Lingüística Computacional

Víctor Mijangos de la Cruz

III. Perspectivas estadísticas

Métodos de segmentación no supervisada

Problemáticas de las perspectivas formales

Las perspectivas formales son una primera aproximación a procesar computacionalmente el lenguaje natural; pero presentan ciertas **desventajas**:

- Requieren de un **conocimiento previo** y profundo de la lengua, que muchas veces no es fácilmente accesible.
- Son **dependientes del lenguaje**: un modelo está pensado para aplicarse únicamente a una lengua.
- Las reglas son **rígidas** y no me permiten incertidumbre con respecto a los resultados.
- El conjunto de reglas que describe el lenguaje natural es **demasiado amplio** y no es fácilmente abarcable.

Perspectivas estadísticas

Ante las dificultades que presentan los métodos formales, la lingüística computacional ha adoptado perspectivas **estadísticas**.

Estas tienen las ventajas de:

- Pueden **integrar** aspectos de las perspectivas formales en sus modelos.
- Pueden lidiar con **incertidumbre** del lenguaje y manejar casos de forma menos rígida.
- No requieren de un conocimiento profundo ni de planteamiento de reglas pues pueden realizarse de forma **no supervisada**.

Análisis morfológico estadístico

Análisis morfológico estadístico

El análisis morfológico estadístico hace referencia a los modelos de análisis morfológico utilizando métodos basados en estadísticos. Dos aplicaciones básicas son:

- Etiquetado morfológico (parseo): Identifica unidades y les asigna una etiqueta. Se basa en aprendizaje supervisado.
- Segmentación morfológica: Identifica unidades morfológicas de forma no supervisada.

Revisaremos un método de **segmentación morfológica** estadística basada en métodos de **codificación**. Este método es llamado **Byte-Pair Encodign** (BPE).

Codificación

Una parte importante de la transmisión de mensajes es la **compresión**; ésta permite enviar un código más pequeño con la misma información de un código mayor.

El código depende del alfabeto Σ. En lenguaje natural, este es el alfabeto latino. Sin embargo, existen patrones que permite "comprimir" las cadenas.

La **eficacia** del código puede medirse en base a la **longitud esperada**, L:

$$L = \sum_{i=i}^{n} p(x_i) l_i$$

Donde x_i es un objeto a codificar (un token), $p(x_i)$ es su probabilidad y l_i su longitud de código.

Codificación óptima

Es claro que todo código debe asignar una longitud **mayor a 0** a todo elemento que codifica. Idealmente también el código debe ser **fácilmente decodificable**.

Para asegurar estas dos propiedades se suele recurrir a la **desigualdad de Kraft**, basada en la longitud del alfabeto $N = |\Sigma|$:

$$\sum_{i=1}^{n} N^{l_i} \leq 1$$

Minimizar la longitud esperada sujeta a la desigualdad de Kraft nos da como resultado que la **longitud esperada óptima** es:

$$H = \sum_{i=1}^{n} p(x_i) \log_N \frac{1}{p(x_i)}$$

A un codificación con longitud $\log_N \frac{1}{p(x_i)}$ se le conoce como de **Shannon-Fano**.

Compresión de codificación

Una forma en que podemos minimizar la longitud esperada de un código es **comprimiendo**. Considérese el alfabeto $\Sigma = \{0, 1\}$ y el lenguaje formado por las siguientes cadenas:

- 0111
- 0100
- 0101
- 0110

En este lenguaje toda cadena comienza por el prefijo 01, por tanto, podemos comprimir al definir un nuevo bit $01 \mapsto A$ de tal forma que:

$$L = \{A11, A00, AA, A10\}$$

Las cadenas se han reducido, pero el alfabeto ahora es $\Sigma = \{0, 1, A\}$.

Byte Pair Encoding

A este método de compresión se le conoce como **byte pair encoding**.

En el lenguaje natural, los morfemas presentan patrones regulares. Sin embargo no es fácil determinarlos.

Podemos, en su lugar, hablar de **subwords**, entendidos como patrones (subcadenas) recurrentes.

La idea es buscar patrones con **poca información** (ergo muy frecuentes) que puedan agruparse; una aproximación a los morfemas.

Byte Pair Encoding

Por ejemplo, tomemos las siguientes palabras y sus frecuencias:

Palabra	Frecuencia
n i ñ a s	5
h o j a s	4
n i ñ o s	2
g a t o s	6

Los espacios en blancos entre las letras indican los símbolos que componen la cadena. Asumimos que están formadas por el alfabeto $\Sigma = \{a, b, ..., x, y, z\}$.

