

# Procesamiento de Texto y Modelo Vectorial

Felipe Bravo Márquez

6 de noviembre de 2013



- ¿Cómo recupera un buscador como Google o Yahoo! documentos relevantes a partir de una consulta enviada?
- ¿Cómo puede procesar una empresa los reclamos que le dejan sus usuarios en sus portales Web?
- ¿Cómo podemos agrupar los comentarios emitidos en un foro y analizar las opiniones de la gente?

Para resolver esos problemas, hay que estudiar las siguientes áreas del conocimiento:

- *Recuperación de Información*: Ciencia encargada de la búsqueda de información en documentos.
- *Text Mining*: Extracción de conocimiento a partir del texto.



# Tokens y Tipos

Dado un documento  $d$ , se llama como *tokenización* a la tarea de separar el texto por palabras llamados *tokens*, además se pueden borrar caracteres especiales, como la puntuación y convertir los caracteres a minúsculas [Manning et al., 2008].

## Ejemplo

Input: Además de inscribir Web Mining, inscribí Data Mining.

Tokens: [además] [de] [inscribir] [web] [mining] [inscribí] [data] [mining]

Se define como un *tipo* como una clase de *token* que contiene una única secuencia de caracteres. Se obtienen identificando los tokens iguales dentro del documento.

## Tipos

Tipos: [además] [data] [de] [inscribí] [inscribir] [mining] [web]

*El token mining estaba repetido*



# Extracción del Vocabulario de términos [1]

Un *término* es un *tipo* normalizado. Se llama vocabulario  $V$ , al conjunto de términos de una colección de documentos  $D$ .

## Borrado de Stopwords

Para reducir la dimensión del vocabulario y eliminar términos que no aportan información, se eliminan los términos que aparecen con mucha frecuencia en la mayoría de los documentos (stopwords). Como artículos, pronombres, preposiciones y conjunciones.

Ejemplo: [el, la, ellos, ellas, nosotros, un, una, de, con, a, además, ya, y, muy, otro, cuando, cuanto].

## Stemming

Proceso donde se transforman los términos a su raíz para reducir la dimensión del vocabulario. Se realiza en base a un conjunto de reglas de reducción de palabras. Ejemplo: Algoritmo de Porter.

(F)	Rule		Example
	SSES	→ SS	caresses → caress
	IES	→ I	ponies → poni
	SS	→ SS	caress → caress
	S	→	cats → cat



## Lematización

- Es otra estrategia para llevar las palabras a su raíz. Realiza un análisis morfológico por medio de diccionarios de referencia para crear clases de equivalencia entre *tipos*.
- Por ejemplo para el token *saw*, una regla de stemming podría construir un término *s*, mientras que mediante lematización un diccionario nos entregaría *see*.

Eliminando las stopwords y haciendo uso de stemming, el vocabulario del documento  $d$  queda de la siguiente forma:

termId	value
t1	data
t2	inscrib
t3	mining
t4	web



- La ley de Zipf, propuesta por *George Kingsley Zipf* en [Zipf, 1935], se usa para el análisis de frecuencia de aparición de términos dentro de una colección de documentos.
- Dice que la frecuencia  $f$  de aparición de un término en una colección es inversamente proporcional a su ranking  $r$  en una tabla ordenada de frecuencias.

$$f = \frac{cf}{r^\beta} \quad (1)$$

- Donde  $cf$  es una constante dependiente de la colección y  $\beta > 0$  modela la razón de decaimiento.
- Si  $\beta = 1$ , entonces  $f$  sigue exactamente la ley de Zipf, si no se dice que sigue una distribución Zipf-like. A mayor  $\beta$  menor es la calidad del lenguaje en los documentos.
- La ley se relaciona con el principio de mínimo esfuerzo. Usamos muchas veces unas pocas palabras para escribir las ideas.



