Módulo II: Modelos de Lenguaje y Etiquetado de Secuencia

Dr. Gaddiel Desirena López

Procesamiento de Lenguaje Natural (NLP) 2021

I.- Vectores Palabra (Word Embeddings)

Vectores Palabra

- Un componente importante en las redes neuronales para el lenguaje es el uso de una capa embebida.
- Un mapeo de símbolos discretos a vectores continuos.
- Al embeber una palabra, se transforman de símbolos distintos aislados en símbolos matemáticos objetos sobre los que se puede operar.
- La distancia entre vectores se puede equiparar a la distancia entre palabras.
- Esto facilita la generalización del comportamiento de una palabra a otra.

Vectores de distribución

- **Hipótesis distributiva** [Harris, 1954]:las palabras que aparecen en el mismo **contextos** tienden a tener significados similares.
- **Representaciones distributivas**: las palabras están representadas por **vectores de alta dimensión** según el contexto donde ocurren.

Matrices de contexto de palabras

- Los vectores de distribución se construyen a partir de matrices de contexto de palabras M.
- Cada celda (i, j) es un valor de asociación basado en la co-ocurrencia entre una **palabra de destino** w_i y una **contexto** c_j calculada a partir de un corpus de documentos.
- Los contextos se definen comúnmente como ventanas de palabras que rodean w_i.
- La longitud de la ventana k es un parámetro (entre 1 y 8 palabras en los lados izquierdo y derecho de w_i).
- Si el vocabulario de las palabras de destino y las palabras de contexto es el mismo, M tiene dimensionalidad $|\mathcal{V}| \times |\mathcal{V}|$.
- Mientras que es probable que las ventanas más cortas capturen **información sintáctica** (por ejemplo, POS), es más probable que las ventanas más largas capturen similitudes temáticas [Goldberg, 2016, Jurafsky and Martin, 2008].

Distributional Vectors with context windows of size 1

Example corpus:

- I like deep learning.
- I like NLP.
- I enjoy flying.

counts	I	like	enjoy	deep	learning	NLP	flying	.
I	0	2	1	0	0	0	0	0
like	2	0	0	1	0	1	0	0
enjoy	1	0	0	0	0	0	1	0
deep	0	1	0	0	1	0	0	0
learning	0	0	0	1	0	0	0	1
NLP	0	1	0	0	0	0	0	1
flying	0	0	1	0	0	0	0	1
.	0	0	0	0	1	1	1	0

[0]Ejemplo tomado de:
http://cs224d.stanford.edu/lectures/CS224d-Lecture2.pdf

Matrices de contexto de palabras

Las asociaciones entre palabras y contextos se pueden calcular utilizando diferentes enfoques:

1. Recuentos de co-ocurrencia.
2. Información mutua puntual positiva (PPMI).

El más común de todos según [Jurafsky and Martin, 2008] es PPMI.

Los métodos de distribución también se denominan métodos basados en recuento.

PPMI

- PMI calcula el logaritmo de la probabilidad de que los pares palabra-contexto ocurran juntos sobre la probabilidad de que sean independientes.

$$\mathrm{PMI}(w,c) = \log_2 \left(\frac{P(w,c)}{P(w)P(c)} \right) = \log_2 \left(\frac{\mathrm{count}(w,c) \times |D|}{\mathrm{count}(w) \times \mathrm{count}(c)} \right) \quad (1)$$

- Los valores negativos del PMI sugieren que el par coexiste con menos frecuencia que el azar.
- Estas estimaciones no son fiables a menos que los recuentos se calculen a partir de un corpus muy grande [Jurafsky and Martin, 2008].
- PPMI corrige este problema reemplazando los valores negativos por cero:

$$\mathrm{PPMI}(w,c) = \max(0, \mathrm{PMI}(w,c)) \quad (2)$$

Vectores distribuidos o Wordembeddings

- Los vectores de distribución basados en conteo aumentan de tamaño con el vocabulario, es decir, pueden tener una dimensionalidad muy alta.
- El almacenamiento explícito de la matriz de co-ocurrencia puede consumir mucha memoria.
- Algunos modelos de clasificación no se adaptan bien a datos de gran dimensión.
- La comunidad de redes neuronales prefiere usar **representaciones distribuidas** [1] o **wordembeddings**.
- Los Word**embeddings** son vectores de palabras densas continuas de baja dimensión entrenados a partir de corpus de documentos usando **redes neuronales**.
- Las dimensiones no se pueden interpretar directamente, es decir, representan características latentes de la palabra, " con suerte capturando propiedades sintácticas y semánticas útiles " cite turian2010word.
- Se han convertido en un componente crucial de las arquitecturas de redes neuronales para la PNL.

