos ocultos de Markov

Modelo oculto de Markov

Un modelo oculto de Markov (HMM) se define como un modelo gráfico generativo y dirigido $HMM = (S, O, A, B, \Pi)$, donde S son emisiones, O observaciones, $A = (a_{i,j}) = p(s_j|s_i)$ es matriz de transiciones, $B = (b_{i,j}) = p(w_j|s_i)$ es matriz de observaciones y $\Pi = (\pi_i) = p(s_i)$ es vector de iniciales.

La cadena de emisiones se predice a partir de maximizar la función:

$$\arg \max_{y_1,...,y_n} p(y_1, y_2, ..., y_n, x_1, x_2, ..., x_n) = \arg \max_{y_1,...,y_n} \prod_{i=1}^{n} p(x_i|y_i)p(y_i|y_{i-1})$$

Entrenamiento y evaluación de HMMs

Los HMMs son algoritmos de aprendizaje de máquina y como tales se componen de dos fases:

Fase de entrenamiento: Se cuenta con un conjunto de entrenamiento de la forma:

$$\mathcal{S} = \{(w^{(1)}...w^{(n)}, s^{(1)}...s^{(n)}) : w^{(1)}...w^{(n)} \in \Sigma^*, s^{(1)}...s^{(n)} \in S^*\}$$

A partir de este data set se estima el modelo oculto de Markov $HMM = (S, O, A, B, \Pi)$.
Los iniciales y la matriz de transición se estiman a partir de agregar símbolos BOS y EOS a la cadena de emisiones: $< BOS > s^{(1)}...s^{(n)} < EOS >$.

Fase de evaluación: Se predicen la cadena de emisiones óptima para cadenas de observaciones de un dataset de evaluación. Para esto se suele usar el **algoritmo de Viterbi**. Se evalúa el modelo.

Ejemplo: entrenamiento de HMMs

Consideremos el corpus etiquetado con las siguientes oraciones:

1. (el, TD) (gato, NC) (duerme, VM)
2. (el, TD) (gato, NC) (come, VM) (whiskas, NC)
3. (el, TD) (perro, NC) (duerme, VM)
4. (Juan, NP) (come, VM)

Nuestros vocabularios de emisiones y observaciones son:

$$\Sigma = \{el, gato, perro, Juan, come, duerme\}$$
$$S = \{TD, NC, NP, VM\}$$

Para obtener las probabilidades iniciales y de transición asumiremos los símbolos BOS y EOS.

Ejemplo: entrenamiento de HMMs

Para evitar dispersión usamos la **probabilidad corregida Add-One** ($V = \Sigma, S$):

$$p(w_j|w_i) = \frac{f_{i,j} + 1}{f_i + |V|}$$

Obtenemos así el vector de iniciales determinado como:

	TD	NC	NP	VM
BOS	$\frac{4}{8}$	$\frac{1}{8}$	$\frac{2}{8}$	$\frac{1}{8}$

Y la matriz de transiciones $\mathbf{A}$:

	TD	NC	NP	VM
TD	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{1}{6}$	$\frac{1}{9}$
NC	$\frac{4}{8}$	$\frac{1}{8}$	$\frac{1}{6}$	$\frac{2}{9}$
NP	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{1}{6}$	$\frac{1}{9}$
VM	$\frac{1}{8}$	$\frac{3}{8}$	$\frac{2}{6}$	$\frac{1}{9}$
EOS	$\frac{1}{8}$	$\frac{2}{8}$	$\frac{1}{6}$	$\frac{4}{9}$

Ejemplo: entrenamiento de HMMs

Para completar el modelo falta determinar la matriz de probabilidades de observaciones $\mathbf{B}$:

	TD	NC	NP	VM
el	$\frac{4}{9}$	$\frac{1}{9}$	$\frac{1}{7}$	$\frac{1}{10}$
gato	$\frac{1}{9}$	$\frac{3}{9}$	$\frac{1}{7}$	$\frac{1}{10}$
perro	$\frac{1}{9}$	$\frac{2}{9}$	$\frac{1}{7}$	$\frac{1}{10}$
Juan	$\frac{1}{9}$	$\frac{1}{9}$	$\frac{2}{7}$	$\frac{1}{10}$
come	$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{7}$	$\frac{3}{10}$
duerme	$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{7}$	$\frac{3}{10}$

De esta forma hemos determinado el modelo oculto de Markov $HMM = (S, O, A, B, \Pi)$.