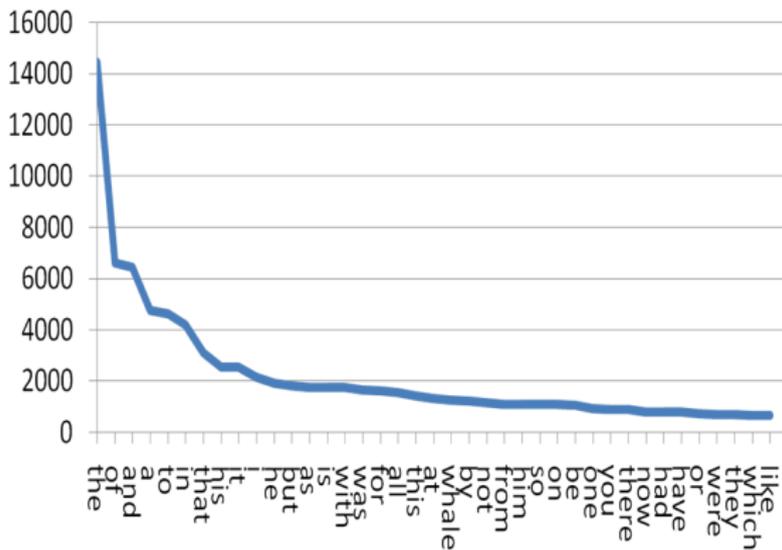


Figura: Ley de Zipf

- Si se realiza un gráfico  $\log - \log$ , se obtiene una recta de pendiente  $-\beta^{-1}$ .
- Los términos más frecuentes se pueden usar para crear la lista de *stopwords*.

# Lista de Posteo e Índice Invertido

Sea  $D$  una colección de documentos y  $V$  el vocabulario de todos los términos extraídos de la colección:

- La lista de posteo de un término, es la lista de todos los documentos en los que aparece al menos una vez.
- Un índice invertido en una estructura de datos de diccionario que mapea cada término  $t_i \in V$  a su lista de posteo.

$\langle term \rangle \rightarrow \langle docId \rangle^*$

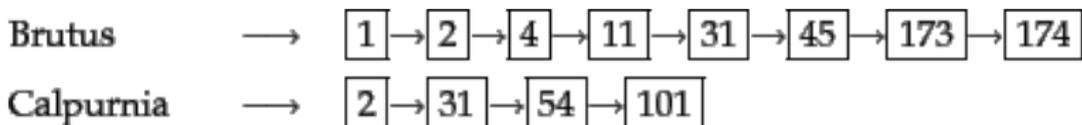


Figura: Índice Invertido

# Motor de Búsqueda [1]

Un motor de búsqueda es un sistema de recuperación de información diseñado para la búsqueda de información en la Web [Gutiérrez et al., 2008]. Sus componentes básicos son:

- Crawler: Un robot que navega la Web según una estrategia definida. Generalmente comienza navegando por un conjunto de páginas semilla (seeds) y continua navegando por sus hipervínculos.
- Indexador: Encargado de mantener un índice invertido con el contenido de las páginas recorridas por el Crawler.
- Máquina de consultas: Encargado de procesar las consultas y buscar en el índice los documentos con mayor similitud a ella.
- Función de ranking: Es la función que tiene la máquina de consulta para rankear los documentos indexados en la colección por relevancia para una consulta.
- Interfaz: Interactúa con el usuario, recibe la consulta como entrada y retorna los documentos rankeados por similitud.



