Modelo de clasificación para detectar Spam

Francisco Calderon Fonseca[1], Alexis Monjarrez Calderón[1], Nicole Padilla Fonseca[1], Nuris Díaz Brenes[1]
jose.calderonfonseca@ucr.ac.cr,alexis.monjarrez@ucr.ac.cr,nicole.padilla@ucr.ac.cr, nuris.diaz@ucr.ca.cr

RESUMEN

El creciente volumen de mensajes no deseados, comúnmente conocidos como spam, representa un problema significativo para la productividad y la seguridad en línea ya que estos pueden ser utilizados como medio de extorsión. Por lo cual en este trabajo se propone implementar modelos de deeplearning para clasificar correctamente correos spam y no spam, utilizando arquitecturas como las redes neuronales convolucionales (CNN), Redes neuronales recurrentes (RNN), Transformers y el uso de un modelo preentrenado BERT. La CNN optimizada, resultó ser la estructura más adecuada para esta tarea, con una precisión del 99%, una pérdida global baja del 4%. Si bien este resultado es muy bueno, es importante destacar que podría estar influenciada por la relativa simplicidad del conjunto de datos.

PALABRAS CLAVE: correos no deseados, convolución, Transformers.

INTRODUCCIÓN

El término spam es un concepto muy utilizado para referirse a gran cantidad de mensajes no deseados, presentes en redes sociales, navegación por internet, mensajes de texto sms y correo electrónico, los cuales suelen ser molestos y en ocasiones ser utilizados como medio para realizar ataques fraudulentos (Belcic, 2020). Según la empresa McAfee en el 2008 se enviaron un total de 62 billones de mensajes spam en todo el mundo, lo cual destaca la importancia en la toma de medidas regulatorias.

Este mecanismo de mensajería se ha vuelto tan utilizado como medio de publicidad, que en países como los Estados Unidos existen leyes como CAN-SPAM, creada para proteger a los consumidores de correos no solicitados o engañosos cuyo propósito sea la promoción de un producto o servicio, además de otorgar al destinatario el derecho a no le sigan llevando dichos mensajes. (Federal TradeCommission, 2023).

Afortunadamente, los sistemas de correos actuales incorporan mecanismos automatizados de filtrado de spam, basándose en reglas predefinidas (Peña, s.f.), proporcionando una primera línea de defensa contra el correo no solicitado. Usualmente estos procesos implementan el uso de redes neuronales convolucionales diseñadas para la clasificación de grandes volúmenes de datos en formato texto (McClure, 2018).

[1] Estudiantes de la Universidad de Costa Rica

Bajo esta idea, este trabajo pretende implementar los conocimientos aprendidos durante el curso, con el objetivo de poner a prueba varios modelos de clasificación de texto mediante el uso de redes neuronales artificiales, ajustando diferentes parámetros en la arquitectura neuronal, así como evaluar el desempeño de los modelos propuestos sobre el conjunto de datos Spam-detection-dataset- splits.

METODOLOGÍA

Para la realización de este estudio se utilizó el conjunto de datos Spam-detection-dataset- splits de la página HuggingFace, compuesto por correos electrónicos en inglés clasificados como spam y no spam. El set de datos se encuentra dividido en tres subconjuntos, entrenamiento con 5450 datos de entrenamiento (2750 spam, 2700 no spam), el conjunto de validación cuenta con 2725 muestras (1375 de spam, 1350 no spam) y un conjunto de prueba con 2725 datos (1375 de spam, 1350 no spam).

Como primer paso en el procesamiento de los datos, se realizó una limpieza para preparar el texto y eliminar posibles sesgos en el rendimiento del modelo. Para ello se requirió eliminar caracteres especiales (emojis, [[]#%@^]) y signos de puntuación que no fueran relevantes para la tarea de clasificación. Así mismo se hizo uso de técnicas como la de stopword, la cual consiste en eliminar palabras vacías dentro del texto que no van a representar un aporte importante en la clasificación, seguidamente se utilizó la tokenización de las palabras el cual realiza una división del texto en palabras obtienen una comprensión básica de la estructura y el contexto del texto (Funch, 2021).

La metodología propuesta implica la implementación de siete arquitecturas, donde se consideran redes neuronales convolucionales(CNN), redes neuronales recurrentes (RNN) y Transformers los cuales son paralelizables y requieren significativamente menos tiempo para entrenarse (Vaswaniet al., 2023), cada una con dos variantes en donde se modificanciertos parámetros para evaluar el desempeño en cada versión.

Adicionalmente se utilizará una séptima arquitectura con el uso de modelos pre entrenados BERT, el cual consiste en un método para realizar un entrenamiento previo de representaciones del procesamiento del lenguaje natural (Nayak, 2019). Durante el proceso de entrenamiento se hace uso de Adam como optimizador, como función de pérdida entropía binaria cruzada y se evalúa el desempeño del modelo mediante métricas de prisión, AUC, precisión y recall así como una capa de salida con activación sigmoid, por ser un problema binario.