Byte Pair Encoding

Para encontrar patrones recurrentes, hacemos **pares** y los asociamos a sus frecuencias:

$$[n, i] : 7, [i, ñ] : 7, [ñ.o] : 5, [a, s] : 9, [h, o] : 4, ...$$

Buscamos el par con más frecuencia y lo reescribimos como un nuevo símbolo:

$$\arg\max\{[a, b]\} \mapsto ab$$

En este caso, tenemos $[a, s] \mapsto as$
Se sustituye este símbolo cada vez que se observe el par y se repite el proceso.

Palabra	Frecuencia
n i ñ as	5
h o j as	4
n i ñ o s	2
g a t o s	6

Byte Pair Encoding

En una siguiente iteración, debemos agregar *as* al alfabeto y formar nuevos pares considerando esto:

$$[\text{ñ}, as] : 5, [j, as] : 4, ...$$

En esta segunda iteración, el par más frecuente es $[o, s]$, lo reescribimos como el nuevo símbolo *os*, agregamos al alfabeto y sustituimos cada ves que vemos el par:

Palabra	**Frecuencia**
n i ñ as	5
h o j as	4
n i ñ os	2
g a t os	6

Los símbolos coinciden con sufijos. El procesos se repite *k* veces.

Algoritmo de BPE

Algorithm Algoritmo de BPE

1: **procedure** BPE(word_list, Σ, k)
2: **Inicialización**: Se indican los símbolos del alfabeto Σ presentes en las palabras. Generalmente por medio de espacios. Cada carácter será un símbolo.
3: **for** 1 **to** k iteraciones **do**
4: $[a_i, b_i] : f([a_i, b_i])$ — Obtener frecuencias
5: $[a, b] = \arg\max_{a_i, b_j}\{f([a_i, b_j]) : a_i, b_j \in \Sigma\}$ — Símbolo de mayor frecuencia
6: $ab \leftarrow [a, b]$ **and** $\Sigma \leftarrow \Sigma \cup \{ab\}$ — Crear nuevo símbolo y añadir a alfabeto
7: word_list $\leftarrow$ REPLACE(ab) — Remplazar el nuevo símbolo
8: **end for**
9: **returns** word_list, Σ
10: **end procedure**

Byte Pair Encoding

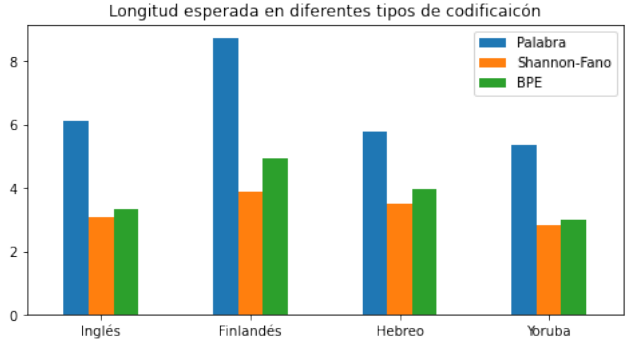
El algoritmo de BPE busca reemplazar los símbolos que tienen mayor frecuencia. Lo que equivale a reemplazar los símbolos con menor información:

$$\begin{aligned}\arg\max_{a_i,b_j}\{f([a_i,b_j]) : a_i,b_j \in \Sigma\} &= \arg\max_{a_i,b_j}\{p([a_i,b_j]) : a_i,b_j \in \Sigma\} \\ &= \arg\min_{a_i,b_j}\{\log \frac{1}{p([a_i,b_j])} : a_i,b_j \in \Sigma\}\end{aligned}$$

Los símbolos con menor información son fácilmente decodificables, por tanto su compresión afecta poco al mensaje.

Longitud esperada y BPE

La codificación que encuentra el algoritmo de BPE se acerca más al óptimo establecido por la codificación de Shannon-Fano. Si bien su optimilidad depende de varios parámetros, se puede ver que es una mejor codificación (en términos de longitud esperada) que la codificación por caracteres (véase Pimentel et al (2021)).



Ley de Zipf y leyes empíricas del lenguaje

Modelos estadísticos

Las **frecuencias** juegan un papel importante en las descripciones estadísticas del lenguaje.

Las frecuencias nos dan información acerca de la **distribución de los tokens**, de la cual podemos **estimar probabilidades**.

Existen diferente **leyes empíricas del lenguaje** que nos dicen cómo se comporta una lengua a nivele stadístico.

A partir de estas leyes y otras técnicas estadísticas podemos crear un **modelo del lenguaje**; es decir, asignar probabilidades a unidades lingüísticas.