# Motor de Búsqueda [2]

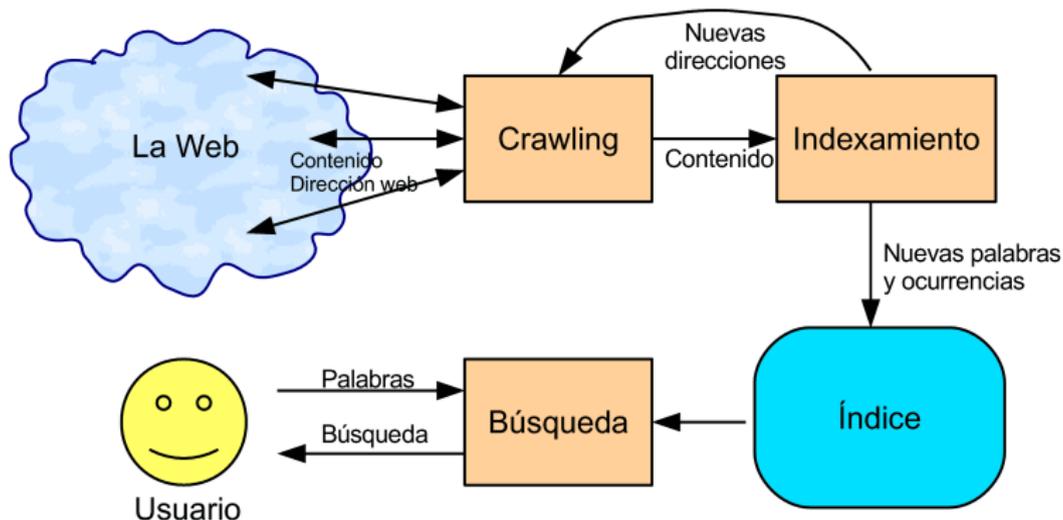


Figura: Diagrama Motor de Búsqueda [Gutiérrez et al., 2008].

- Para poder rankear las consultas, o medir la similitud entre dos documentos necesitamos una métrica de similitud.
- Representamos los documentos como vectores de términos, donde cada término es una dimensión.
- A este tipos de modelos se le llame *Bag of Words*. Perdemos el orden de las palabras.
- El valor de cada dimensión, es un peso que representa la relevancia del término  $t_i$  en el documento  $d$ .

$$d_j \rightarrow \vec{d}_j = (w(t_1, d_j), \dots, w(t_{|V|}, d_j)) \quad (2)$$

- ¿Cómo podemos modelar el aporte de información de un término en un documento?



# Term Frequency - Inverted Document Frequency [1]

- Se define  $Tf_{i,j}$ , como la frecuencia del término  $t_i$  en el documento  $d_j$ .
- Un término que aparece 10 veces debiese aportar mayor información que uno aparece una vez.
- ¿Qué pasa cuando tenemos documentos muchos más largos que otros?
- Podemos normalizar por la frecuencia máxima de término en el documento.

$$Tf_{i,j} = \frac{f_{i,j}}{\max f_{i,j}}$$

- ¿Un término que aparece en muy pocos documentos aporta más o menos información que uno que aparece varias veces?
- Por ejemplo, el documento *El señor alcalde de Malloco*. El término *Malloco* aparece en menos documentos que *alcalde*, por lo que debiese ser más descriptivo.



# Term Frequency - Inverted Document Frequency [2]

- Sea  $Q$  el número de documentos en la colección y  $n_i$  el número de documentos donde aparece el término  $t_i$ , se define el  $idf$  del término  $t_i$  como:

$$idf_{t_i} = \log_{10}\left(\frac{Q}{n_i}\right)$$

- Un término que aparece en todos los documentos tendría  $idf = 0$  y uno que aparece en el 10% de la colección tendría  $idf = 1$ .
- El modelo de score  $Tf - idf$  combina ambos modelos, quedando el peso  $w$  de un término sobre un documento como:

$$w(t_i, d_j) = Tf_i \times \log_{10}\left(\frac{Q}{n_i}\right)$$

- Las consultas a un motor de búsqueda también pueden modelarse como vectores, pero las consultas tienen en promedio entre 2 y 3 términos. Para evitar tener tantas dimensiones nulas, se usa un factor de suavizamiento en el vector:

$$w(t_i, d_j) = (0,5 + 0,5 \times Tf_{i,j}) \log_{10}\left(\frac{Q}{n_i}\right)$$



