AEC2 - SPAM

Memoria – Competición Kaggle SPAM / NOT SPAM 2025

Introducción y contexto

Esta memoria se enmarca dentro de la AEC2 (Actividad de Evaluación Continua 2) de la asignatura de Inteligencia Artificial. El objetivo principal de este proyecto fue desarrollar un modelo de Procesamiento de Lenguaje Natural (NLP) capaz de clasificar mensajes de texto como SPAM o NOT SPAM con la mayor precisión posible.

Al igual que en la actividad anterior, la competición se desarrolló en la plataforma Kaggle, donde evalué el rendimiento de los modelos mediante el Matthews Correlation Coefficient (MCC). Esta métrica resulta especialmente adecuada para este problema, ya que proporciona una evaluación equilibrada incluso cuando las clases están desbalanceadas, una situación habitual en la detección de correo no deseado.

A lo largo de este documento describo, de forma cronológica, las distintas iteraciones que realicé durante el desarrollo del proyecto, detallando los cambios introducidos, la motivación detrás de cada decisión y los resultados obtenidos.

Desarrollo del modelo

Iteración 0: Starter Notebook (V0)

Fecha: 28/11/2025

Descripción del cambio

Planteé esta primera iteración como punto de partida, utilizando el notebook proporcionado por la asignatura e implementando una red neuronal básica para la clasificación de texto.

Hipótesis y justificación

El objetivo de esta iteración fue establecer un entorno de trabajo funcional y comprender el flujo completo de datos, incluyendo la carga, el preprocesamiento, el entrenamiento del modelo y la generación del archivo de submission.

Resultados obtenidos

Validation MCC aproximado de 0.60. El modelo resultó funcional, aunque muy simple, sirviendo únicamente como base para futuras mejoras.

Iteración 1: Modelo baseline Bi-LSTM (V1)

Fecha: 04/12/2025

Descripción del cambio

En esta iteración utilicé el notebook base proporcionado como plantilla e incorporé una arquitectura de red neuronal recurrente con una capa de Embedding seguida de una LSTM bidireccional de 128 unidades y una capa GlobalMaxPooling1D.

Hipótesis y justificación

La hipótesis principal fue que una LSTM bidireccional permitiría capturar mejor el contexto semántico de los mensajes al analizar el texto tanto de izquierda a derecha como de derecha a izquierda. El uso de GlobalMaxPooling1D permite extraer las características más relevantes de toda la secuencia procesada por la LSTM.

Resultados obtenidos:

Validation MCC de 0.8585. El rendimiento fue notablemente superior al modelo inicial, aunque se observó un sobreajuste evidente entre los conjuntos de entrenamiento y validación.

Referencia:

Stack Overflow. Difference between LSTM and Bidirectional LSTM.

https://stackoverflow.com/questions/43035827/whats-the-difference-between-lstm-and-bidirectional-lstm

Schuster, M. & Paliwal, K. Bidirectional recurrent neural networks.

https://ieeexplore.ieee.org/document/650093

Estas 2 referencias se utilizaron para respaldar teóricamente el uso de LSTM bidireccionales para capturar dependencias contextuales en secuencias de texto.

Stack Overflow. Difference between LSTM and Bidirectional LSTM.

https://stackoverflow.com/questions/43035827/whats-the-difference-between-lstm-and-bidirectional-lstm

Esta referencia se utilizó para confirmar que, en secuencias de texto cortas, la bidireccionalidad suele aportar una mejora adicional de rendimiento.

Iteración 2: Mejora en la regularización (V2)

Fecha: 04/12/2025

Descripción del cambio

En esta versión introduje regularización L2 con un valor de 1e-4 en las capas densas y recurrentes, incrementé el Dropout hasta 0.5 y añadí una capa SpatialDropout1D con valor 0.3 justo después del Embedding.

Hipótesis y justificación

Tras detectar un fuerte sobreajuste en la versión anterior, planteé que el modelo estaba memorizando palabras concretas del conjunto de entrenamiento. SpatialDropout1D elimina canales completos del embedding, obligando al modelo a no depender de dimensiones específicas del vector de palabras. La regularización L2 penaliza pesos excesivamente grandes y favorece una mejor generalización.