Frecuencias

Dentro de un texto, las palabras aparecen organizadas según una estructura del lenguaje. Esta estructura se muestra en las regularidades con que cada palabra aparece. Por ejemplo:

> *Humanismo es un concepto polisémico que se aplica tanto al estudio de las letras humanas, los estudios clásicos y la filología grecorromana como a una genérica doctrina o actitud vital que concibe de forma integrada los valores humanos. Por otro lado, también se denomina humanismo al «sistema de creencias centrado en el principio de que las necesidades de la sensibilidad y de la inteligencia humana pueden satisfacerse sin tener que aceptar la existencia de Dios y la predicación de las religiones», lo que se aproxima al laicismo o a posturas secularistas. Se aplica como denominación a distintas corrientes filosóficas, aunque de forma particular, al humanismo renacentista1 (la corriente cultural europea desarrollada de forma paralela al Renacimiento a partir de sus orígenes en la Italia del siglo XV), caracterizado a la vez por su vocación filológica clásica y por su antropocentrismo frente al teocentrismo medieval.*

Frecuencias

Este pequeño texto cuenta con **156 tokens** y **102 tipos**. Podríamos pensar que cada palabra tiene una aparición de **1.5** veces en el texto.

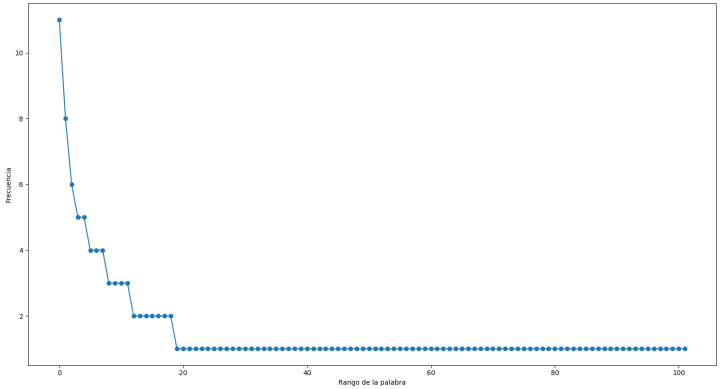
Sin embargo, la distribución de las palabras no es uniforme. Hay palabras que tienen una mayor frecuencia. Las primeras cinco palabras más frecuentes son:

1. "'de"': 11
2. "'la"': 8
3. "'al"': 6
4. "'que"': 5
5. "'a"': 5

Las palabras con mayor frecuencia son palabras funcionales.

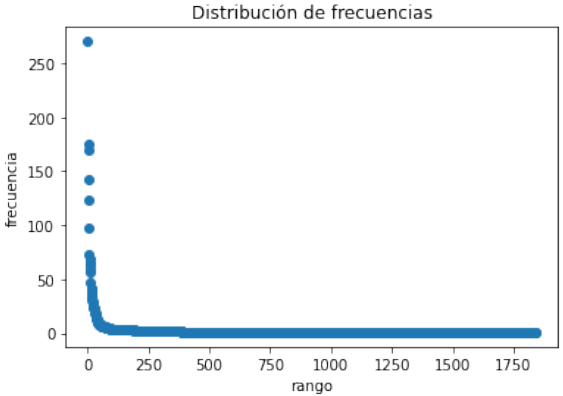
Frecuencias

Si se grafica las palabras a aprtir de su frecuencia (rango) se observa:



Frecuencias

Con corpus más grandes, se tiene una tendencia mas clara:



Corpus con 4562 tokens y 1840 tipos

Ley de Zipf

En el corpus CREA (Real Academia Española) las 3 palabras más frecuentes son:

1. "'de"', fr: 9999518
2. "'la"', fr: 6277560
3. "'que"', fr: 4681839

Se observa una tendencia, la segunda palabra más frecuente tiene casi la mitad de la primera y la tercera, casi la tercera parte.

En base a esto, George K. Zipf (1949) propone la **Ley de Zipf**:

$$fr \propto \frac{1}{r^{\alpha}}$$

Donde *fr* es la frecuencia de la palabra y *r* es su rango. $\alpha \in \mathbb{R}$ es un parámetro.

Estimación de probabilidad

A partir de la Ley de Zipf se pueden estimar las probabilidades de las palabras.

Defínase una variable aleatoria con relación al rango:

$$(X \leq r)$$

Es decir, la variable X puede tomar los valores desde 1 hasta el número de tipos t.

