

Semántica distribuida

Curso de procesamiento de lenguaje natural

Julio Waissman

Maestría en Tecnologías de la Información
UNaM/UNEE

9 y 10 de agosto de 2018



Similaridad entre palabras

- Dos palabras (tokens) son más similares si comparten más características de significado
- Es una relación entre los sentidos de las palabras
- Es posible calcular la similaridad entre palabras con un **thesaurus**
- Funciona bien con sustantivos, pero no con adjetivos o verbos
- Es difícil y en muchas ocasiones el contexto es diferente

Se puede conocer una palabra por su compañía

- Estudiar las palabras por su contexto (en una ventana de tamaño fijo)
- Calcular las coocurrencias en el corpus de las palabras u y v dentro de la misma ventana n_{uv}
- Calcular la información mutua palabra a palabra (PMI, por *Pointwise Mutual Information*)

$$PMI = \log \frac{\Pr(u, v)}{\Pr(u) \Pr(v)} = \log \frac{n_{uv} n}{n_u n_v}$$

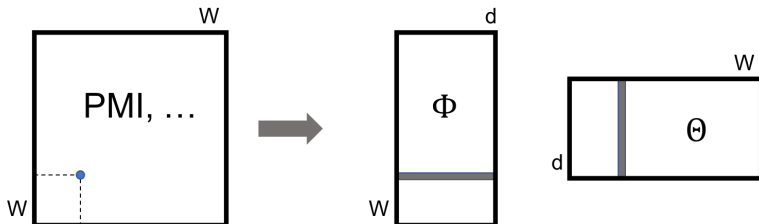
donde n es el número de palabras del corpus, n_u y n_v son el número de veces que aparecen las palabras u y v respectivamente.

- Usar solo la parte positiva (pPMI)

$$pPMI = \max(0, PMI)$$

Semantica distribuida por métodos lineales (SVD)

- Se basa en una matriz de similaridad entre palabras (PMI, cuentas, ...)
- Reduccion de la dimensionalidad por SVD
- Resultado: Una matriz Φ de dimensión $W \times d$ tal que el renglón i de la matriz es un vector de dimensión d que codifica la palabra w_i del vocabulario



Semantica distribuida por optimización (GloVe)

En lugar de estimar directamente la matriz Φ , se infiere del problema de optimización

$$\max_{\Phi, \Theta, b, b'} \sum_{u \in V} \sum_{v \in V} f(n_{uv})(\phi_u^T \theta_v + b_u + b'_v - \log n_{uv})^2$$

donde $f : [0, \infty] \rightarrow [0, 1]$ es una función de saturación

- La función a optimizar es cuadrática y existen métodos bien desarrollados para solucionarlo.
- No deja de ser computacionalmente costoso (en tiempo y memoria)
- Solamente se basa en la matriz de coocurrencias
- Uno de los primeros métodos desarrollados mas allá de la descomposición matricial en valores singulares.

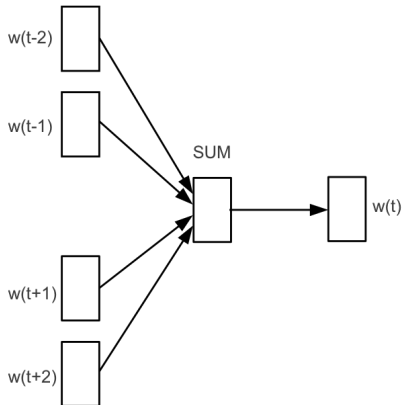
- Basados en la clasificación correcta de palabras en lugar de matrices de similitud
- Se basa en el contexto sobre una ventana deslizante
- Dos estrategias básicas
 - 1 *Continuous BOW (CBOW)*. Se basa en encontrar la probabilidad de una palabra, dado su contexto

$$\Pr(w_i | w_{i-n}, \dots, w_{i-1}, w_{i+1}, \dots, w_{i+n})$$

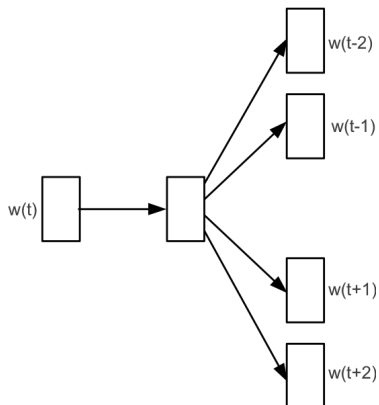
- 2 *Skip-gram*. Se trata de encontrar el contexto de una palabra, si conociéramos la palabra