Para la primera arquitectura se hace uso de un modelo básico secuencial mediante una red CNN, en donde se propone una capa de embedding, una segunda capa de convolución 1D con 128 neuronas, y un kernel de tamaño 5x5 con una capa de activación ReLu. Se hace uso de un maxpooling con el fin de reducir la dimensionalidad de la salida convolucional. Posteriormente se continúa con una capa densa totalmente conectada con 64 neuronas y una activación ReLu, incluyendo una capa de dropout del 50% para prevenir el

sobreajuste del modelo. Este modelo es entrenado por un periodo de 30 epoch y un tamaño de lote de 32 muestras. Una segunda versión de esta arquitectura es el CNN optimizado, donde se incluye la misma arquitectura que el modelo básico, pero se cambia el entrenamiento por 50 periodos y un tamaño de lote de 64 muestras.

Como una primera propuesta de RNN, se incorpora una unidad recurrente cerrada (GRU) permitiendo la retención selectiva de la memoria y el uso de la puerta de actualización y la puerta de reinicio para resolver el problema del gradiente desvaneciente (Data ScienceTeam, 2022). Para este modelo se inicia modificando los parámetros del tamaño del vocabulario siendo esta de 7934, tomando como referencia el 95% de las palabras más frecuentes, un embedding de 100 y una longitud máxima de las secuencias de entrada de 100. Se implementa el uso de una primera capa bidireccional GRU con 64 neuronas y un dropout del 0.2, posterior a ello se agrega una segunda capa GRU de 32 neuronas y una capa densa de 64 con activación ReLu, seguido de un dropout del 0.5. Este modelo fue entrenado con 50 periodos y un tamaño de lote de 32. Igualmente se realizó un cambio de parámetros en la estructura para tener una variante mixta del modelo RNN, para esta ocasión los primeros dropout corresponden al 0.3, se utiliza una capa de memoria a corto y largo plazo (LSTM) con 32 neuronas y un tercer dropout de 0.5.

Para el modelo de transformer básico, se utilizan 4 cabezas de atención en bloque Transformer y un el uso de averagepooling con el fin de promediar la secuencia. Se utilizan dropout con valores de 0.2 y 0.5 y una capa densa de 64 neuronas con activación ReLU. Fue entrenado durante 50 épocas con un tamaño de lote de 32. En el caso de su versión avanzada se incluye 8 cabezas de atención, dos bloques adicionales de Transformer, y dropout ajustado a 0.3 y 0.4.

Finalmente, para el modelo transformers pre entrenado BERT (base uncased) adaptado para la clasificación binaria en detección de spam. Durante el entrenamiento se ajustó una tasa de aprendizaje del 0.00001, con un tamaño de lote de 16. Su entrenamiento fue de 10 periodos y la optimización fue con un decaimiento del peso 0.01 para regularización.

Para el desarrollo del código se utilizó Google Colab, el cual permite escribir y ejecutar código Python directamente desde el navegador, y el uso de las siguientes librerías TensorFlow (TensorFlow, s.f.), re (Python, 2024), string (Python, 2024), emoji (Kim, T., Wurster, K., 2024), matplotlib.pyplot (Hunter, J.,et al., 2022), seaborn (Waskom, M. L., 2021), numpy (NumPyteam, s.f.), random, nltk (*Bird, S., et al., 2009)*, pandas (McKinney,W., 2010).

RESULTADOS

Se comenzó el análisis con un modelo CNN básico, en donde se observa que al inicio hay una pérdida alta en el conjunto de entrenamiento, pero conforme avanzan los epochs disminuye llegando alrededor de 0%, por otro lado, la validación se mantiene con un comportamiento similar, en donde comienza con una pérdida más baja, pero se aprecia

una ligera tendencia creciente a partir del epoch 23aproximadamente, en donde alcanza valores por debajo del 5%, asimismo, hay una ligera diferencia entre ambos conjuntos, lo que puede sugerir presencia de sobreajuste. Con respecto a la precisión, en el entrenamiento aumenta rápidamente, alcanzando un valor de casi el 100%, de igual forma para la validación, por lo cual se puede decir que el modelo presenta un buen ajuste, ya que posee pérdida baja y precisión muy alta (Figura 2).

Para el modelo CNN optimizado, la pérdida de entrenamiento es más alta, esto se debe a que el modelo está aprendiendo, pero rápidamente disminuye, tomando un valor de casi 0%, en validación la pérdida se mantiene ligeramente constante con ciertas fluctuaciones con valores muy bajos, estabilizándose cerca al 0%, no obstante, se aprecia una pequeña diferencia en ambos conjuntos, lo cual indica que puede presentar sobreajuste. Con respecto a la precisión, el comportamiento es similar, ambos conjuntos se estabilizan en valores de casi 100% (Figura 3).

