NATURAL LANGUAGE PROCESSING

UNIVERSIDAD ICESI

TEXTO

- Los modelos de NLP mas tradicionales (pre Deep learning) representaban textos como **Bag of Words (BoW)**, donde el orden de las palabras no era tenido en cuenta, solo su presencia o ausencia.
- Al igual que con las imágenes, vamos a explotar la **estructura inherente** de los datos analizados (**secuencia**)
- Cada palabra aporta una parte del **contexto** de un texto, información valiosa en términos del significado del mismo.
- Cada nuevo elemento aporta nueva información a la obtenida a través de los elementos anteriores (**actualización del conocimiento**).
- Pretratamiento:

TOKENIZACIÓN

- Los datos de **texto** son categóricos deben codificarse (convertirlos en números) para poder ser tratados por los modelos
- Para simplificar las secuencias de textos, definir los términos que se considerarán como atómicos. A nivel carácter se pierde la semántica del texto.
- Ejemplo:
  No(3) quiero(4) volver(5) a(6) la(1) oficina(2)
  Prefiero(8) trabajar(9) en(7) la(1) casa(10) que(11) en(7) la(1) oficina(2)
- Creación de diccionario de tokens
  - Correspondencia palabra token
  - Toda secuencia de palabras debe utilizar el mismo diccionario
  - Token específico para representar palabras desconocidas (<> o <OOV> Out Of Vocabulary)
- Una alternativa es usar **one-hot encoding**, donde cada elemento textual se representaría por una posición en un vector binario del tamaño del vocabulario. Problema: **dispersión** de los datos por culpa de la dimensionalidad del vocabulario.

TOKENIZACIÓN

- Granularidad de los tokens:
  - En el caso de tokens **caracteres**, vocabulario muy pequeño, tokens compactos, reducción de parámetros en los modelos, PERO cero información semántica
  - En el caso de tokens **palabras**, vocabulario grande que puede ser rico semánticamente PERO tokens dispersos, muchos parámetros en los modelos, y pérdida de información al limitarlo (uso de token <UNK> para palabras muy diferentes entre ellas)
  - Uso de tokens a nivel de **sub-words**, a mitad de camino entre los dos anteriores:
    - Palabras comunes hacen parte del vocabulario
    - Despedazar las otras palabras en sub-palabras comunes con alguna semántica (prefijos, sufijos, raíces), se incluye tokens caracteres
    - Permite codificar palabras nunca antes encontradas
    - Usado en Bert, GPT-3, Electra, …

TOKENIZACIÓN

- Operaciones de pre-procesamiento sobre texto:
  - Eliminación de puntuación, volver todas las letras minúsculas
  - **Stopwords**: eliminar palabras demasiado usuales, sin componente semántico
  - **Stemming**: dejar únicamente la raíz semántica de la palabra (e.g. trabajar → trabaj)
  - **Lematización**: considera el vocabulario completo de un lenguaje para hacer reemplazos semánticos (e.g. quiero → querer, mejor→ bueno)
  - **N-grams**: transformación que agrupa tokens contiguos en nuevos tokens, dado un tamaño de ventana (e.g. con N=2, "quiero trabajar en casa" → "quiero trabajar", "trabajar en", "en casa")
- Normalización del largo de las secuencias:
  - **Padding**: Rellenar secuencias con tokens especiales, al comienzo o al final, para completar el largo definido (token especial de relleno que el tokenizador convertirá en <OOV>)
  - **Truncating**: Cortar secuencias de tokens que sobrepasen el largo establecido, eliminando los primeros o los últimos tokens
  - Estas operaciones se pueden hacer al comienzo (pre) o al final de las secuencias (post)