# Similitud entre Vectores

- Una vez representados, los documentos y consultas como vectores, podemos medir su similitud.
- Una alternativa sería usar la distancia euclidiana, pero la variabilidad de largo entre documentos afectaría a la métrica.
- Lo más usado es usar el coseno del ángulo entre los vectores como medida de similitud.
- Si los documentos son iguales, el ángulo vale 0 y el coseno 1. En cambio si son ortogonales el coseno vale 0.
- Los vectores, deben ser normalizados por su norma euclidiana  $\|d\|_2$ , la similitud de calcula de la siguiente manera:

$$\cos(d_1, d_2) = \frac{d_1 \cdot d_2}{|d_1| \times |d_2|} = \frac{\sum_{i=1}^{|V|} (w(t_i, d_1) \times w(t_i, d_2))}{\sqrt{\sum_{i=1}^{|V|} w(t_i, d_1)^2} \times \sqrt{\sum_{i=1}^{|V|} w(t_i, d_2)^2}}$$

- Erróneamente se llama *distancia coseno*, realmente es una medida de similitud.



# Similitud Coseno

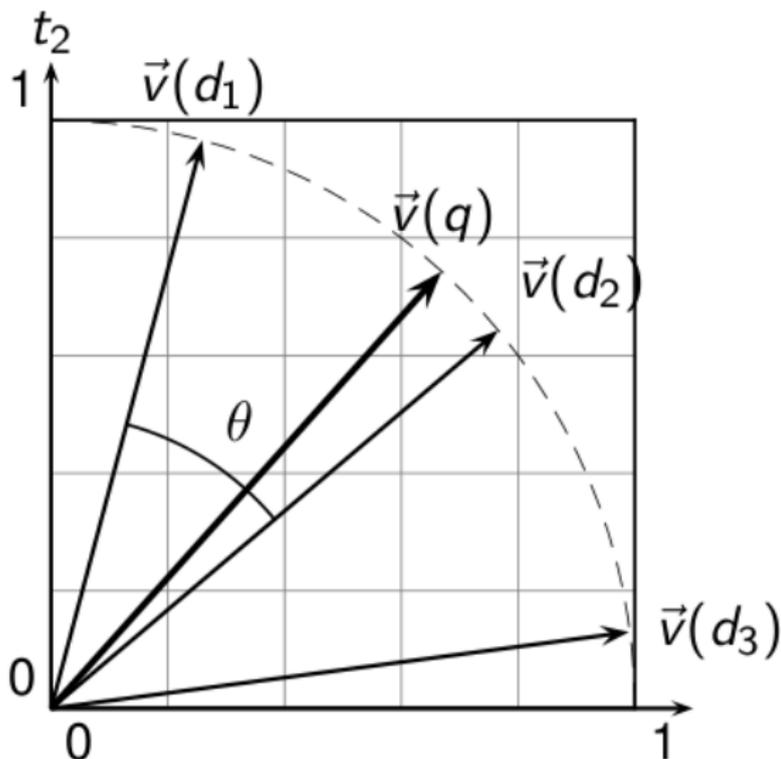


Figura: Similitud coseno.

- Supongamos que tenemos 3 documentos, los cuales se forman a partir de las siguientes secuencias de términos:  
 $d_1 \rightarrow t_4 t_3 t_1 t_4$   
 $d_2 \rightarrow t_5 t_4 t_2 t_3 t_5$   
 $d_3 \rightarrow t_2 t_1 t_4 t_4$
- Construya una matriz término-documento de dimensión  $5 \times 3$  usando los pesos  $Tf - idf$  simples (sin normalización).
- Le recomendamos construir primero una lista con la cantidad de documentos en los que aparece cada término (para el  $idf$ )
- Calcule luego el  $idf$  de cada término.
- Llene las celdas con los valores  $Tf - idf$
- ¿ A qué documento está más cercano  $d_1$  ?