[1] Idea: El significado de la palabra está " distribuido " sobre una combinación de dimensiones.

Vectores distribuidos o Wordembeddings (2)

- Hay dos enfoques principales para obtener incrustaciones de palabras:
  1. capas embebidas: uso de una capa embebida en una arquitectura de red neuronal específica de la tarea entrenada a partir de ejemplos etiquetados (por ejemplo, análisis de sentimientos).
  2. Pre-trained wordembeddings: crear una tarea predictiva auxiliar a partir de un corpus sin etiquetar (por ejemplo, predecir la siguiente palabra) en la que los wordembeddings surgirán naturalmente de la arquitectura de la red neuronal.
- Estos enfoques también se pueden combinar: se puede inicializar una capa embebida de una red neuronal específica de la tarea con los wordembeddings previamente entrenadas obtenidas con el segundo enfoque.

Vectores distribuidos o Wordembeddings (2)

- Los modelos más populares basados en el segundo enfoque son skip-gram [Mikolov et al., 2013], bolsa de palabras continua [Mikolov et al., 2013] y Glove [Pennington et al., 2014].
- Los Wordembeddings han demostrado ser más poderosas que los enfoques distributivos en muchas tareas de PNL [Baroni et al., 2014].
- En [Amir et al., 2015], se usaron como **características** en un modelo de regresión para determinar la asociación entre palabras de Twitter y **sentimientos positivos**.

Word2Vec

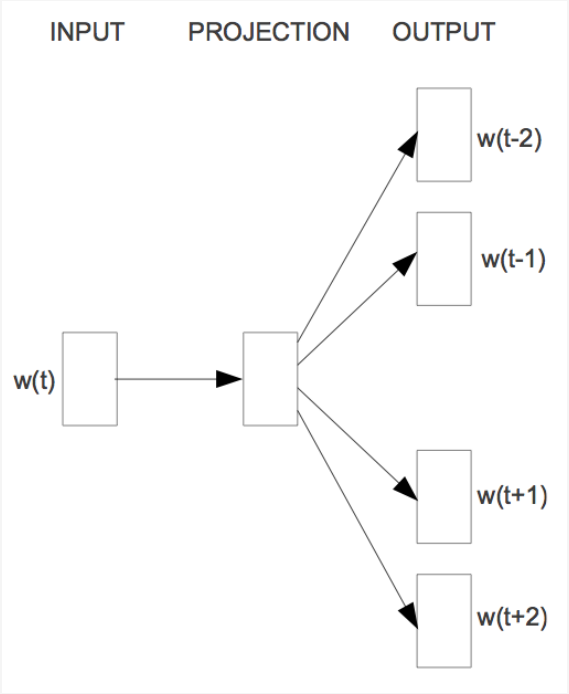
- Word2Vec es un paquete de software que implementa dos arquitecturas de redes neuronales para entrenar wordembeddings: Continuous Bag of Words (CBOW) y Skip-gram.
- Implementa dos modelos de optimización: Muestreo Negativo y Softmax Jerárquico.
- Estos modelos son redes neuronales superficiales que están capacitadas para predecir los contextos de las palabras.
- Un tutorial muy completo sobre los algoritmos de word2vec: `https://arxiv.org/pdf/1411.2738.pdf`.