Cálculo de probabilidades con HMMs

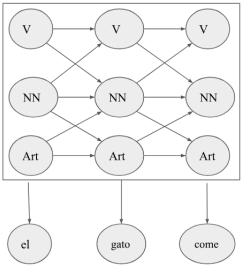
Con un modelo entrenado de HMM podemos calcular las probabilidades conjuntas de las cadenas. Por ejemplo, podemos calcular:

$$p(NP, VM, Juan, duerme) = p(NP)p(Juan|NP)p(VM|NP)p(duerme|VM)$$
$$= \frac{2}{8}\frac{2}{7}\frac{2}{6}\frac{3}{10} = \frac{24}{3360}$$

Sin embargo, para que la máquina prediga la cadena adecuada de emisiones tendría que revisar todas las posibles combinaciones.

Cálculo de probabilidades con HMMs

Los posibles caminos de emisiones que se pueden dar a partir de una observación se ilustran en un **diagrama de Treliis**.



En general, un algoritmo que determine una **cadena de emisiones** para una observación deberá revisar todos las N emisiones en cada estado, su complejidad es $\mathbf{O(N^n)}$.

Algoritmos dinámicos para estimación de emisiones

Para simplificar el problema de estimar la cadena más emisiones dada una observación se propone el uso de algorimos dinámicos. En particular dos:

- **Algoritmo de avance-retroceso**: Explora los caminos posibles de emisiones en dos partes: 1) Avance, recorre caminos de atrás hacia adelante; 2) Retroceso, recorre los caminos de adelante hacia atrás. Combina estos dos casos para estimar una etiqueta adecuada.
- **Algoritmo de Viterbi:** En cada estado, considera sólo los caminos más probables precedentes, olvidándose de esos caminos que no maximicen la probabilidad.

Procedimiento de Avance

El **procedimiento de Avance** busca determinar la probabilidad de una cadena de emisiones avanzando sobre esta.

Se define una variable que almacena las probabilidades conjuntas hasta el estado t:

$$\alpha_i(t) = p(w^{(1)} w^{(2)} \ldots w^{(t)}, s_i^{(t)})$$

De tal forma que la probabilidad de una observación está determinada como:

$$p(w^{(1)} \ldots w^{(n)}) = \sum_{i=1}^{N} \alpha_i(n)$$

Algoritmo de Avance

Algorithm Procedimiento de Avance

1: **procedure** FORWARD($w^{(1)}...w^{(n)}$, HMM)
2: $\alpha_i(0) = \pi_i,\ 1 \leq i \leq N = |S|$ — Inicialización
3: **for** t **from** 0 **to** $n-1$ **do**
4: $\alpha_i(t+1) = \sum_{j=1}^{N} p(w^{(t+1)}|s_i^{(t+1)}) p(s^{(t+1)}|s_j^{(t)}) \alpha_j(t)$ — Inducción
5: **end for**
6: $p(w^{(1)}...w^{(n)}) \leftarrow \sum_{i=1}^{N} \alpha_i(n)$ — Terminación
7: **return** $\alpha_i(t),\ t = 1, .., n,\ i = 1, ..., N$
8: **end procedure**

Procedimiento de Retroceso

El **procedimiento de Retroceso** busca determinar la probabilidad de una cadena de emisiones retrocediendo sobre esta.

Se define una variable que almacena las probabilidades conjuntas hasta el estado t:

$$\beta_j(t) = p(w^{(t+1)}w^{(t+2)}...w^{(n)}|s_j^{(t)})$$

De tal forma que la probabilidad de una observación está determinada como:

$$p(w^{(1)}...w^{(n)}) = \sum_{j=1}^{N} p(s_j)\beta_j(0)$$

Algoritmo de Retroceso

Algorithm Procedimiento de Retroceso

1: **procedure** BACKWARD($w^{(1)}...w^{(n)}$, HMM)
2: $\beta_j(n) = 1,\ 1 \leq j \leq N = |S|$ — Inicialización
3: **for** t **from** n **to** 0 **do**
4: $\beta_j(t) = \sum_{i=1}^{N} p(w^{(t)}|s_i^{(t)}) p(s_i^{(t)}|s_j^{(t)}) \beta_i(t+1)$ — Inducción
5: **end for**
6: $p(w^{(1)}...w^{(n)}) \leftarrow \sum_{j=1}^{N} \pi_j \beta_j(0)$ — Terminación
7: **return** $\beta_j(t),\ t = 1,..,n,\ j = 1,...,N$
8: **end procedure**

Procedimiento de Avance-Retroceso

Los procedimientos de avance y retroceso pueden usarse para calcular las probabilidades de cadenas de observaciones. Se pueden combinar para obtener:

$$\begin{aligned} p(w^{(1)}w^{(2)}...w^{(n)}) &= \sum_i p(w^{(1)}w^{(2)}...w^{(n)}, s_i^{(t)}) \\ &= \sum_i p(w^{(1)}...w^{(t)}, s_i^{(t)}) p(w^{(t+1)}...w^{(n)}|s_i^{(t)}) \\ &= \sum_i \alpha_i(t)\beta_i(t) \end{aligned}$$