Para el caso del modelo RNN, la pérdida en entrenamiento comienza alta, sin embargo, conforme avanzan los epochs la pérdida disminuye rápidamente y se estabiliza aproximadamente en 0%. La pérdida en validación sigue un comportamiento similar, con ciertas fluctuaciones, pero en general, ambas pérdidas son muy bajas. En cuanto a la precisión, al inicio, en el entrenamiento es baja, porque como se ha mencionado anteriormente, el modelo está aprendiendo, seguidamente aumenta rápidamente, alcanzado casi un 100%. Para validación, se presentan varias fluctuaciones, pero se mantiene constante en valores de más del 95%. En síntesis, el modelo presenta buen ajuste, porque la pérdida es baja y la precisión muy alta (Figura 4).

El modelo RNN mixto, el conjunto de entrenamiento y validación se estabilizan en un valor de pérdida baja, sin embargo, hay una ligera diferencia entre ambas curvas, lo cual puede ser un indicio de un posible sobreajuste. En relación con la precisión, ambos conjuntos alcanzan casi el 100%. Lo cual sugiere un buen ajuste del modelo (Figura 5).

El modelo de Transformers básico presenta un comportamiento parecido a los otros modelos, se observa una pérdida baja tanto para entrenamiento como para validación, en este caso, el conjunto de validación presenta ciertas fluctuaciones, pero ambas curvas se estabilizan alrededor de un 0%. La precisión para este modelo en los primeros epochs de entrenamiento, comienza con un valor bajo mientras el modelo aprende, pero conforme avanzan los epochs, aumenta rápidamente llegando aproximadamente al 100%, y para validación, comienza con algunas fluctuaciones, pero cuando se estabiliza ambos conjuntos poseen prácticamente la misma precisión a partir del epoch 20 aproximadamente (Figura 6). Esto sugiere que el modelo tiene buena precisión para clasificar los correos de spam y no spam, ya que posee pérdida muy baja y precisión muy alta.

Para el modelo Transformers avanzado, conforme avanzan los epoch la pérdida va disminuyendo, alcanzando valores alrededor del 0% para ambos conjuntos. El comportamiento de la precisión en entrenamiento comienza con valores alrededor del 65%,

rápidamente crece estabilizando alrededor del 100%. Para el conjunto de validación, la precisión presenta fluctuaciones, pero siempre en valores muy altos, finalmente cuando se estabiliza alcanza aproximadamente el 100%, como en el conjunto de entrenamiento.

Para el modelo pre entrenado Transformers, la pérdida en entrenamiento es nula, y para validación de alrededor de un 0,004 lo cual es un valor muy pequeño, casi nulo, sin embargo, por la escala del gráfico puede parecer que exista una diferencia amplia. La precisión para el conjunto de entrenamiento es bastante alta, al estabilizarse alcanza valores cercanos al 100%.

Las métricas de rendimiento de los modelos analizados se presentan en el Cuadro 1, donde se evidencia que únicamente el modelo CNN básico muestra una menor precisión, mayor pérdida y un valor de AUC más bajo. En contraste, los demás modelos alcanzan una precisión y un AUC del 100%, junto con una pérdida del 0%. Por esta razón, se decidió evaluar las métricas en el conjunto de validación, además de comparar los falsos positivos y falsos negativos, con el objetivo de seleccionar el modelo más adecuado.

Cuadro 1

Métricas de rendimiento de los distintos modelos analizados para el conjunto de entrenamiento

Modelos	Precisión global	Pérdida global	AUC
CNN básico	0,971	0,19	0,95
CNN optimizado	1,00	0,00	1,00
RNN	1,00	0,00	1,00
RNN (mixto)	1,00	0,00	1,00
Transformers básico	1,00	0,00	1,00
Transformers avanzado	1,00	0,00	1,00
Transformers pre entrenado	1,00	0,00	1,00

En relación con los falsos positivos y falsos negativos (ver Cuadro 2) de los modelos evaluados, se observa que el modelo RNN no presenta falsos positivos, sin embargo, es el modelo con la mayor cantidad de falsos negativos (41 casos). En contraste, el modelo de Transformers preentrenado muestra mejores métricas, con solo 1 falso positivo y 4 falsos negativos. Le sigue CNN optimizado, que presenta 5 falsos positivos y 2 falsos negativos. En este contexto, es preferible clasificar incorrectamente un correo de spam como no spam, ya que caso contrario, podría resultar en la pérdida de correos importantes al ser identificados incorrectamente como spam.

Cuadro 2

Falsos positivos y negativos de los distintos modelos analizados

Modelos	Falsos Positivos	Falsos Negativos
CNN básico	8	2
CNN optimizado	**5**	**2**
RNN	**0**	**41**
RNN (mixto)	2	12
Transformers básico	2	7
Transformers avanzado	8	8
Transformers pre entrenado	**1**	**4**

Los resultados obtenidos en el conjunto de testeo muestran que todos los modelos analizados tienen un rendimiento sobresaliente, con métricas de precisión global y AUC cercanas a 0.99, lo que indica una capacidad casi perfecta para clasificar los datos no vistos. Además, las diferencias en la pérdida global entre los modelos son mínimas, lo que sugiere que todos son igualmente efectivos para la clasificación de spam. Por lo tanto, depende de factores como la simplicidad de implementación, el tiempo de entrenamiento o la disponibilidad de recursos computacionales, ya que el rendimiento final es prácticamente equivalente entre las distintas arquitecturas evaluadas (Cuadro 3).