TALLER

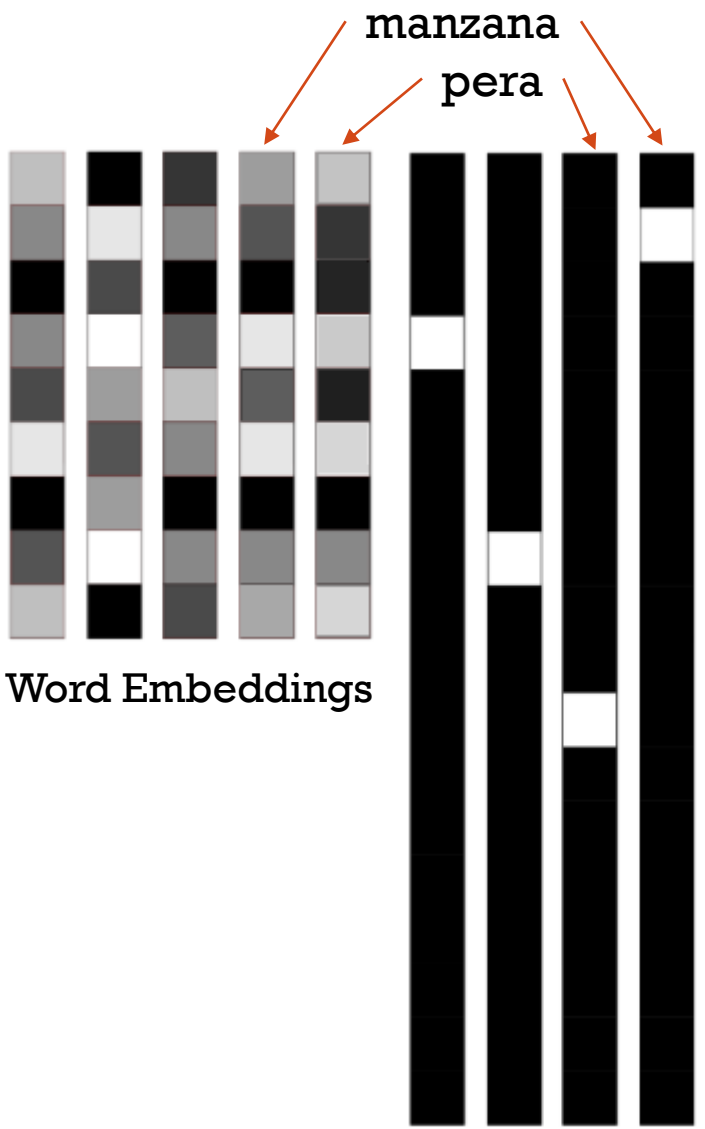
- Ejecutar el notebook de tokenización de textos simples

CODIFICACIÓN

UNIVERSIDAD ICESI

CODIFICACIÓN

- Para poder procesar y analizar secuencias de términos hay primero que definir el tipo de estructura que se va a utilizar para representarlas
- Representaciones:
  - Vectores de palabras con codificación **one-hot**
    - Espacio de representación en "código duro", no interpretable
    - Datos dispersos, alto número de dimensiones, solo ocupan las "esquinas" del espacio
    - Todos los tokens con un ángulo de 90° y a la misma distancia de los demás
  - Vectores de palabras en un **embedding**
    - Datos densos, dimensionalidad reducida, se ocupa mejor el espacio de representación
    - Espacio de representación **semántico** aprendido desde los datos (análogo a los features extraídos de las imágenes con CNNs)



