Nicolás Rozo Fajardo – 202112920
Nicolás Bedoya Figueroa – 202212100

INFORME DE IMPLEMENTACIÓN – TAREA 4

1. PREPARACIÓN DEL CONJUNTO DE DATOS

El proceso de preparación del conjunto de datos comienza con la limpieza y tokenización de los textos provenientes del *Project Gutenberg*. Cada archivo de texto es leído y procesado, eliminando los encabezados y pies de página. De esta manera, se conserva únicamente el contenido literario relevante de cada obra, evitando la introducción de ruido en el corpus.

Posteriormente, se realiza una tokenización a nivel de oraciones y palabras. En esta etapa, el texto se convierte a minúsculas, se eliminan signos innecesarios, números, palabras poco relevantes (*stopwords*), y se filtran aquellos tokens que no cumplen con un patrón léxico válido. Al finalizar el proceso de tokenización solo se conservan las oraciones con al menos dos palabras significativas, generando una representación limpia del lenguaje propio de cada autor.

Cada libro presente en el conjunto de datos se descompone en una lista de oraciones tokenizadas que se agregan a un corpus general, representado como una lista de listas de palabras. Este corpus constituye la base de datos textual sobre la que posteriormente se entrenan los modelos de embeddings o incrustaciones.

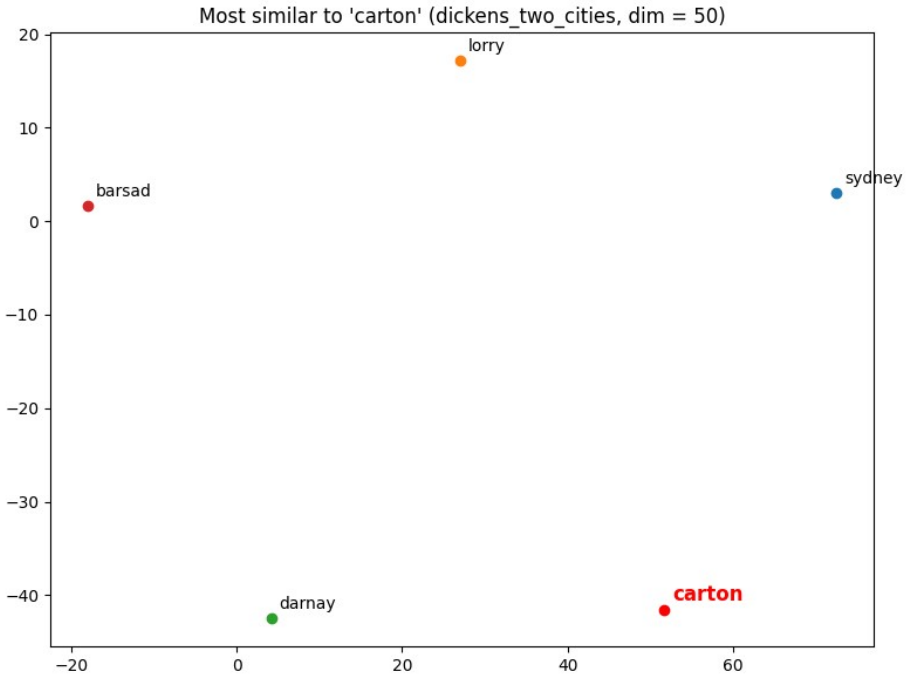
Para la construcción del conjunto de datos de clasificación se usan los textos preprocesados para crear ejemplos etiquetados según el autor de cada obra. Antes de etiquetar, el texto se divide en chunks, segmentando cada obra en bloques de tamaño definido y, por ende, garantizando un número constante de tokens por fragmento a procesar. Finalmente, el conjunto de datos de clasificación se divide en los subconjuntos de entrenamiento (70%), validación (15 %) y prueba (15 %), asegurando una distribución balanceada de clases entre los distintos autores. El resultado de este proceso es un corpus estructurado y etiquetado, adecuado para llevar a cabo tareas de clasificación.

2. ESTRATEGIA DE VISUALIZACIÓN DE EMBEDDINGS EN 2 DIMENSIONES

Para explorar las relaciones semánticas entre los personajes literarios representados en los modelos de embeddings, es necesario reducir la dimensionalidad de los vectores para poder visualizarlos en dos dimensiones. Cada modelo fue entrenado con un corpus específico de novelas o autores (por ejemplo, *austen_pride*, *dickens_expectations*, *twain_tom*, etc.), produciendo vectores de palabras de 50 dimensiones que capturan similitudes contextuales entre términos y nombres propios.