Cuadro 3

Métricas de rendimiento de los distintos modelos analizados para el conjunto de validación

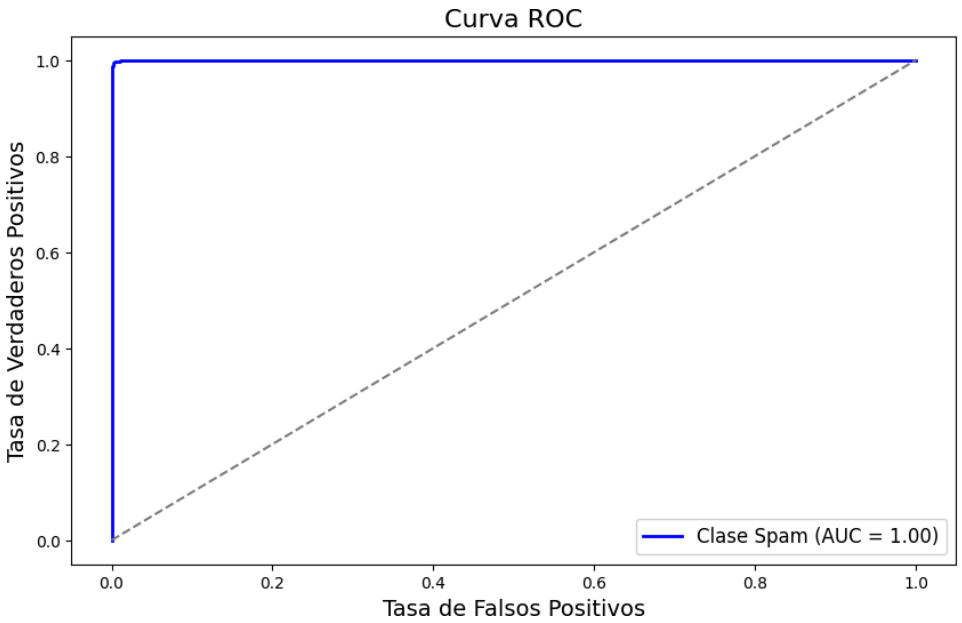
Modelos	Precisión global	Pérdida global	AUC
CNN básico	0,99	0,03	0,99
CNN optimizado	0,99	0,04	0,99
RNN	0,98	0,08	0,99
RNN (mixto)	0,99	0,05	0,99
Transformers básico	0,99	0,02	0,99
Transformers avanzado	0,99	0,03	0,99
Transformers pre entrenado	0,99	0,01	0,99

En resumen, aunque el modelo Transformer pre entrenado presenta la menor pérdida global y una cantidad reducida de falsos positivos y negativos, su complejidad computacional es considerable. Por esta razón, se selecciona como modelo ganador al CNN optimizado. Este modelo posee una precisión igualmente alta del 99%, una pérdida global

baja del 4%, y presenta solo 5 falsos positivos y 2 falsos negativos. Además, requiere menos poder computacional y clasifica de manera eficiente los correos de spam y no spam.

Figura 1

Curva ROC del modelo CNN optimizado para la clasificación de Spam



Al analizar los textos clasificados como SPAM por el modelo CNN optimizado se caracterizan por un uso excesivo de términos relacionados con dinero y promociones, como "money" y "exclusive discount", así como frases diseñadas para generar urgencia, como "click link" o "getrich." Por otro lado, los textos clasificados como NO SPAM destacan por su lenguaje técnico e informativo, con términos como "database" y "businessintelligence," y por la ausencia de llamados a la acción agresivos, reflejando un enfoque más educativo o profesional. Estas características permitieron al modelo diferenciar eficazmente entre contenido promocional y técnico (Cuadro 4).

CONCLUSIÓN

Los modelos evaluados (incluyendo CNN, RNN y Transformers) presentan un rendimiento sobresaliente para la tarea de detección de spam, con métricas de precisión y AUC cercanas al 99%. Esto evidencia que cualquiera de estas arquitecturas podría ser adecuada para detectar un correo electrónico como spam. Estudios recientes también respaldan estas conclusiones, como el de Govindan et al. (2023), quienes compararon diferentes arquitecturas de aprendizaje profundo, mostrando que las RNN lograron la mayor precisión de 98.36% al capturar eficientemente las dependencias temporales en los datos de texto.

El modelo basado en Transformers sobresale ligeramente al mostrar la menor pérdida global, lo que destaca su capacidad para capturar relaciones complejas en el texto y su superioridad en términos de precisión absoluta. Este hallazgo concuerda con estudios

recientes, como el modelo cybert79 basado en BERT, que alcanzó un 99.55% de precisión en clasificación de spam con únicamente 3 epochs (cybert79, n.d.). Sin embargo, considerando factores como la simplicidad de implementación y la eficiencia computacional, el CNN optimizado resulta ser la opción más equilibrada, aunque los modelos Transformers demuestran ser altamente efectivos, la elección final del modelo dependerá del balance entre precisión, capacidad computacional.

