Informe - GloVe

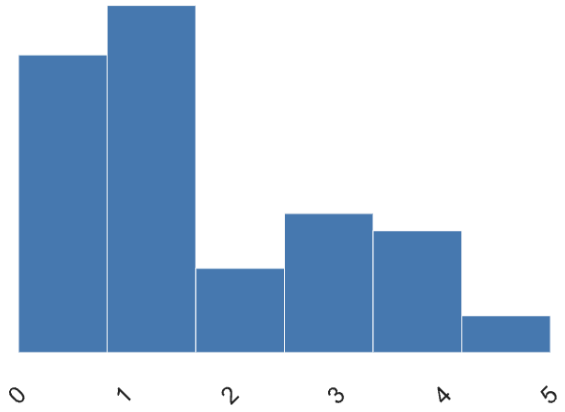
Objetivo
El propósito de este trabajo fue mejorar el rendimiento del modelo base mediante el uso de embeddings pre entrenados de GloVe.

Preprocesamiento de Datos
En primer lugar, se llevó a cabo un preprocesamiento exhaustivo de los datos, que incluyó la eliminación de palabras vacías (stopwords) para reducir el ruido y enfocarse en los términos más relevantes del texto.

Antes **Después**

Valores Duplicados
El análisis exploratorio de datos reveló la presencia de un único valor duplicado. Dado que su impacto en el modelo es mínimo, se decidió mantenerlo en el conjunto de datos.

Manejo del Desbalance de Clases
Dado el considerable desbalance de clases presente en los datos, se intentaron aplicar técnicas de sobre-muestreo (oversampling) y sub-muestreo (undersampling). Sin embargo, estas técnicas no mostraron mejoras significativas en los resultados, por lo que optamos por no implementarlas en la versión final del modelo. Creemos que el fracaso de estas técnicas puede estar relacionado con la marcada disparidad en la cantidad de datos entre las clases mayoritarias (0 y 1) y las demás clases. En el caso del sub-muestreo, la reducción drástica de datos podría haber disminuido la capacidad del modelo para generalizar correctamente. Por otro lado, en el sobre-muestreo, la triplicación o más de los datos podría haber introducido un sesgo y un sobreajuste, lo que afecta negativamente la capacidad del modelo para manejar nuevas muestras.



Implementación de Técnicas de Pérdida Avanzadas
Inicialmente, se optó por utilizar la función de pérdida Cross-Entropy, que es comúnmente empleada en tareas de clasificación. Sin embargo, durante el entrenamiento, se observó que las clases minoritarias presentaban un rendimiento considerablemente inferior. Para abordar este desafío, se decidió implementar la Focal Loss, una función de pérdida diseñada para dar mayor énfasis a los ejemplos difíciles, como aquellos pertenecientes a las clases menos representadas.

El uso de Focal Loss resultó en una reducción en la pérdida de validación, lo que sugiere que el modelo estaba mejorando su capacidad de aprendizaje en general. No obstante, también se detectó una disminución en la precisión global del modelo. Para mitigar este efecto, se optó por implementar una función de pérdida híbrida (Hybrid Loss), que combina Focal Loss con Cross-Entropy Loss. El objetivo de esta combinación es equilibrar la mejora en el rendimiento de las clases minoritarias sin sacrificar la precisión general del modelo.

La Hybrid Loss es una función de pérdida que integra dos funciones de pérdida distintas, como Focal Loss y Cross-Entropy Loss, para aprovechar las ventajas de ambas. Su propósito es mejorar la capacidad del modelo para manejar clases desbalanceadas (gracias a la Focal Loss) mientras se preserva la precisión global (gracias a la Cross-Entropy Loss). Esto se logra mediante una ponderación controlada de ambas pérdidas durante el proceso de entrenamiento.

Selección del Modelo y Estrategias de Optimización
Se probaron varios modelos para abordar el problema, incluyendo RNN, LSTM unidireccional y bidireccional, y SVM. Sin embargo, el modelo que ofreció el mejor rendimiento fue la CNN, por lo que se decidió utilizar esta arquitectura como modelo final.

Para optimizar el entrenamiento, se implementó la técnica de Early Stopping con el objetivo de prevenir el sobreajuste, deteniendo el entrenamiento cuando la pérdida de validación dejaba de mejorar después de un número determinado de épocas.

Se experimentó con dos optimizadores diferentes: Adam y AdamW. Finalmente, se optó por AdamW, ya que demostró una mejor capacidad para manejar la regularización mediante el decaimiento del peso (weight decay), lo que contribuyó a mejorar la generalización del modelo.

Además, se decidió utilizar un scheduler ReduceLROnPlateau con el fin de ajustar dinámicamente la tasa de aprendizaje, reduciéndola cuando la pérdida de validación dejaba de mejorar, lo que permitió al modelo refinar su aprendizaje en las últimas etapas del entrenamiento.

Finalmente, se ajustaron los hiperparámetros, como la tasa de aprendizaje (learning rate) y el decaimiento de peso (weight decay), la cantidad de capas convolucionales, y el tamaño del batch, entre otros, hasta encontrar la combinación que ofreciera el mejor rendimiento.