Para proyectar estos vectores de alta dimensión en un plano bidimensional se pueden emplear distintas técnicas, como PCA o t-SNE. En este caso, se optó por t-SNE (t-Distributed Stochastic Neighbor Embedding), un método no lineal diseñado para preservar las relaciones locales entre puntos. t-SNE funciona en dos etapas principales: primero, calcula probabilidades de similitud entre puntos en el espacio original, de modo que pares de vectores similares tengan altas probabilidades de aparecer juntos. Luego, en el espacio de menor dimensión, busca una distribución que reproduzca esas probabilidades lo mejor posible, minimizando la divergencia entre ambas distribuciones mediante un procedimiento iterativo. El resultado es una representación visual donde las palabras con contextos similares se agrupan de forma natural.

Una vez obtenida la proyección en 2D, se seleccionaron las palabras más cercanas a cada personaje principal según la similitud coseno. Estas palabras se graficaron, destacando en rojo el nombre del protagonista. Las visualizaciones permiten examinar de forma cualitativa cómo la proximidad semántica en el espacio de embeddings refleja las estructuras narrativas y relacionales dentro de cada obra. En todas las representaciones se observa que las palabras se agrupan de manera coherente alrededor de los protagonistas, lo que demuestra que los embeddings capturan efectivamente el contexto narrativo. Todas las gráficas realizadas se encuentran en el directorio *images*, siguiendo una estructura similar a la de la siguiente imagen.



A continuación, se presentan las relaciones más importantes identificadas a partir de las obras de cada autor presente en el conjunto de datos.

- **Mark Twain:** En Las aventuras de Tom Sawyer y Huckleberry Finn, Tom, Becky y Huck aparecen próximos entre sí, reflejando su estrecho vínculo narrativo. En cambio, el vecindario semántico de Jim incluye palabras como “contrabando” y “gritos”, lo que evoca su papel en el viaje por el río y los temas de la esclavitud. En El príncipe y el mendigo, los grupos de palabras muestran una clara oposición: “príncipe” se asocia con “rey” y “Gales”, mientras que “mendigo” se relaciona con “adorar” y “cuelga”, codificando así el contraste social entre riqueza y pobreza.
- **Jane Austen:** En *Orgullo y prejuicio*, los vectores de Darcy y Elizabeth están estrechamente alineados con Bennet y Bingley, reflejando las asociaciones románticas y familiares centrales de la trama. En *Emma*, Emma y Knightley aparecen próximos a Harriet y Weston, representando la red social y las dinámicas emocionales de la novela. De forma similar, en *Sentido y sensibilidad*, Elinor y Marianne se agrupan con Dashwood y Willoughby, mostrando las conexiones familiares y la tensión afectiva que atraviesa la historia.
- **Charles Dickens:** En *Grandes esperanzas*, Pip se asocia semánticamente con Joe, mientras que Havisham aparece cerca de Estella y Miss, lo que corresponde a los temas de clase y moral. En *Historia de dos ciudades*, Carton y Darnay ocupan posiciones vecinas, reflejando su dualidad moral y física. En *Oliver Twist*, Oliver se agrupa con Brownlow y Twist, mientras que Fagin se conecta con Sikes y Charley, ilustrando el contraste entre la inocencia y la corrupción.

3. DESCRIPCIÓN DE LAS ARQUITECTURAS FEED-FORWARD

El modelo A utiliza una arquitectura sencilla orientada a la eficiencia. Comienza con una capa de embeddings que convierte cada palabra en un vector de tamaño fijo, generando una representación tridimensional del texto con forma (batch_size, secuencia, embedding_dim). Luego, la capa GlobalAveragePooling1D promedia los vectores de toda la secuencia, condensando la información en un único vector por texto de tamaño (embedding_dim). Posteriormente, una capa densa con 32 neuronas y activación ReLU extrae características relevantes a partir de esa representación global. Finalmente, la capa de salida con activación softmax produce una distribución de probabilidad sobre los autores posibles, generando un vector de tamaño (num_authors). Esta arquitectura es compacta y eficiente, adecuada para capturar diferencias generales en estilo sin sobreajuste.

El modelo B expande la capacidad representacional al incluir dos capas densas intermedias. Al igual que en el modelo A, la capa de embeddings y el GlobalAveragePooling1D convierten cada texto en un vector promedio de tamaño (embedding_dim). A partir de ahí, la primera capa densa con 128 neuronas y

activación ReLU extrae un gran número de características. Una segunda capa con 64 neuronas obtiene más características antes de llegar a la capa de salida softmax, encargada de clasificar el texto entre los distintos autores. Gracias a esta estructura jerárquica, el modelo B logra un equilibrio entre complejidad y generalización, siendo más expresivo que el modelo A sin un costo computacional excesivo.