El conjunto de datos utilizado en este análisis presentó una estructura relativamente sencilla, con mensajes de texto cortos que tenían un promedio de 68 palabras. Este nivel de simplicidad facilitó que los modelos lograran altos niveles de precisión, ya que las características de los textos eran fácilmente distinguibles como spam o no spam. Por estas razones, los resultados obtenidos podrían sobreestimar el rendimiento de los modelos en entornos reales. Para futuros estudios, sería recomendable utilizar conjuntos de datos más complejos y representativos, incluyendo textos más extensos, diversos y que reflejen los correos electrónicos actuales, los cuales suelen combinar texto con imágenes y otros elementos. Estas mejoras permitirían evaluar la robustez de los modelos y desarrollar soluciones más efectivas para escenarios reales.

ANEXOS

Figura 2

Pérdida y precisión del modelo CNN básico

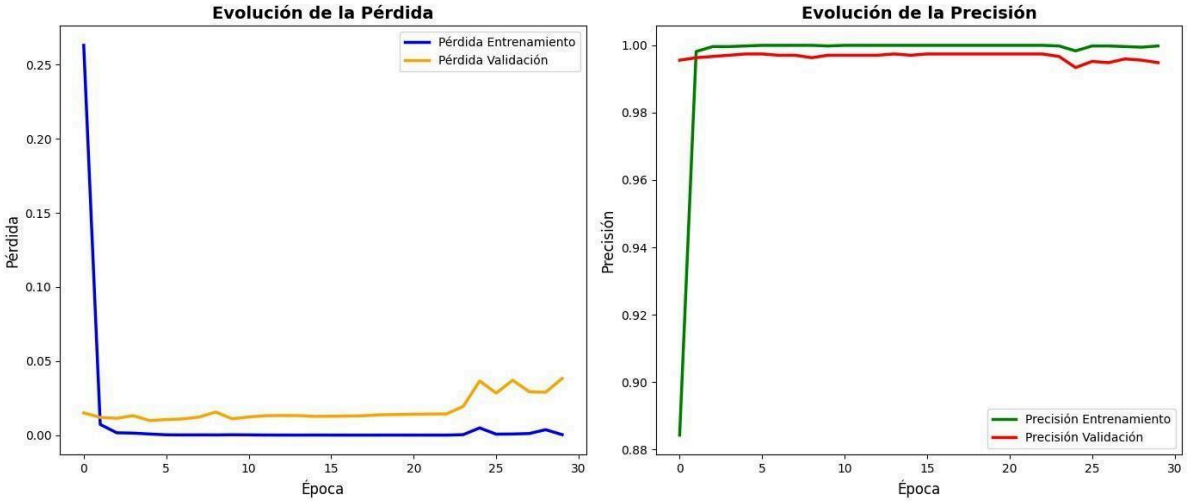


Figura 3

Pérdida y precisión del modelo CNN optimizado

