Escuela de ingeniería y ciencias.

Campus **Monterrey**

Unidad de formación **TC3006C.102**

Inteligencia artificial avanzada para la ciencia de datos II

Semana 6: Text Embeddings.

Equipo **3**

Grupo: **102**

Regina Reyes Juárez - **A01275790**
Nadia Salgado Alvarez - **A01174509**
Gilberto Angel Camacho Lara - **A01613895**
Santiago Miguel Lozano Cedillo - **A01198114**

09 de noviembre

Semana 6: Text Embeddings.ipynb

Introduccion
En este trabajo se desarrolla un modelo de clasificación de mensajes SMS para identificar si un mensaje es spam (no deseado) o ham (legítimo). El objetivo principal del notebook es aplicar redes neuronales con Text Embeddings, probando distintas arquitecturas de Deep Learning para comparar su desempeño y elegir la más adecuada.

Descripción del dataset

Para ello, utilizamos el dataset SMS Spam Collection, el cual contiene mensajes reales etiquetados como spam o no spam. Primero realizamos una exploración y limpieza de los textos para dejarlos en un formato adecuado para el modelo, eliminando stopwords (palabras como conectores, comas, símbolos, espacios y urls que aparezcan en el texto). Después, convertimos cada mensaje en una representación numérica mediante embeddings, ya que una red neuronal no puede trabajar directamente con texto, por lo que tenemos que convertirlo en vectores numéricos.

El dataset utilizado es SMS Spam Collection, disponible en Kaggle y comúnmente usado en tareas de clasificación de texto. Contiene 5,574 mensajes SMS en inglés, cada uno etiquetado como ham (mensaje legítimo) o spam (mensaje no deseado). Los mensajes provienen de diferentes fuentes reales, como reportes de usuarios y corpus académicos. Este dataset es relevante porque representa un problema cotidiano: identificar automáticamente mensajes de spam, por lo que es ideal para entrenar y evaluar modelos de procesamiento de lenguaje natural.

Es importante mencionar que el dataset presenta un desbalance, ya que la mayoría de los mensajes corresponden a la categoría ham. Por esta razón, además de la precisión, se consideraron métricas como F1 y AUC, que permiten evaluar mejor el desempeño del modelo en la detección de la clase minoritaria (spam).

Para preparar los mensajes antes de entrenar los modelos, primero convertimos las etiquetas de texto a valores numéricos, donde spam se representó como 1 y ham como 0. Esto facilita que los modelos puedan interpretar la variable objetiva durante el entrenamiento.

El objetivo de estas transformaciones fue obtener mensajes más uniformes y libres de ruido, manteniendo únicamente la información valiosa. Después, cada palabra fue convertida en un número entero mediante tokenización, basándonos en la frecuencia de aparición dentro del conjunto de datos. Finalmente, se aplicó padding para que todos los mensajes tuvieran la misma longitud (max_len = 40) donde si algunos no

tiene la misma longitud se rellenan con ceros haciendo que este hiperparámetro sea fundamental debido a que si es mayor o menor de lo necesario podemos meter ruido o cortar información importante. Esto permitió que los modelos de Deep Learning pudieran procesar los mensajes de forma consistente, ya que todos comparten la misma estructura de entrada.

Modelos

Entrenamos tres modelos distintos para ver su diferente rendimiento:

- Modelo A: Embedding + Global Average Pooling
- Modelo B: Embedding + BiLSTM
- Modelo C: Embedding + Conv1D

“Para datos cortos y con señales léxicas muy claras, modelos simples generalizan igual o mejor que modelos profundos. El mejor modelo no es el más complejo, sino el que logra el mejor desempeño con el menor costo computacional dependiendo del caso.”

Modelo A

Este modelo promedia todos los vectores generados por el embedding de cada palabra del mensaje.
Al calcular el promedio, pierde el orden de las palabras, por lo que no captura relaciones secuenciales, pero es muy rápido y eficiente.
Su simplicidad lo hace ideal para datasets pequeños y para usarlo como modelo base (baseline) de comparación inicial.

Modelo B

Este modelo utiliza una red neuronal recurrente bidireccional (BiLSTM), que procesa el texto tanto de izquierda a derecha como de derecha a izquierda.
Esto le permite comprender el contexto completo de una oración y recordar dependencias entre palabras gracias a la memoria interna de las LSTM.
Es más robusto y preciso en tareas donde el orden de las palabras importa, aunque requiere más recursos computacionales y tiene más hiperparámetros que ajustar.

Modelo C

Este modelo emplea una capa convolucional 1D (Conv1D) para extraer características locales del texto.

Los filtros de la convolución funcionan como detectores de patrones (por ejemplo, frases típicas de spam como “free offer” o “click here”). Gracias a esto, es muy eficaz en tareas de clasificación de texto, especialmente con grandes volúmenes de datos, siendo además más eficiente que una RNN en velocidad de entrenamiento.

Resultados

Modelo	**Accuracy**	**Precision**	**Recall**	**F1**	**AUC**
Modelo C	*0.98*	*0.99*	*0.91*	*0.94*	*0.99*
Modelo A	*0.98*	*1.00*	*0.94*	*0.94*	*0.98*
Modelo B	*0.98*	*0.95*	*0.91*	*0.93*	*0.98*

Para asegurar la consistencia de los resultados, se fijó una semilla aleatoria (SEED = 42). Tras comparar las métricas de desempeño, se observó que el Modelo C (Embedding + Conv1D) obtuvo el mejor rendimiento, aunque por una diferencia muy pequeña respecto a los otros modelos.
Este resultado tiene sentido, ya que se trata de un problema de clasificación, donde las convoluciones son especialmente eficaces para detectar patrones locales en el texto.
Además, el Modelo C presenta menor costo computacional y mayor eficiencia en el manejo de volúmenes de datos grandes, lo que representa una ventaja clara frente a los demás.
No obstante, esto no implica que los otros modelos sean inadecuados, sino que, para este caso específico, el enfoque convolucional resultó ser el más apropiado y equilibrado.

Referencias

- Ishansoni. (2018, 6 octubre). *SMS spam collection Dataset*. Kaggle.

  https://www.kaggle.com/code/ishansoni/sms-spam-collection-dataset/notebook