Cuadro: Matriz Tf-idf

	d1	d2	d3
t1	0.176	0.000	0.176
t2	0.000	0.176	0.176
t3	0.176	0.176	0.000
t4	0.000	0.000	0.000
t5	0.000	0.954	0.000

# Clustering de Documentos [1]

- ¿Qué pasa si queremos agrupar los documentos de contenidos similares?
- Agrupamos los documentos en conjuntos, donde todos los elementos sean similares entre sí.
- A cada conjunto se le llama *cluster*.
- El problema de clusterizar, se basa en identificar grupos que maximicen la similitud interna dentro de un cluster y minimicen la similitud entre documentos pertenecientes a distintos clusters [Velasquez and Palade, 2008].

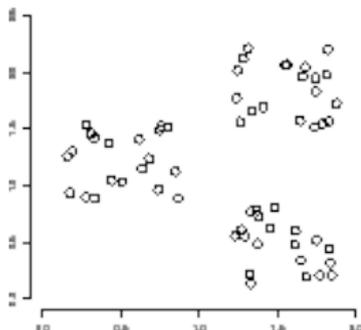


Figura: Conjunto de documentos donde se identifica claramente cada cluster

# Clustering de Documentos [2]

- Permite identificar grupos de opiniones similares, o reducir el espacio de búsqueda para una consulta en un buscador.
- K-medias es un algoritmo simple de clustering que requiere la cantidad  $k$  de clusters a construir como parámetro.
  - 1 Primero, se identifican aleatoriamente  $k$  elementos. Los valores de los atributos de éstos elementos se copian en nuevos elementos llamados centroides de la misma dimensión que éstos. Cada centroeide representará un cluster.
  - 2 Luego se calcula la distancia de todos los  $n$  elementos a los  $k$  centroides y se asigna cada elemento al cluster del centroeide más cercano.
  - 3 Luego se recalcula el valor de los centroides promediando el valor de los atributos de todos los elementos pertenecientes al cluster.
  - 4 Repite el proceso de calcular las distancias, agrupar los más cercanos y recalcular los centroides hasta que éstos dejen de cambiar.



```
K-MEANS( $\{\vec{x}_1, \dots, \vec{x}_N\}, K$ )
1   $(\vec{s}_1, \vec{s}_2, \dots, \vec{s}_K) \leftarrow \text{SELECTRANDOMSEEDS}(\{\vec{x}_1, \dots, \vec{x}_N\}, K)$ 
2  for  $k \leftarrow 1$  to  $K$ 
3  do  $\vec{\mu}_k \leftarrow \vec{s}_k$ 
4  while stopping criterion has not been met
5  do for  $k \leftarrow 1$  to  $K$ 
6      do  $\omega_k \leftarrow \{\}$ 
7      for  $n \leftarrow 1$  to  $N$ 
8          do  $j \leftarrow \arg \min_j |\vec{\mu}_j - \vec{x}_n|$ 
9               $\omega_j \leftarrow \omega_j \cup \{\vec{x}_n\}$  (reassignment of vectors)
10     for  $k \leftarrow 1$  to  $K$ 
11         do  $\vec{\mu}_k \leftarrow \frac{1}{|\omega_k|} \sum_{\vec{x} \in \omega_k} \vec{x}$  (recomputation of centroids)
12 return  $\{\vec{\mu}_1, \dots, \vec{\mu}_K\}$ 
```

Figura: Algoritmo K-medias

# References I

-  Gutiérrez, C., Navarro, G., Baeza-Yates, R., Hurtado, C., Arenas, M., Marín, M., Piquer, J. M., Rodríguez, M., del Solar, J. R., and Velasco, J. (2008). *Cómo funciona la Web*. Autoeditada.
-  Manning, C. D., Raghavan, P., and Schütze, H. (2008). *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA.
-  Velasquez, J. D. and Palade, V. (2008). *Adaptive Web Sites: A Knowledge Extraction from Web Data Approach*.
-  Zipf, G. K. (1935). *The Psychobiology of Language*. Houghton-Mifflin, New York, NY, USA.