Chollet, 2018

EMBEDDINGS

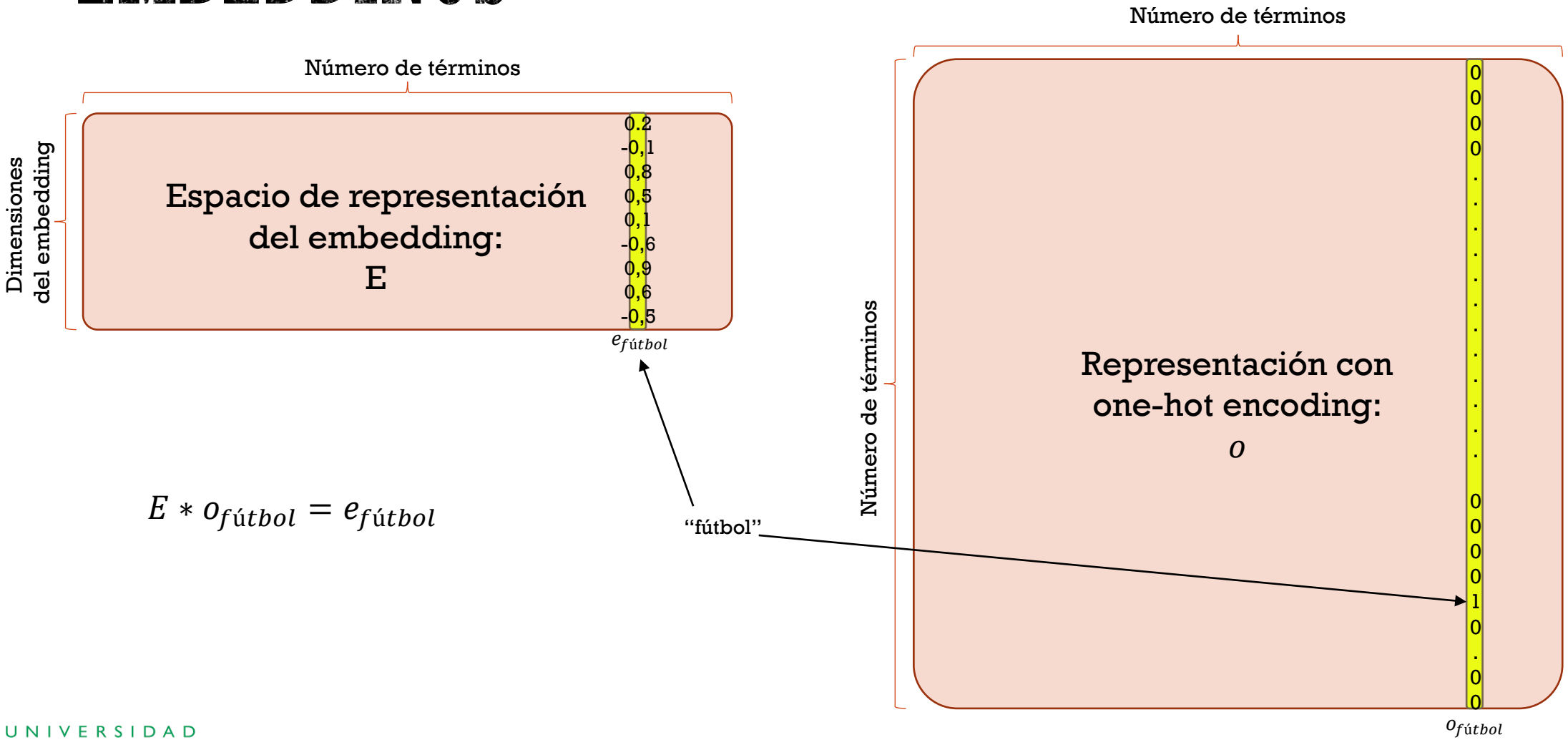
- La representación numérica de los términos debe tener nociones semánticas
  - Cada término se va a **incrustar** en un nuevo espacio de representación (**embedding**) que se debe **aprender** teniendo en cuenta unas secuencias de entrenamiento
  - También se puede utilizar **transfer learning**, importando un espacio de representación semántica pre-entrenado
  - No debe ser producto de un orden arbitrario (i.e. alfabético, frecuencia, aleatorio)
  - Co ocurrencia de términos en contextos similares (con otro tipo de términos en común):

    "Felipe se quebró la pierna jugando fútbol"

    "James se quebró el ligamiento jugando _____"

    playstation, ajedrez, fortnite
  - En un embedding, se utiliza el coseno como medida de similitud entre términos
  - Un proceso de reducción de la dimensionalidad final escogida, como **t-SNE**, debe reconocer la proximidad semántica de los términos