La medida de probabilidad sobre está variable estará dada por:

$$p(X = r; \alpha, t) = \frac{1/r^{\alpha}}{\zeta(\alpha, t)}$$

donde:

$$\zeta(\alpha, t) := \sum_{i=1}^{t} \frac{1}{r_i^{\alpha}}$$

Estimación de probabilidad

Estimar la probabilidad de una palabra con el rango r, entonces depende de los parámetros α y t. Por ejemplo:

- Si $\alpha = 0.69$
- Y si $t = 100000$

Entonces, la probabilidad de $r = 5$ es:

$$\begin{aligned} p(X = 5; 0.69, 100000) &= \frac{1/5^{0.69}}{\zeta(0.69, 100)} \\ &\approx \frac{0.32}{111.784} \\ &\approx 0.002 \end{aligned}$$

Empíricamente, t es el número de tipos, pero ¿cómo estimamos α?.

Estimación de α

Para estimar el parámetro α podemos utilizar un método como la entropía cruzada (o la máxima verosimilitud):

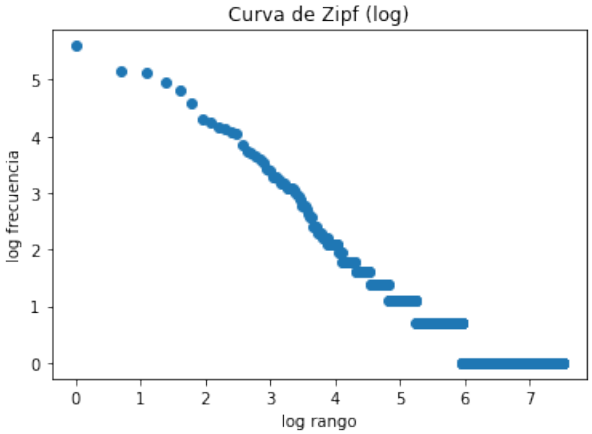
$$\hat{\alpha} = \arg\min_{\alpha} -\frac{1}{t}\sum_{r=1}^{t} \log p(X = r; \alpha)$$

$$= \arg\min_{\alpha} -\frac{1}{t}\sum_{r=1}^{t} \log \frac{1/r^{\alpha}}{\zeta(\alpha, t)}$$

De forma más sencilla, podemos observar que α, al ser un exponente, puede **linearizarse** por medio del logaritmo.

Escala logarítmica en Zipf

En escala logarítmica, la curva de Zipf se asemeja a una recta con pendiente negativa:



Linearización con logaritmo

Ya que $fr(w_r) \propto \frac{1}{r^\alpha}$ implica que existe un β tal que $fr(w_r) = \frac{\beta}{r^\alpha}$, tenemos que:

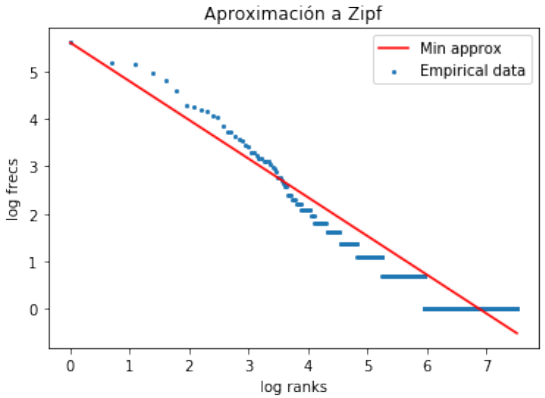
$$\begin{aligned} \log fr(w_r) &= \log \frac{\beta}{r^\alpha} \\ &= \log \beta - \log r^\alpha \\ &= \log \beta - \alpha \log r \end{aligned}$$

Para encontrar el $\hat{\alpha}$ que mejor se ajuste a los datos, podemos usar **mínimos cuadrados**:

$$\hat{\alpha} = \arg\min_\alpha \frac{1}{2} \sum_{r=1}^{t} (\log fr(w_r) + \alpha \log r - \log \beta)^2$$

$\log \beta$ es un sesgo (o traslación) que corresponde al logaritmo de la frecuencia de la palabra con mayor rango; esto es $\beta = fr(w_1)$

Aproximación de α



Aproximación con mínimos cuadrados, $\alpha = 0.81$

Otras leyes empíricas del lenguaje

Cuando el corpus es suficientemente grande, se ha observado que $\alpha \to 1$. Es común ver la ley de Zipf enunciada como:

$$fr \propto \frac{1}{r}$$

Lo que deja de lado la estimación de α.