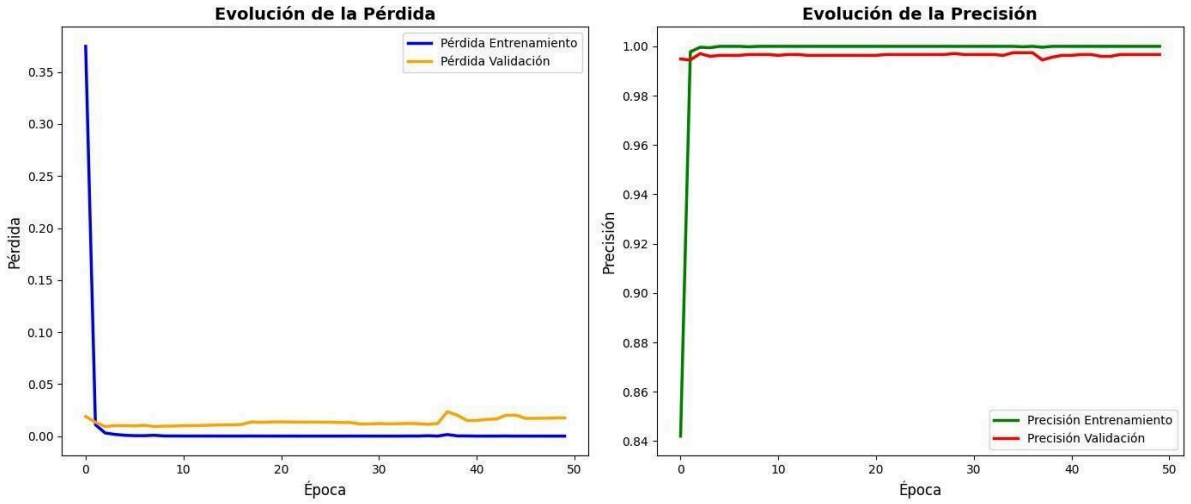


Figura 4

Pérdida y precisión del modelo RNN

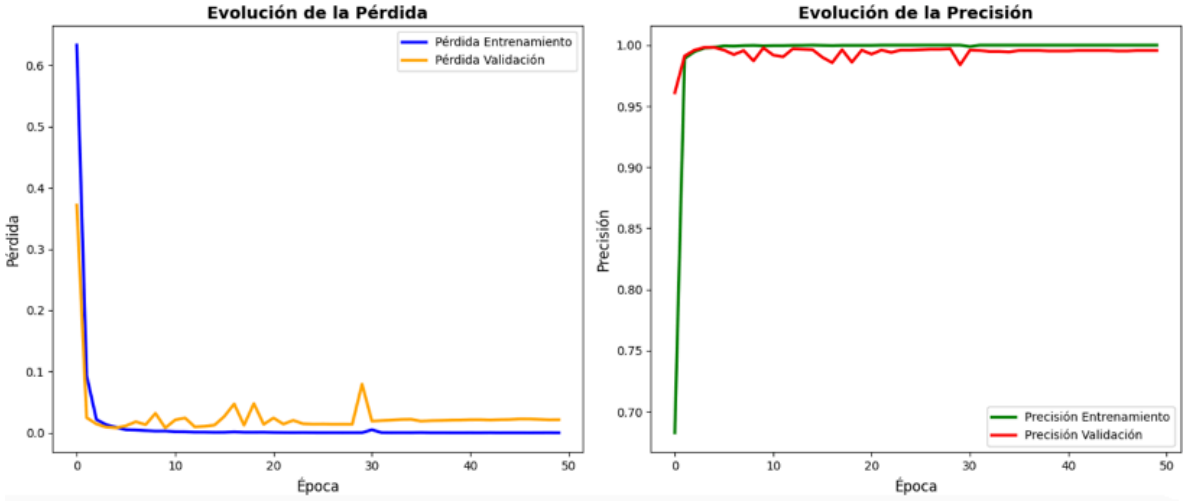


Figura 5

Pérdida y precisión del modelo RNN mixto

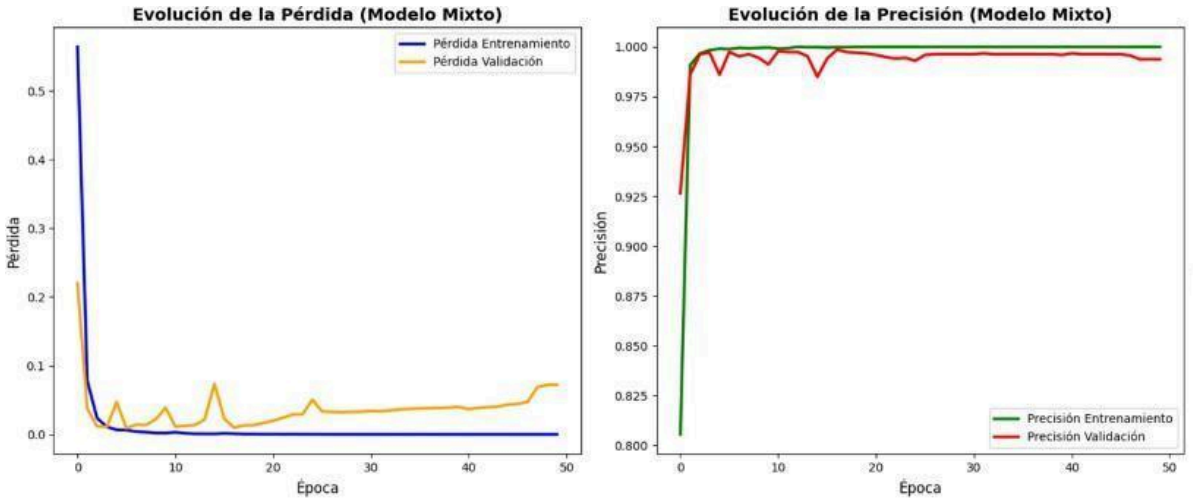


Figura 6

Pérdida y precisión del modelo Transformers básico

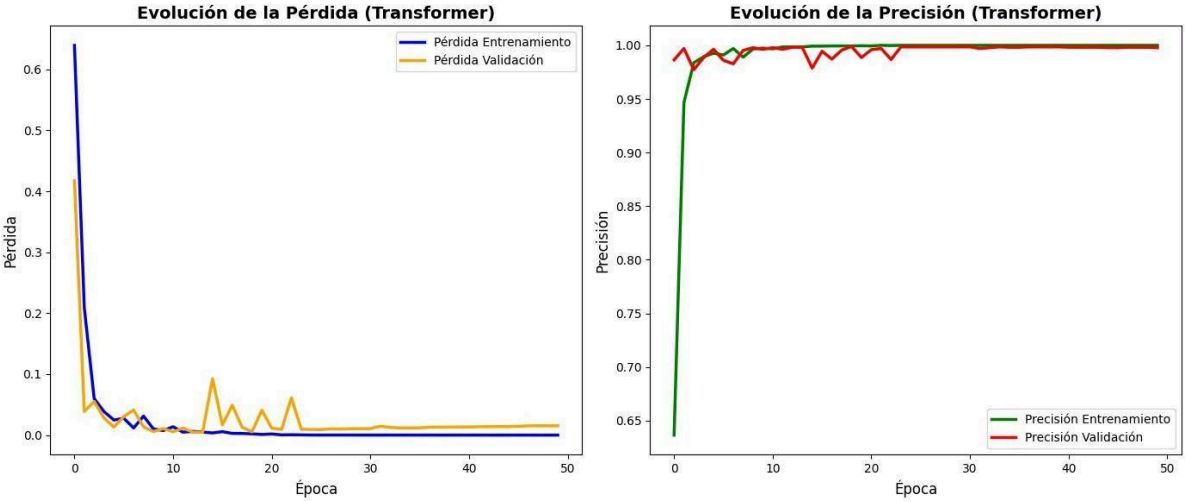


Figura 7

Pérdida y precisión del modelo Transformers avanzado

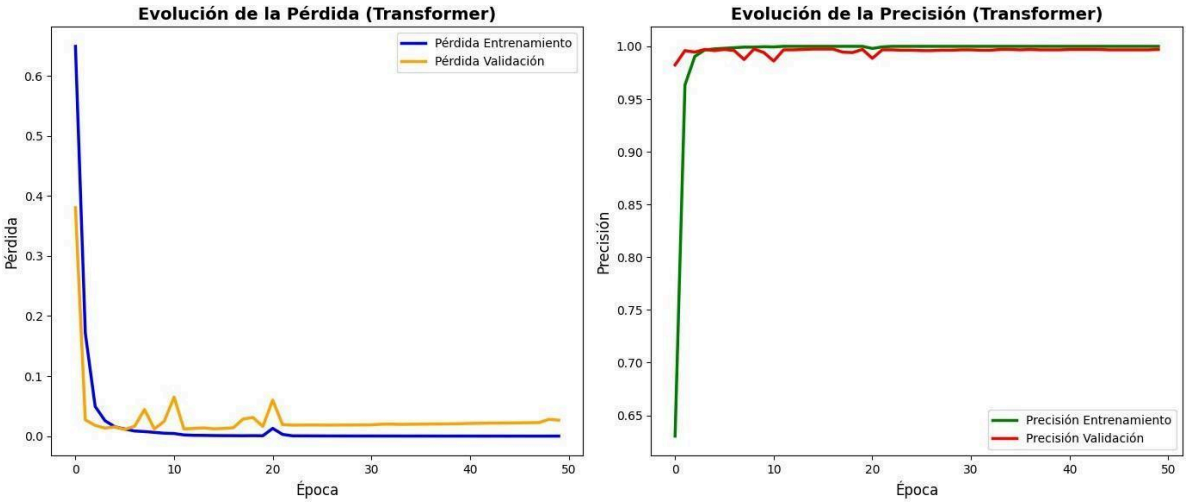
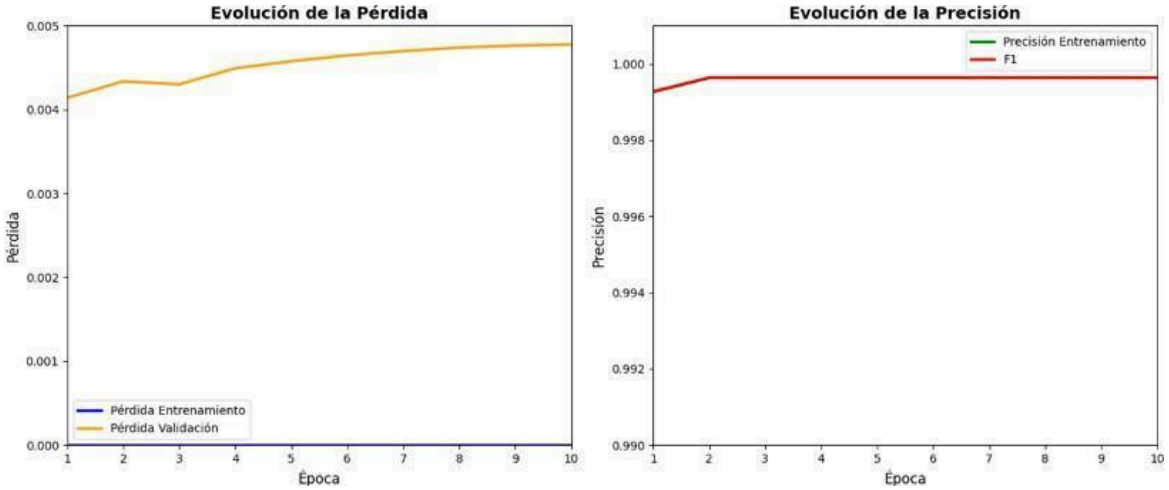