EMBEDDINGS



UNIVERSIDAD ICESI

EMBEDDINGS

- Para aprender un embedding se necesita establecer:
  - Un **corpus** de secuencias (e.g. textos)
  - La granularidad de los términos para lo **tokenización** de las secuencias (e.g. palabras o caracteres)
  - El **vocabulario** de tokens que se van a considerar (e.g. #de términos más frecuentes) con un índice correspondiente
  - El tamaño de las secuencias de tokens de entrada (**padding**)
  - El número de dimensiones de representación del espacio vectorial del **embedding**

medium.com/@jayeshbahire/introduction-to-word-vectors-ea1d4e4b84bf

UNIVERSIDAD ICESI

EMBEDDINGS

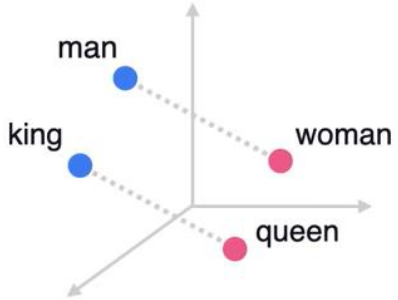
- No hay un embedding perfecto: **dependen del contexto** de las secuencias (e.g. un buen embedding para analizar los sentimientos de las críticas de películas de pronto es muy malo para analizar documentos legales)
- La calidad de un embedding depende del **tamaño del conjunto de secuencias de entrenamiento**. Entra más grande, mejor.
- Una capa embedding:
  - se aprende como una capa inicial de una red, a través de **back-propagation**, conectada a otras capas para el aprendizaje de una **tarea definida** (e.g. capas densas o recurrentes)
  - constituye un diccionario que **mapea** los índices de los tokens a los vectores en el nuevo espacio de representación

EMBEDDINGS

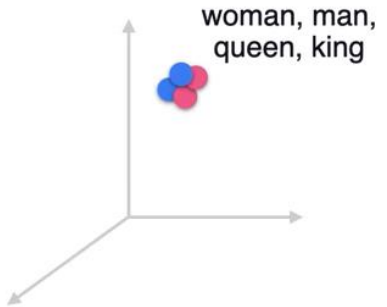
- Una capa **embedding** en una red neuronal:
  - se inicializa aleatoriamente por defecto
  - Procesa secuencias de entrada del mismo largo (en un mismo batch), por lo que se puede tomar 2 decisiones con respecto a un largo definido:
    - **Truncarlas,** si son más largas
    - **Paddearlas** completándolas (con 0s), si son mas cortas
  - recibe tensores de rango 2 como entrada: instancias x largo de la secuencia de términos
  - produce tensores de rango 3 como salida: instancias x largo de secuencia x dimensionalidad del embedding

EMBEDDINGS

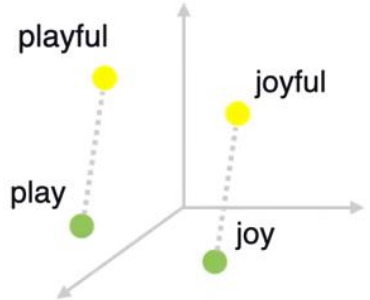
- En el nuevo espacio de representación las relaciones **semánticas** se expresan a partir de relaciones **geométricas vectoriales**.
  - Los ejes se especializan automáticamente siguiendo algún concepto semántico → algunos tokens se van a encontrar cercanos en algunos ejes y lejanos en otros
  - Distancias geométricas representan distancias semánticas
  - Los sinónimos deben estar cerca (coseno como similitud)
  - Las direcciones tienen significado
  - Vectorialmente podemos identificar relaciones como: pluralidad, género, tamaño, conjugaciones
  - Relaciones geométricas y analogías:
    - $\overrightarrow{rey} - \overrightarrow{hombre} + \overrightarrow{mujer} = \overrightarrow{reina}$
    - $\overrightarrow{animal\ salvaje}$ es a $\overrightarrow{mascota}$ lo que $\overrightarrow{lobo}$ es a $\overrightarrow{perro}$