$$\Pr(w_{i-n}, \dots, w_{i-1}, w_{i+1}, \dots, w_{i+n} | w_i)$$

CBOW y Skip-gram

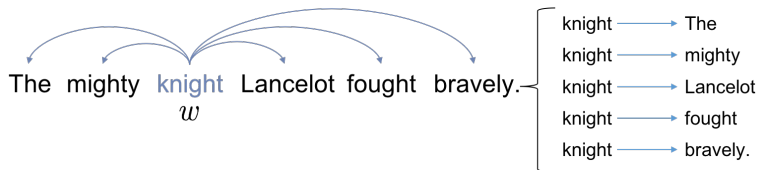


CBOW



Skip-gram

Skip-gram

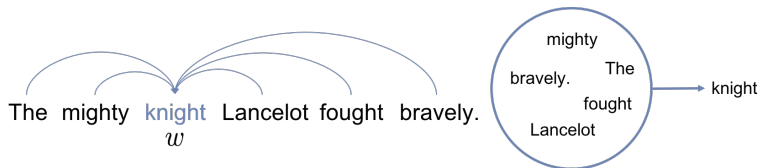


$$\Pr(c|w) = \frac{\exp(\phi_w^T \theta_c)}{\sum_{v \in V} \exp(\phi_w^T \theta_v)}, \quad \phi_w, \theta_v \in \mathbb{R}^d$$

Si tenemos un corpus (w_1, w_2, \dots, w_T) entonces Φ se obtiene de resolver

$$\min_{\Phi, \Theta} - \sum_{t=1}^T \sum_{c \in \mathcal{C}_t} \log \frac{\exp(\phi_{w_t}^T \theta_c)}{\sum_{v \in V} \exp(\phi_{w_t}^T \theta_v)}$$

donde \mathcal{C}_t son las palabras de contexto de w_t .



$$\Pr(c|w) = \frac{\exp(h_w^T \theta_w)}{\sum_{v \in V} \exp(h_w^T \theta_v)}, \quad h_w, \theta_v \in \mathbb{R}^d$$

donde

$$h_w = \sum_{c \in \mathcal{C}_w} \phi_c$$



$$\Pr(c|w) = \frac{\exp(h_w^T \theta_c)}{\sum_{v \in V} \exp(h_w^T \theta_v)}, \quad h_w, \theta_v \in \mathbb{R}^d$$

donde

$$h_w = \sum_{g \in w} x_g, \quad x_g \in \mathbb{R}^d$$

es la suma de n-gramas de caracteres

FastText permite estimar vectores para palabras OOV

Similitud entre palabras

- Dado un par de palabras, encontrar su similitud.
- La similitud de las palabras no depende de la distancia entre sus vectores, si no de **el ángulo entre vectores**.
- Similitud coseno:

$$s(w, v) = \frac{\phi_w^T \phi_v}{\|\phi_w\|_2 \|\phi_v\|_2}$$

- Funciona bien para lenguajes morfológicamente ricos (como el español)

- Sea a y a' dos palabras, tenemos la palabra b y queremos encontrar b' que tenga una relación con b análoga a la que tienen a y a' , por ejemplo

hombre es a rey lo que mujer es a ?

- En forma de vectores de palabras se podría expresar como:

$$\phi_a - \phi_{a'} \approx \phi_b - \phi_{b'},$$

y por lo tanto,

$$\phi_{b'} \approx \phi_{a'} - \phi_a + \phi_b$$

- Tomando en cuenta la similaridad coseno

$$b' = \arg \max_{x \notin \{a, a', b\}} \cos(b - a + a', x)$$

- Funciona relativamente bien para analogías sintácticas

Codificación de sentencias o textos

- Existen métodos para entrenar sentencias.
- En la práctica, se prueba primero generar un vector por sentencia

Sea $s = (w_1, \dots, w_T)$ una *sentencia* o *frase*, la cual está compuesta de una serie de palabras o tokens. Entonces, la sentencia se puede codificar como:

$$\phi_s = \frac{1}{T} \sum_{t=1}^T \phi_{w_t}$$

- Nada más se toman en cuenta los tokens que existen en el vocabulario
- Suele funcionar extrañamente bien

Veamos unos ejemplos