Modelo Skip-gram

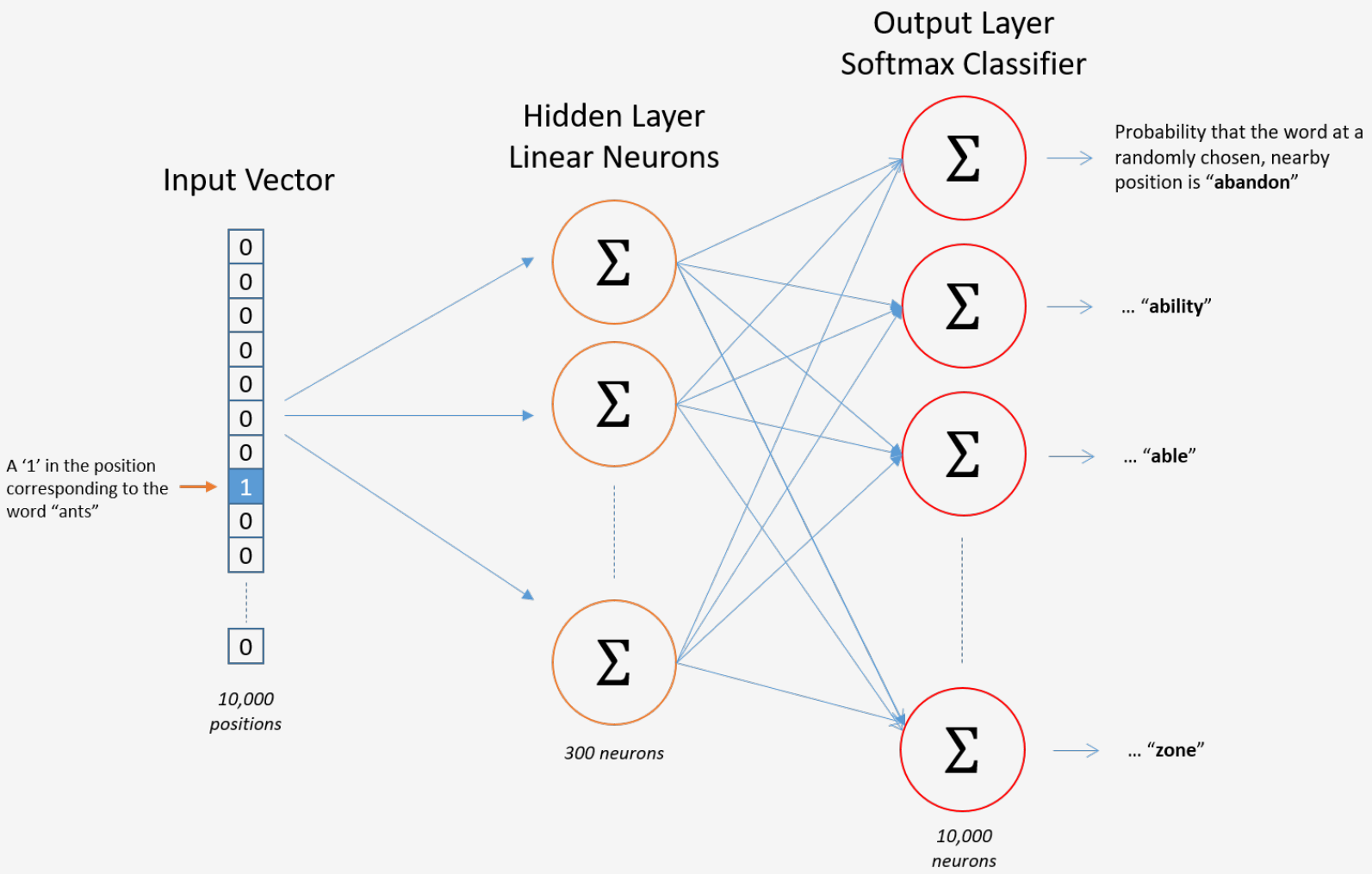
- Una red neuronal con una capa de “ proyección ” u “ oculta ” está entrenada para predecir las palabras que rodean una palabra central, dentro de una ventana de tamaño k que se desplaza a lo largo del corpus de entrada.
- El centro y las palabras k circundantes corresponden a las capas de entrada y salida de la red.
- Las palabras se representan inicialmente mediante vectores 1-hot: vectores del tamaño del vocabulario ($|V|$) con valores cero en todas las entradas excepto el índice de palabras correspondiente que recibe un valor de 1.

Modelo Skip-gram

- La capa de salida combina los vectores k 1-hot de las palabras circundantes.
- La capa oculta tiene una dimensionalidad d, que determina el tamaño de las incrustaciones (normalmente $d\ ll|V|$).