De aquí, podemos obtener un resultado importante:

$$p(w^{(1)}w^{(2)}...w^{(n)}, s_i^{(t)}) = \alpha_i(t)\beta_i(t)$$

Predicción de etiqueta con avance retroceso

De esta forma, dada una cadena de observaciones $w^{(1)}w^{(2)}...w^{(n)}$ podemos estimar la etiqueta de la observación $t = 1, ..., n$ de la siguiente forma:

$$\begin{aligned}\hat{s}^{(t)} &= \arg\max_i p(s_i^{(t)}|w^{(1)}w^{(2)}...w^{(n)}) \\ &= \arg\max_i \frac{p(s_i^{(t)}, w^{(1)}w^{(2)}...w^{(n)})}{p(w^{(1)}w^{(2)}...w^{(n)})} \\ &= \arg\max_i \frac{\alpha_i(t)\beta_i(t)}{\sum_i \alpha_i(t)\beta_i(t)}\end{aligned}$$

Y ya que $p(s_i^{(t)}|w^{(1)}w^{(2)}...w^{(n)}) \propto p(s_i^{(t)}, w^{(1)}w^{(2)}...w^{(n)})$:

$$\hat{s}^{(t)} = \arg\max_i p(s_i^{(t)}, w^{(1)}w^{(2)}...w^{(n)})$$

Algoritmo de avance retroceso

Algorithm Procedimiento de Avance-Retroceso

1: **procedure** FORWARD-BACKWARD($w^{(1)}...w^{(n)}$, HMM)
2: $\alpha_i(t) \leftarrow$ FORWARD($w^{(1)}...w^{(n)}$, HMM)
3: $\beta_i(t) \leftarrow$ BACKWARD($w^{(1)}...w^{(n)}$, HMM)
4: **for** t **from** 1 **to** n **do**
5: $\hat{s}^{(t)} \leftarrow \arg\max_i \alpha_i(t)\beta_i(t)$
6: **end for**
7: **return** $\hat{s}^{(1)}\hat{s}^{(2)}...\hat{s}^{(n)}$
8: **end procedure**

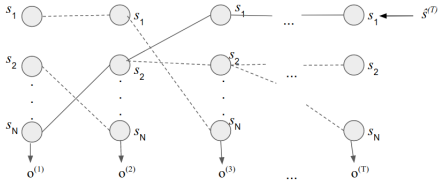
Algoritmo de Viterbi

El algoritmo de **avance-retroceso** tiene algunas **desventajas**:

- En cada iteración sólo determina el argumento que maximiza una etiqueta.
- Las etiquetas en cada estado no tienen memoria de las etiquetas tomadas en estados anteriores.

Algoritmo de Viterbi

El algoritmo de Viterbo es un algoritmo dinámico que encuentra el camino (dentro del diagrama de trellis) que maximiza la probabilidad conjunta de emisiones cuando se tiene una cadena de observaciones.



Algoritmo de Viterbo

El algoritmo de Viterbi busca **maximizar** la probabilidad de una secuencia de emisiones, observando los elementos **previos que maximizan** la probabilidad actual:

$$\max_{i_1,...i_{t-1}} p(s_{i_1}^{(1)},..,s_{i_{t-1}}^{(t-1)},s_j^{(t)},w^{(1)}...w^{(n)}) = \max_{i_1,...i_{t-1}} p(w^{(1)}...w^{(n)}|s_{i_1}^{(1)},..,s_{i_{t-1}}^{(t-1)},s_j^{(t)})$$
$$p(s_{i_1}^{(1)},..,s_{i_{t-1}}^{(t-1)},s_j^{(t)})$$
$$= \max_{i_1,...i_{t-1}} \prod_{t=1}^{t-1} p(w^{(t)}|s_{i_t}^{(t)})p(s_{i_t}^{(t)}|s_{i_{t-1}}^{(t-1)})$$

Este **camino máximo** hacia el símbolo s_j en el estado t se almacena en una variable:

$$\delta_j(t) = \max_{i_1,...i_{t-1}} p(s_{i_1}^{(1)},..,s_{i_{t-1}}^{(t-1)},s_j^{(t)},w^{(1)}...w^{(n)})$$

Algoritmo de Viterbi

Algorithm Algoritmo de Viterbi

procedure VITERBI($w^{(1)}...w^{(n)}$, HMM)
  $\delta_j(1) = p(w^{(1)}|s_j)\pi_j,\ 1 \leq j \leq N = |S|$
  for t **from** 1 **to** $n-1$ **do**
    $\delta_j(t+1) = \max_i p(w^{(t+1)}|s_j)p(s_j|s_i)\delta_i(t)$
    $\phi_j(t+1) = \arg\max_i p(w^{(t+1)}|s_j)p(s_j|s_i)\delta_i(t)$
  end for
  $\hat{s}^{(n)} \leftarrow \arg\max_j \delta_j(n)$
  for t **from** $n-1$ **to** 1 **do**
    $\hat{s}^t \leftarrow \phi_{\hat{s}^{(t+1)}}(t+1)$
  end for
  return $\hat{s}^{(1)}\hat{s}^{(2)}...\hat{s}^{(n)}$
end procedure