El modelo C presenta la arquitectura más profunda y de mayor dimensionalidad entre las tres. A diferencia de los anteriores, sustituye el GlobalAveragePooling1D por una capa Flatten, que aplana la secuencia de embeddings completa, transformando la salida en un vector largo de tamaño (sequence_length × embedding_dim). Esto conserva información más granular de cada palabra en la secuencia, aunque incrementa significativamente el número de parámetros. Luego, dos capas densas sucesivas de 256 y 128 neuronas con activación ReLU permiten aprender representaciones altamente no lineales, capturando características en los textos. La capa final softmax clasifica los textos según el autor. Esta arquitectura es más potente pero también más propensa al sobreajuste, ideal para capturar relaciones profundas si se dispone de un corpus suficientemente grande.

	accuracy	precision	recall
A_50D	0.967	0.966	0.968
B_50D	0.972	0.967	0.976
C_50D	0.965	0.964	0.963
A_100D	0.957	0.959	0.954
B_100D	0.980	0.976	0.983
C_100D	0.954	0.959	0.948
A_200D	0.970	0.966	0.972
B_200D	0.980	0.979	0.980
C_200D	0.952	0.958	0.941

Los resultados muestran un desempeño consistentemente alto en las tres arquitecturas y para todas las dimensiones de embeddings (50D, 100D y 200D), con precisiones, recall y exactitudes superiores al 94%. Sin embargo, se observa que el modelo B presenta el mejor rendimiento general, alcanzando los valores más altos de accuracy (0.980) tanto con embeddings de 100 como de 200 dimensiones. Esto sugiere que su estructura intermedia logra un equilibrio óptimo entre capacidad de representación y generalización, capturando mejor los patrones estilísticos de los

autores sin sobreajustarse. En contraste, el modelo C, a pesar de ser más complejo, muestra un desempeño ligeramente inferior, especialmente en recall, lo cual puede deberse a un exceso de parámetros que no aporta mejoras significativas al trabajar con un corpus limitado. Finalmente, el modelo A, aunque más simple, mantiene resultados notables y estables, demostrando que incluso arquitecturas ligeras pueden ser efectivas cuando se realiza el entrenamiento de embeddings para capturar relaciones semánticas.

4. COMPARACIÓN DE RESULTADOS CON EMBEDDINGS GLOVE

	accuracy	precision	recall
A_50D	0.830	0.827	0.820
B_50D	0.871	0.874	0.858
C_50D	0.800	0.807	0.772
A_100D	0.858	0.856	0.854
B_100D	0.891	0.924	0.860
C_100D	0.828	0.837	0.801
A_200D	0.896	0.897	0.889
B_200D	0.934	0.932	0.928
C_200D	0.853	0.861	0.829

Al comparar ambos conjuntos de resultados, se observa una diferencia clara entre el uso de embeddings personalizados entrenados sobre el corpus y los embeddings preentrenados GloVe. Los modelos con embeddings personalizados (primer conjunto de resultados) alcanzan métricas más altas en general (todas por encima de 0.95 en accuracy, precision y recall) mientras que los modelos con GloVe (segundo conjunto de resultados) presentan valores más bajos, situándose entre 0.80 y 0.93. Esto indica que, en este caso, los embeddings entrenados directamente sobre el corpus capturan mejor los patrones lingüísticos y estilísticos particulares de los textos, mientras que los embeddings preentrenados, aunque más generales, no reflejan las particularidades de estilo propias de cada autor.

En cuanto a la influencia de la dimensionalidad, los resultados muestran un comportamiento distinto según el tipo de embedding. En los embeddings personalizados, el incremento de dimensiones (de 50D a 200D) no produce mejoras significativas, manteniéndose un rendimiento muy alto en todos los casos. Esto sugiere que el modelo logra representar suficientemente bien la información estilística incluso con dimensiones moderadas, posiblemente porque el corpus es limitado y las dimensiones adicionales no generan un gran aporte. En cambio, con los embeddings preentrenados GloVe, sí se observa una mejora notable al aumentar

la dimensionalidad: los resultados crecen progresivamente desde 50D hasta 200D, alcanzando el mejor desempeño en B_200D (0.934 de accuracy). Esto se debe a que embeddings de mayor dimensión en GloVe contienen representaciones semánticas más finas que permiten compensar de forma parcial la falta de ajuste al dominio del corpus.

En resumen, los embeddings entrenados en el corpus son más efectivos para tareas donde el estilo y el vocabulario son específicos (como la atribución de autoría), mientras que los preentrenados GloVe pueden ser útiles en contextos más generales o cuando no hay suficiente texto para entrenar embeddings propios. La dimensionalidad, en ambos casos, influye positivamente hasta cierto punto, pero su beneficio depende de la naturaleza del embedding: en embeddings generales, más dimensiones aportan riqueza semántica; en embeddings personalizados, la ganancia es marginal una vez que se captura la variabilidad del corpus.