Modelo Skip-gram



Parametrización del modelo Skip-gram

- Se nos da un corpus de entrada formado por una secuencia de palabras $w_1, w_2, w_3, ..., w_T$ y un tamaño de ventana k.
- Denotamos las palabras objetivo o (centrales) con la letra w y las palabras de contexto circundantes con la letra c.
- La ventana de contexto $c_{1:k}$ de la palabra w_t corresponde a las palabras $w_{tk/2}, \ldots, w_{t-1}, w_{t+1}, \ldots, w_{t+k/2}$ (asumiendo que k es un número par).

Parametrización del modelo Skip-gram

- El objetivo del modelo Skip-gram es maximizar la probabilidad logarítmica promedio de las palabras de contexto dadas las palabras objetivo:

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{c\in c_{1:k}}\log P(c|w_t)$$

- La probabilidad condicional de una palabra de contexto c dada una palabra central w se modela con un softmax (C es el conjunto de todas las palabras de contexto, que suele ser el mismo que el vocabulario):

$$P(c|w) = \frac{e^{\vec{c}\cdot\vec{w}}}{\sum_{c'\in C} e^{\vec{c'}\cdot\vec{w}}}$$

- Parámetros del modelo θ: $\vec{c}$ y $\vec{w}$ (representaciones vectoriales de contextos y palabras objetivo).

Parametrización del modelo Skip-gram

- Sea D el conjunto de pares correctos palabra-contexto (es decir, pares de palabras que se observan en el Corpus).
- El objetivo de la optimización es maximizar la probabilidad logarítmica condicional de los contextos c (esto es equivalente a minimizar la pérdida de entropía cruzada):

$$\arg\max_{\vec{c},\vec{w}} \sum_{(w,c)\in D} \log P(c|w) = \sum_{(w,c)\in D} (\log e^{\vec{c}\cdot\vec{w}} - \log \sum_{c'\in C} e^{\vec{c'}\cdot\vec{w}}) \quad (3)$$

- : Suposición: maximizar esta función resultará en buenos wordembeddings $\vec{w}$ es decir, palabras similares tendrán vectores similares.
- El término $P(c|w)$ es computacionalmente costoso debido a la suma $\sum_{c'\in C} e^{\vec{c'}\cdot\vec{w}}$ sobre todos los contextos c'.
- Fix: reemplace el softmax con un softmax jerárquico (el vocabulario se representa con un árbol binario de Huffman).
- Los árboles de Huffman asignan códigos binarios cortos a palabras frecuentes, lo que reduce el número de unidades de salida a evaluar.

Skip-gram con muestreo Negativo

- El muestreo negativo (NS) se presenta como un modelo más eficiente para calcular los skip-gram embeddings.
- Sin embargo, optimiza una función objetivo diferente [Goldberg and Levy, 2014].
- NS maximiza la probabilidad de que un par palabra-contexto (w, c) provenga del conjunto de pares correctos palabra-contexto D usando una función sigmoidea:

$$P(D = 1|w, c_i) = \frac{1}{1 + e^{-\vec{w} \cdot \vec{c_i}}}$$

- Supuesto: las palabras de contexto c_i son independientes entre sí:

$$P(D = 1|w, c_{1:k}) = \prod_{i=1}^{k} P(D = 1|w, c_i) = \prod_{i=1}^{k} \frac{1}{1 + e^{-\vec{w} \cdot \vec{c_i}}}$$

- Esto conduce a la siguiente función objetivo (log-likelihood):

$$\arg\max_{\vec{c}, \vec{w}} \quad \log P(D = 1|w, c_{1:k}) = \sum_{i=1}^{k} \log \frac{1}{1 + e^{-\vec{w} \cdot \vec{c_i}}} \tag{4}$$

Skip-gram con muestreo Negativo (2)

- Este objetivo tiene una solución trivial si establecemos $\vec{w}$, $\vec{c}$ tal que $P(D = 1|w, c) = 1$ para cada par (w, c) desde D.
- Esto se logra configurando $\vec{w} = \vec{c}$ y $\vec{w} \cdot \vec{c} = K$ para todos $\vec{w}, \vec{c}$, donde K es un número grande.
- Necesitamos un mecanismo que evite que todos los vectores tengan el mismo valor, al no permitir algunas combinaciones (w, c).
- Una forma de hacerlo es presentar el modelo con algunos (w, c) pares para los cuales $P(D = 1|w, c)$ debe ser bajo, es decir pares que no están en los datos.
- Esto se logra tomando muestras negativas de $tildeD$.