Ejemplo

	TD	NC	VM
TD	$\frac{1}{7}$	$\frac{1}{7}$	$\frac{1}{8}$
NC	$\frac{4}{7}$	$\frac{1}{7}$	$\frac{2}{8}$
VM	$\frac{1}{7}$	$\frac{3}{7}$	$\frac{1}{8}$
EOS	$\frac{1}{7}$	$\frac{2}{7}$	$\frac{4}{8}$

	TD	NC	VM
el	$\frac{4}{8}$	$\frac{1}{8}$	$\frac{1}{9}$
gato	$\frac{1}{8}$	$\frac{3}{8}$	$\frac{1}{9}$
perro	$\frac{1}{8}$	$\frac{2}{8}$	$\frac{1}{9}$
come	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{3}{9}$
duerme	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{3}{9}$

	TD	NC	VM
BOS	$\frac{4}{6}$	$\frac{1}{6}$	$\frac{1}{6}$

Etiquetar la oración “el gato come”:

- **Inicialización:**
  $\delta_{TD}(1) = \frac{16}{48}, \delta_{NC}(1) = \frac{1}{48}, \delta_{VM}(1) = \frac{1}{54}$
- **Inducción:** $\delta_{TD}(2) = \frac{16}{2688} - TD, \delta_{NC}(2) = \frac{192}{2688} - TD, \delta_{VM}(2) = \frac{16}{3024} - TD$
  $\delta_{TD}(2) = \frac{192}{150528} - NC, \delta_{NC}(2) = \frac{192}{150528} - NC, \delta_{VM}(2) = \frac{1728}{193536} - NC$
- **Retroceso:** $\hat{s}^3 = VM, \hat{s}^2 = NC, \hat{s}^1 = TD$

Reconocimiento de entidades nombradas

Entidad nombrada

Una entidad nombrada es un token (o conjunto de tokens) que refieren a un objeto de la realidad, tales como una persona, un lugar, una organización, un producto, una fecha, etc.

Las entidades nombradas son importantes para identificar los **temas** de un documento, así como **eventos** que se describen en este.

NER

El reconocimiento de entidades nombradas (Named Entity Recognition o NER) refiere a la tarea de identificar, dentro de un texto, a todas las entidades nombradas que lo componen.

El reconocimiento de entidades nombradas será útil para realizar búsquedas y recuperación de información relevante en textos.

Etiquetado de entidades nombradas

Algunas de las etiquetas que se pueden encontrar dentro del NER son:

Etiqueta	Entidad
PER	Nombres de Personas
LOC	Lugares
ORG	Organización
MISC	Otras entidades

Por ejemplo, en el siguiente texto:

> *Operaciones realizadas durante el Proceso de Reorganización Nacional, período en el cual la deuda creció durante los dos gobiernos sucesivos de Carlos Menem.*

Tenemos dos entidades nombradas que determinan un evento.

Etiquetado NER

Para realizar el etiquetado generalmente se utilizan **etiquetas BIO**. Estas etiquetas indican:

BIO	Uso
B-tag	Inicio de la categoría 'tag'
I-tag	Interior de la categoría 'tag'
O	Token que no pertenece a la categoría

Por ejemplo, un etiquetado sería:

(Operaciones, O) (realizadas, O) (durante, O) (el, O) (Proceso, B-ORG) (de, I-ORG) (Reorganización, I-ORG) (Nacional, I-ORG)

Con base en este etiquetado se peude aplicar el **algoritmo de HMM**.

Textos recomendados

Shannon, C. E. (1948). *A mathematical theory of communication*. Bell Systems Technical Journal 27(3), pp. 379-423.

Ben-Naim, A. (2007). *La entropia desvelada. El mito de la segunda ley de la termondinámica y el sentido común*. Tusquets editores.

Kotlerman, L., Dagan, I., Szpector, I. y Zhitomirsky-Geffet, M.(2010). 'Directional distributional similarity for lexical inference'. *Natural Language Enigneering*, 16(4), pp. 359-389.

Franzi, K., Ananiadou, S. y Mima, H. (2000). "Automatic recognition of multi-word terms: the C-value/NC-value method". *Natural language processing for digital libraries*, pp. 115-130.

Manning, C. y Schütze, H. (1999). "9. Markov Models"; "10. Part-of-Speech Tagging". *Foundations of Statistical Natural Language Processing*, pp. 317-380.