Gender dimensions



Part-of-speech dimensions



Part-of-speech dimensions

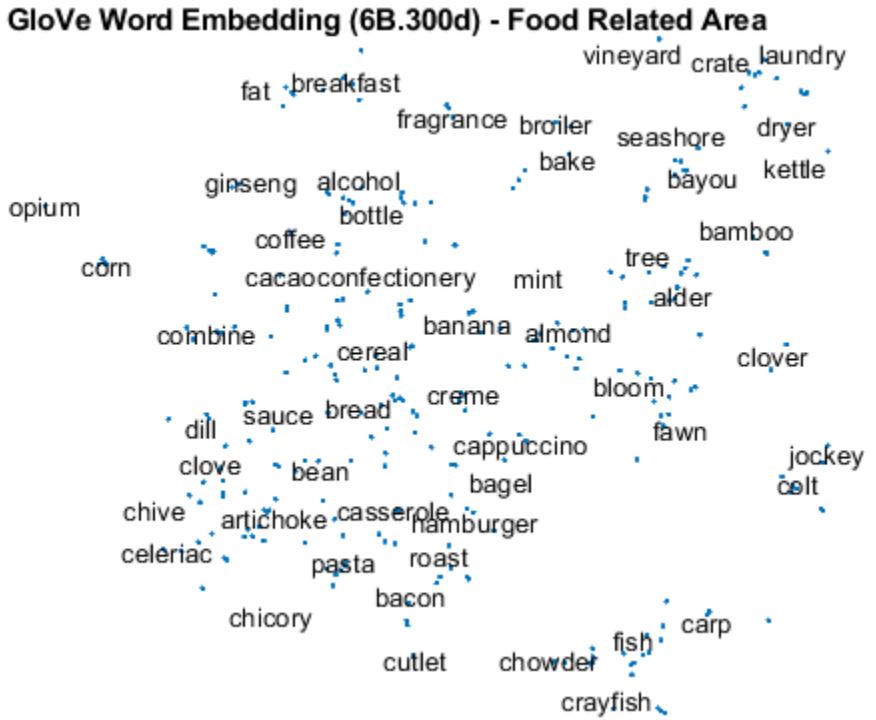
EMBEDDINGS

Transfer Learning:

- El entrenamiento de un embedding tiene **requerimientos** importantes en **tiempo** de procesamiento, **tamaño** del corpus de entrenamiento, y eventualmente de una **infraestructura** computacional importante (GPUs).
- Se puede **reutilizar** embeddings aprendidos para una tarea original, con el fin de realizar tareas análogas con conjuntos de entrenamiento mucho más pequeños.
- Se pueden **afinar** (fine-tune) embeddings transferidos considerando la nueva tarea a aprender.
- Se pueden entrenar embeddings sobre corpus de documentos específicos para diferentes contextos (legal, financiero, etc.)
- Embeddings comunes: Word2Vec, GloVe

T-SNE

- Transformación no lineal de datos representados en alta dimensionalidad a una dimensionalidad reducida (2D o 3D)
- Diferente a PCA (no lineal): los puntos que están cerca tienen una semántica asociada, pero el mapeo aprendido por **T-SNE no conserva las relaciones** geométricas de los embeddings **ni se pueden usar como base para aprendizaje automático** (e.g. no se pueden tomar como base de K-NN o K-Means)
- https://projector.tensorflow.org/