Skip-gram con muestreo Negativo (3)

- Ejemplo de m palabras para cada par palabra-contexto $(w, c) \in D$.
- Agrega cada palabra muestreada w_i junto con el contexto original c como un ejemplo negativo a $\tilde{D}$.
- Funció objetivo final:

$$\arg\max_{\vec{c},\vec{w}} \sum_{(w,c)\in D} \log P(D=1|w, c_{1:k}) + \sum_{(w,c)\in \tilde{D}} \log P(D=0|w, c_{1:k}) \quad (5)$$

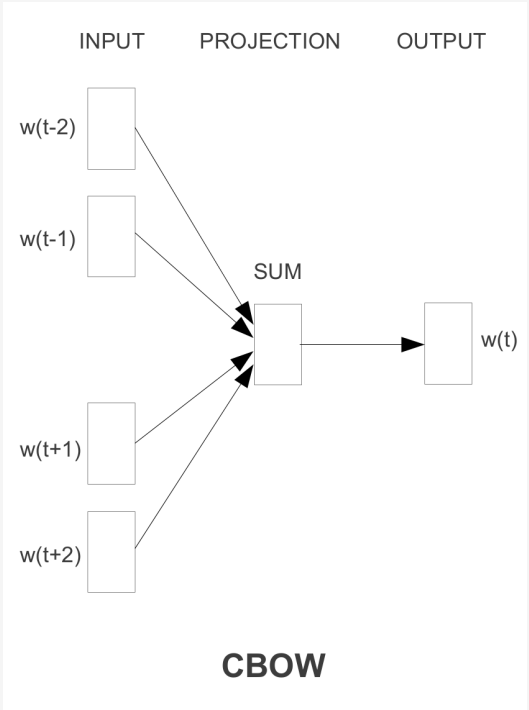
- Las palabras negativas se extraen de la versión suavizada de las frecuencias del corpus:

$$\frac{\#(w)^{0.75}}{\sum_{w'} \#(w')^{0.75}}$$

- Esto le da más peso relativo a las palabras menos frecuentes.

Continuous Bag of Words: CBOW

- Similar al modelo skip-gram , pero ahora la palabra central se predice a partir del contexto circundante.



GloVe

- GloVe (de vectores globales) es otro método popular para entrenar incrustaciones de palabras [Pennington et al., 2014].
- Construye un contexto de palabras explícito matriz, y entrena los vectores de palabra y contexto $\vec{w}$ y $\vec{c}$ intentando satisfacer:

$$\vec{w} \cdot \vec{c} + b_{[w]} + b_{[c]} = \log \#(w, c) \quad \forall (w, c) \in D \tag{6}$$