Resultados obtenidos

Validation MCC de 0.8685. El sobreajuste se redujo de forma apreciable y el valor de MCC se mantuvo, indicando una mejora en la capacidad de generalización.

Referencias

Reddit r/MachineLearning. SpatialDropout1D vs Dropout.

https://www.reddit.com/r/MachineLearning/comments/5fiv98/d_spatial_dropout_vs_normal_dropout/

Srivastava, N. et al. Dropout: A Simple Way to Prevent Neural Networks from Overfitting.

https://jmlr.org/papers/v15/srivastava14a.html

Ambas referencias se utilizaron para justificar el uso combinado de Dropout, SpatialDropout1D y regularización como mecanismos efectivos contra el overfitting.

Iteración 3: Regularización agresiva (V3)

Fecha: 05/12/2025

Descripción del cambio

En esta iteración apliqué una regularización más intensa, elevando el Dropout hasta 0.7, aumentando los coeficientes de L2 y reduciendo el número de unidades de la LSTM con el objetivo de simplificar el modelo.

Hipótesis y justificación

El propósito de esta iteración fue comprobar si una regularización más agresiva permitiría mejorar la generalización en el conjunto de test de Kaggle, aun a costa de una ligera pérdida de rendimiento en validación.

Resultados obtenidos

Validation MCC de 0.8533. El modelo mostró síntomas de underfitting parcial.

Iteración 4: Transfer learning con DistilBERT (V4)

Fecha: 06/12/2025

Descripción del cambio

En esta iteración experimenté con un modelo DistilBERT preentrenado de la librería HuggingFace, realizando fine-tuning de las últimas capas y empleando el optimizador AdamW con weight decay.

Hipótesis y justificación

La hipótesis inicial fue que el conocimiento previo adquirido durante el preentrenamiento de DistilBERT permitiría compensar el tamaño limitado del dataset.

Resultados obtenidos

Validation MCC de 0.6456. Los resultados fueron muy inferiores e inestables.

Referencias:

HuggingFace Forums. Fine-tuning BERT on very small datasets.

https://discuss.huggingface.co/t/fine-tuning-bert-on-very-small-dataset/3468

Referencia utilizada para confirmar que, con datasets reducidos, los Transformers requieren tasas de aprendizaje extremadamente bajas para evitar colapsos durante el entrenamiento. El modelo resultó demasiado complejo para el tamaño y características del dataset.

Iteración 5: Arquitectura híbrida CNN + LSTM – Modelo final (V5)

Fecha: 10/12/2025

Descripción del cambio

Diseñé una arquitectura híbrida compuesta por ramas paralelas de Conv1D con tamaños de kernel 2, 3, 4 y 5, junto con una Bi-LSTM. Las salidas de todas las ramas se concatenaron antes de la capa

densa final.

Hipótesis y justificación

Las redes convolucionales extraen patrones locales o n-gramas característicos del SPAM, mientras que la LSTM captura la estructura global y el tono general del mensaje. El uso de múltiples tamaños de kernel permite detectar frases de distintas longitudes y combinar información local y global.

Referencias adicionales:

Kim, Y. Convolutional Neural Networks for Sentence Classification.

https://arxiv.org/abs/1408.5882

Este trabajo se utilizó como base teórica para justificar el uso de convoluciones 1D y múltiples tamaños de kernel en tareas de clasificación de texto.

Stack Overflow. Combining CNN and RNN for text classification.

https://stackoverflow.com/questions/44230635/how-to-combine-cnn-and-rnn-for-text-classification

Referencia utilizada como inspiración para el diseño de la arquitectura híbrida con ramas paralelas CNN y LSTM. Este modelo resultó ser el más equilibrado y con mejor rendimiento global.

Resultados obtenidos:

Validation MCC de 0.86932, Validation Accuracy de 0.9521.