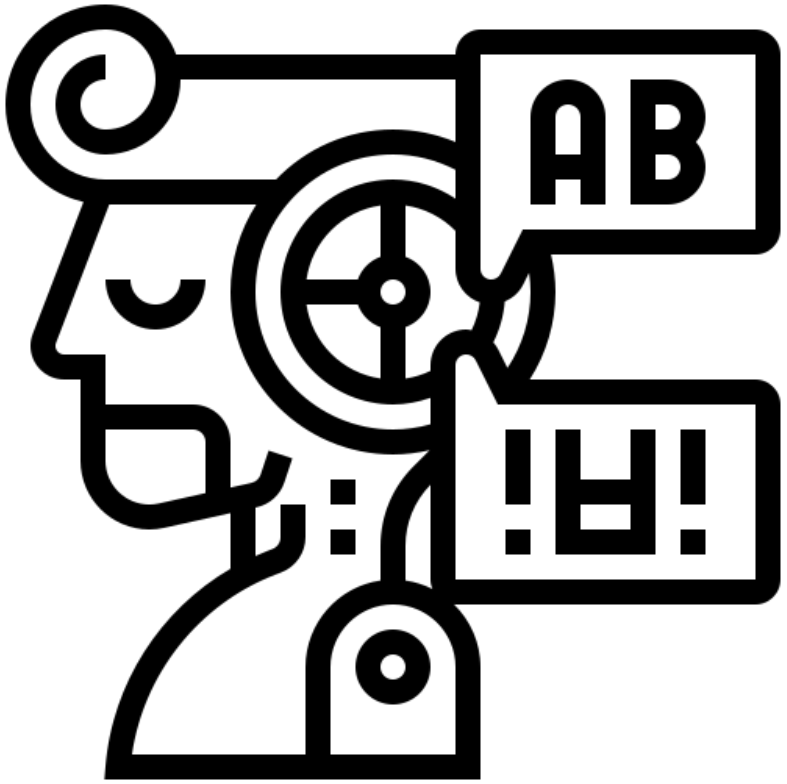


blogs.mathworks.com/loren/2017/09/21/math-with-words-word-embeddings-with-matlab-and-text-analytics-toolbox/

UNIVERSIDAD ICESI

TALLER EMBEDDINGS

- Ejecutar el notebook de aprendizaje de embeddings y de clasificación a partir del dataset de reviews de IMDB.



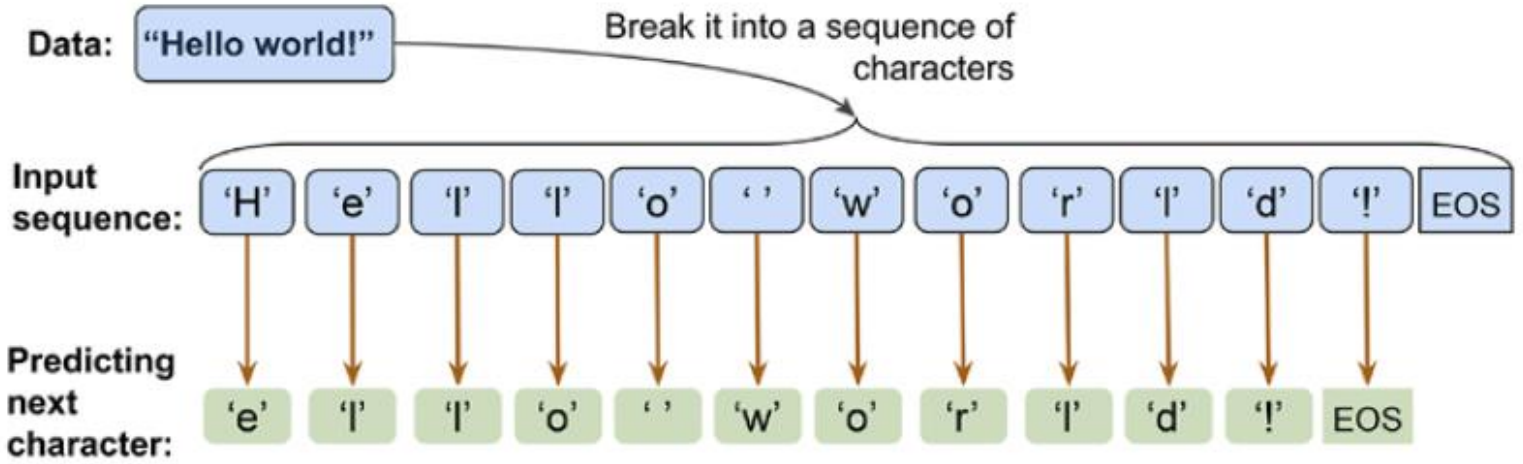
LANGUAGE MODEL

UNIVERSIDAD ICESI

LANGUAGE MODEL

- Un modelo de lenguaje permite realizar la tarea de predecir o generar los siguientes tokens, dados los anteriores; predecir texto a partir del contexto.
- Se modela probabilísticamente así:

$$P(token_n | token_{n-1}, token_{n-2}, \dots, token_0)$$



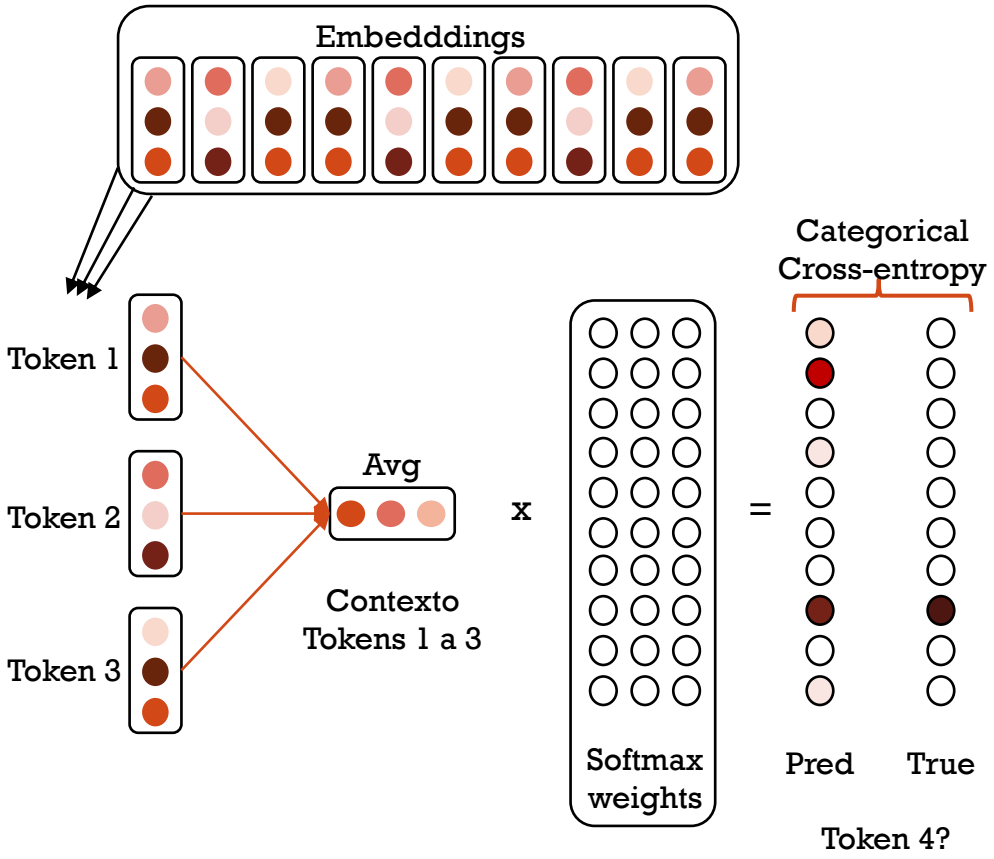
Raschka, 2022

LANGUAGE MODEL

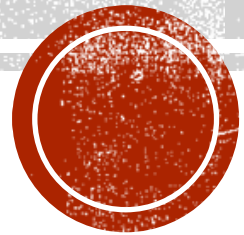
- Un modelo de lenguaje se puede ver como un modelo de clasificación, que a partir de una secuencia de tokens se predice el siguiente.
- Entre mas larga sea la secuencia de entrada de un modelo de lenguaje, mejor será la semántica de la generación de nuevos tokens. Sin embargo, algunos modelos recurrentes simples (RNNs) no logran capturar dependencias de largo alcance en secuencias largas. → Búsqueda de hiperparámetros del largo máximo ideal de secuencia.

WORD2VEC

- Definir un vocabulario
- Definir la dimensionalidad del espacio de representación
- Inicializar embedding aleatorio
- Inicializar matriz de pesos para predicción por softmax
- Entrenar el modelo actualizando los parámetros:
  - Matriz de pesos
  - Embeddings
- Problemas:
  - No puede manejar homónimos
  - No tiene en cuenta secuencialidad de tokens



GRACIAS



UNIVERSIDAD ICESI

REFERENCIAS

- *Machine Learning with PyTorch and Scikit-Learn*, Sebastian Raschka, Yuxi Liu & Vahid Mirjalili, Packt, 2022
- *Dive into Deep Learning*, Aston Zhang, Zachary C. Lipton, Mu Li & Alexander J. Smola, https://d2l.ai/, 2022
- *TensorFlow ML Tech Talks*
- *Python Machine Learning (3rd ed.)*, Sebastian Raschka & Vahid Mirjalili, Packt, 2019
- *Neural Networks and Deep Learning,* Andrew Ng, Coursera, 2017
- *Deep Learning with Python*, Francois Chollet, Manning 2018
- *Hands-On Machine Learning with Scikit-Learn and TensorFlow*, Aurélien Géron, 2017