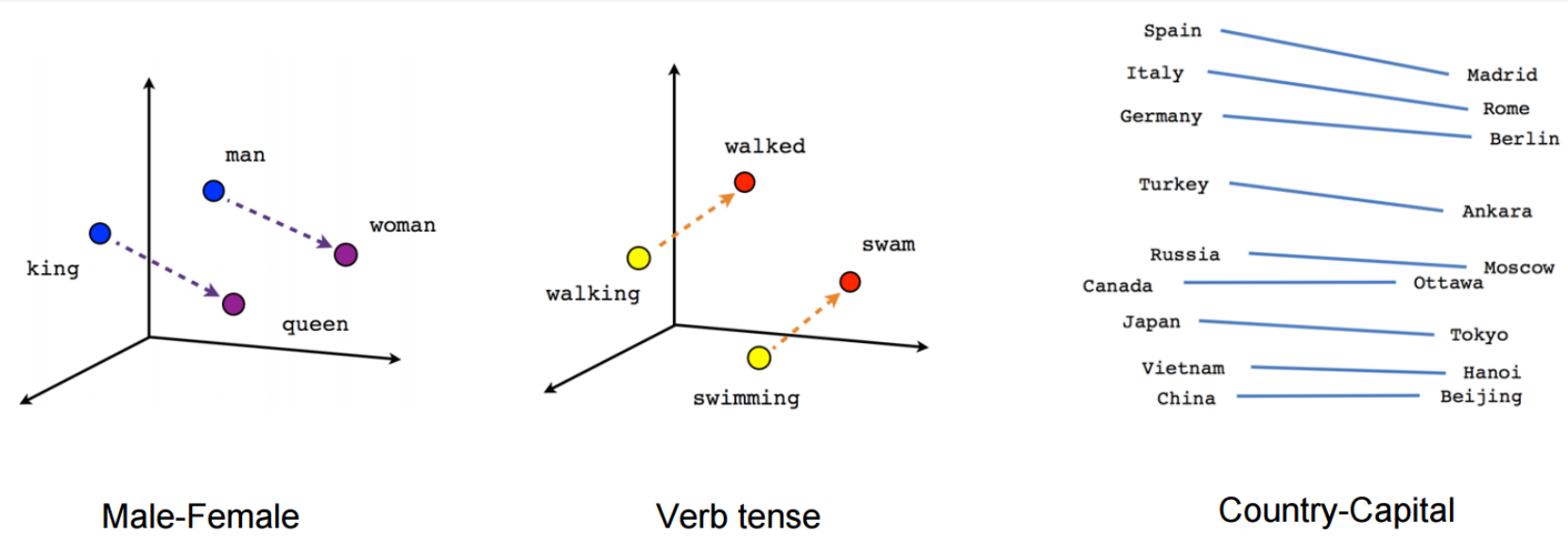
- donde $b_{[w]}$ y $b_{[c]}$ son sesgos entrenados específicos de la palabra y del contexto.

GloVe (2)

- En términos de factorización matricial, si arreglamos $b_{[w]} = \log \#(w)$ y $b_{[c]} = \log \#(c)$ obtendremos un objetivo que es muy similar a factorizar la matriz PMI de contexto de palabras, desplazada por $\log(|D|)$.
- En GloVe, los parámetros de sesgo se aprenden y no se fijan, lo que le da otro grado de libertad.
- El objetivo de optimización es la pérdida por mínimos cuadrados ponderados, asignando más peso a la correcta reconstrucción de los ítems frecuentes.
- Cuando se usa la misma palabra y vocabularios de contexto, el modelo sugiere representar cada palabra como la suma de sus correspondientes vectores de inserción de palabras y contextos.

Word Analogies

- Los wordembeddings pueden capturar ciertas relaciones semánticas, p. ej. hombre-mujer, tiempo verbal y relaciones país-capital entre palabras.
- Por ejemplo, la siguiente relación se encuentra para incrustaciones de palabras entrenado usando Word2Vec: $\vec{w}_{king} - \vec{w}_{man} + \vec{w}_{woman} \approx \vec{w}_{queen}$.



[2]Source: `https://www.tensorflow.org/tutorials/word2vec`

Evaluation

- Hay muchos conjuntos de datos con asociaciones anotadas humanas de pares de palabras o analogías que se pueden usar para evaluar algoritmos de wordembeddings.
- Estos enfoques se denominan *Enfoques de evaluación intrínseca*.
- La mayoría de ellos están implementados en: url https://github.com/kudkudak/word-embeddings-benchmarks.
- Los wordembeddings también se pueden evaluar extrínsecamente usándolas en una tarea externa de NLP (por ejemplo, etiquetado de POS, análisis de sentimiento).

Gensim

Gensim is an open source Python library for natural language processing that implements many algorithms for training word embeddings.

- `https://radimrehurek.com/gensim/`
- `https://machinelearningmastery.com/develop-word-embeddings-python-gensim/`

References

Amir, S., Ling, W., Astudillo, R., Martins, B., Silva, M. J., and Trancoso, I. (2015).
Inesc-id: A regression model for large scale twitter sentiment lexicon induction.
In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 613–618, Denver, Colorado. Association for Computational Linguistics.

Baroni, M., Dinu, G., and Kruszewski, G. (2014).
Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors.
In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 238–247. Association for Computational Linguistics.

Goldberg, Y. (2016).
A primer on neural network models for natural language processing.