Figura 8

Pérdida y precisión del modelo pre entrenado Transformers



Cuadro 4

Clasificación de textos, modelo CNN optimizado

Clasificación	Texto
SPAM	get riiiiichquick money bag money bag rocket rocketrocketjoinamaziiiing network money makers money mouth face money mouth face backhand index pointing right backhand index pointing right backhand index pointing right 9 5 grind sit back watch cash roll money mouth face money mouth face money mouth face money mouth face dollar banknote dollar banknote dollar banknote raising hands raising hands raising hands wait sign make dreams come true money wings money wings money wings
SPAM	tired lame boring want stand coolest kid block need join social network community got latest trends memes platform perfect way show unique style personality
SPAM	woopwoop*u* believe neither got super mega offer *u* guys click link *u* get exclusive discount 99 products sounds super duper cool right
NO SPAM	database class final project design create database randomize data id like find real database base ideas please thank.
NO SPAM	probably going use linode instance spare anyone suggestion way downloadlarge definitions large would appreciated amount reddit comments inspired work like try hand analysis using different format getipbanned.
NO SPAM	hi guys anyone know get datasets details much money area makes per year mostly interested much business intelligence projects generate per year get details areas would helpful checked kaggle looks like nothing like

*Palabras detectadas como spam

*Palabras detectadas no como spam

BIBLIOGRAFÍA

Belcic. I. (08 de enero de 2020). *Qué es el spam: guía esencial para detectar y prevenir el spam*. https://www.avast.com/es-es/c-spam#topic-1

Bird, S., Loper, E., Klein, E. (2009). Natural Language Processing with Python. O'Reilly Media Inc. https://www.nltk.org/

Cybert79. (n.d.). *spamai*. Hugging Face. https://huggingface.co/cybert79/spamai

Data Science Team. (2022). *Unidades Recurrentes Cerradas.* https://datascience.eu/es/aprendizaje-automatico/unidades-recurrentes-cerradas/#google_vignette

Federal Trade Commission. (agosto, 2023). *CAN-SPAM Act: A Compliance Guide for Business.* https://www.ftc.gov/business-guidance/resources/can-spam-act-compliance-guide-business

Funch, M. (25, mayo del 2021). *PNL - Preprocesamiento de texto II (Tokenización y palabras vacías).* https://michael-fuchs-python.netlify.app/2021/05/25/nlp-text-pre-processing-ii-tokenization-and-stop-words/

Govindan, S., Abidin, A. F. A., Mohamed, M. A., Mohd Satar, S. D., & Abdul Kadir, M. F. (2023). Spam detection model using TensorFlow and deep learning algorithm. *Malaysian Journal of Computing and Applied Mathematics,* *6*(2), 11–21. https://doi.org/10.37231/myjcam.2023.6.2.84

Hugging Face. *Spam-detection-dataset-splits.* https://huggingface.co/datasets/tanquangduong/spam-detection-dataset-splits

Hunter, J., Dale, D. Firing, E., Droettboom, M. (2022). *Matplotlib.pyplot.* https://matplotlib.org/3.5.3/api/_as_gen/matplotlib.pyplot.html

Kim, T., Wurster, K. (04 de octubre, 2024).Emojifor Python. https://pypi.org/project/emoji/

Peña, D. (s.f). Convergencia en los modelos de regulación del correo electrónico no solicitado (spamming). https://bdigital.uexternado.edu.co/server/api/core/bitstreams/d4530d4b-81dc-4047-800b-25c44fa207a2/content

MCAFEE y ICF INTERNATIONAL, F. (2009). The Carbon Footprint of Email Spam Report. http://img.en25.com/Web/McAfee/CarbonFootprint_web_final2.pdf

McClure, N. (2018). TensorFlow Machine Learning. Pag 294. https://www.google.co.cr/books/edition/TensorFlow_Machine_Learning_Cookbook/TPFsDwAAQBAJ?hl=es-419&gbpv=1&dq=rnn+y+el+spam&pg=PA296&printsec=frontcover

Nayak, P. (25 de octubre, 2019). *Entender las búsquedas mejor que nunca.* https://blog.google/products/search/search-language-understanding-bert/

NumPy team. (s.f). *NumPy documentation*. https://numpy.org/doc/stable/

Python. (2024). *re — Regular expression operations.* https://docs.python.org/es/3/library/re.html#

Python. (2024). S*tring- Common string operations*. https://docs.python.org/es/3/library/string.html

TensorFlow. (s.f.). *TensorFlow: An open source machine learning framework*. TensorFlow. https://www.tensorflow.org

Vaswani, A., Shazeer,N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A., Kaiser, L. (02 de agosto , 2023). Attention Is All You Need. https://arxiv.org/pdf/1706.03762

Waskom, M. L., (2021). *seaborn: statistical data visualization*. Journal of Open Source Software, 6 (60), 3021. https://doi.org/10.21105/joss.