Aproximación frecuentista

Si X es una variable aleatoria tal que $X \sim Zipf(\alpha, t)$ (X tiene una distribución de Zipf con parámetros α y t), donde t es el número de tipos y $\alpha \in \mathbb{R}$. Entonces:

$$p(X = r; \alpha, t) \approx \frac{fr(r)}{\sum_{r'} fr(r')} \quad (1)$$

Donde $fr(r)$ es la frecuencia de la palabra con rango r.

Distribución de Mandelbrot

Existen otras formas de aproximar la distribución que presenta el lenguaje humano Mandelbrot (2000):

Fórmula de Mandelbrot

Sea r el rango de una palabra en un corpus dado y sea fr su frecuencia dentro de este corpus. Entonces:

$$fr \propto \frac{P}{(r+\rho)^{\alpha}}$$

Donde α es una constante, y P, ρ son parámetros del texto.

Función de Menzerath-Altman

Otros autores han propuesto otro tipo de fórmulas para aproximar la distribución de palabras en el lenguaje (Altmann y Gerlach, 2016):

Función de Menzerath-Altman

Sea r el rango de una palabra en un corpus dado y sea fr su frecuencia dentro de este corpus. Entonces:

$$fr \propto r^b \cdot e^{-\alpha/r}$$

Comparación de distribuciones

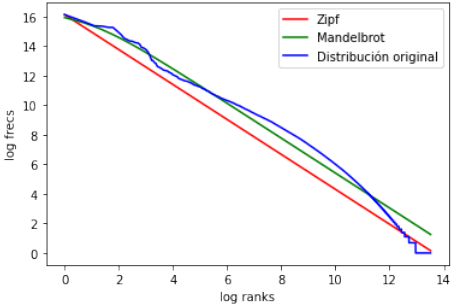
Las leyes empíricas determinan diferentes formas de **estimar las probabilidades** de los tokens en un corpus.

Para determinar que estimación es más adecuada a los datos se puede hacer uso de una **evaluación**. Tomamos en cuenta:

- La **longitud esperada** (L): Una distribución que asigna probabilidades según la longitud de código, minimizando la longitud esperada.
- La **entropía** (H): Una distribución que minimice la entropía o la longitud en base a código de Shannon-Fano.
- La **Perplejidad** (Px): Una distribución que minimice la perplejidad, es decir el valor 2^H.

Comparación de distribuciones

Podemos comparar la distribución frecuentista, Zipf y Mandelbrot para el corpus CREA. Utilizamos las medidas de evaluación H, Px y L.



Estimador	H	Px	L
Frecuentista	1.845	3.59	4.9
Zipf	**1.519**	**2.86**	**4.02**
Mandelbrot	1.803	3.48	4.67

Ley de Herdan

Otra ley empírica importante para el lenguaje es la **ley de Herdan** (o ley de Heap) (Herdan, 1960).

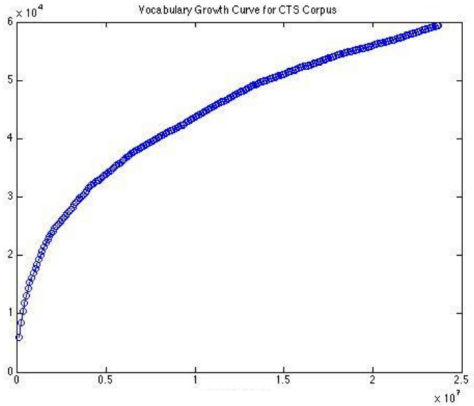
Esta ley observa que:

- En un corpus, el número de tipos es siempre menos (o igual) al número de tokens.
- El número de tipos aumenta más lento que el de tokens.

Se formula la relación entre número de tokens N y número de tipos t:

$$t \propto N^{\frac{1}{\alpha}}$$

Ley de Herdan



Ley de Herdan en el corpus Conversational Telephone Speech (CTS)

Textos recomendados

Sentich, R., Haddow, B. y Birch, A. (2015). "Neural machine translation of rare words with subword units". https://arxiv.org/abs/1508.07909

Pimentel, T., Nikkarinen, I., Mahowald, K., Cotterell, R. y Blasi, D. (2021). "How (non-) optimal is the lexicon?". arXivpreprintarXiv:2104.14279

Zipf, G. (1949). *Human behavior and the principle of least effort*.

Mandelbrot, B. (2000). *Los objetos fractales*. Tusquets.

Altman, G. y Gerlach, M. (2016). "Statistical laws in linguistics". *Creativity and universality in language* Springer, pp. 7–26.

Herdan, G. (1960). *Type-token mathematics*. Mouton ed.