Práctica de Detección de Rostros Reales y Falsos con Redes Neuronales Convolucionales (CNN) en PyTorch

Susana Suárez Mendoza y Mara Pareja del Pino[1]

[1] *Universidad de Las Palmas de Gran Canaria, Grado en Ciencia e Ingeniería de Datos*
[1] *Las Palmas, Canarias, 35001, Spain*

12 de enero de 2024

Este estudio aborda el desafío de desarrollar un modelo de aprendizaje automático capaz de distinguir entre caras reales y falsas. La experimentación se llevó a cabo utilizando Python y abarcó diversas técnicas, incluyendo el aumento de datos, la regularización y el aprendizaje por transferencia. Además, se empleó la plataforma Weight&Biases para visualizar los resultados en tiempo real y facilitar el monitoreo del proceso de entrenamiento. Los resultados más destacados de esta investigación muestran un impresionante nivel de precisión con un accuracy del 81 % en el conjunto de prueba. Además, se obtuvo una sólida curva ROC con un área bajo la curva (AUC) de 0.91. Sin embargo, se observó que el principal desafío reside en la tendencia de la red a clasificar erróneamente caras falsas como reales. En resumen, este estudio representa un esfuerzo significativo que ha logrado desarrollar un modelo efectivo para abordar la problemática de distinguir entre caras reales y falsas.

Palabras clave: CNN, detección de rostros falsos y reales, Pytorch, Medidas de regularización, aprendizaje por transferencia

1. INTRODUCCIÓN

La detección de rostros ha emergido como una tecnología esencial en el ámbito de la visión por computadora y la inteligencia artificial. Esta práctica se enfoca en la detección de rostros reales y falsos mediante el uso de Redes Neuronales Convolucionales (CNN) implementadas en el marco de trabajo PyTorch. Antes de adentrarnos en los detalles de la metodología y los experimentos, es importante contextualizar la relevancia y la motivación detrás de esta tarea, así como explorar brevemente los avances recientes en este campo.

En la actualidad, la autenticación facial se ha convertido en una herramienta fundamental, utilizada en dispositivos móviles, sistemas de seguridad y aplicaciones de identificación personal. La capacidad de distinguir entre rostros reales y falsificados es crucial para garantizar la integridad de estos sistemas y proteger la privacidad de los individuos. La detección de rostros se enfrenta a desafíos significativos, ya que los falsificadores utilizan técnicas cada vez más sofisticadas, como la impresión en 3D y la generación de imágenes de alta calidad. Este escenario hace que el desarrollo de modelos de detección precisos y confiables sea una prioridad en el campo de la seguridad.

Este documento presenta una práctica integral que se centra en la detección de rostros reales y falsos mediante PyTorch, abarcando todas las fases del proceso, desde la preparación de los datos hasta la evaluación del modelo. Además, se exploran enfoques avanzados, como diferentes arquitecturas de CNN y estrategias de aumento de datos.

2. METHODOLOGÍA

Con el propósito de abordar eficazmente el problema planteado, se ha seguido una metodología sistemática que es esencial tanto para científicos de datos como para desarrolladores de modelos de aprendizaje automático. Esta metodología se ha dividido en etapas claramente definidas, cada una de las cuales desempeña un papel crucial en la construcción y evaluación del modelo de detección de rostros reales y falsos basado en Redes Neuronales Convolucionales (CNN) en PyTorch. Cada etapa se ejecuta de manera organizada y con objetivos específicos para lograr un resultado final preciso y confiable en la tarea de detección de rostros.

- **Preprocesamiento de Datos:** El preprocesamiento de datos desempeña un papel crítico en cualquier tarea de aprendizaje automático.En esta fase inicial, se descarga y descomprime el conjunto de datos, lo que permite acceder a las imágenes que se utilizarán para entrenar y evaluar el modelo.Luego, se procede a dividir los datos en conjuntos de entrenamiento y prueba, una práctica esencial para poder entrenar el modelo y, al mismo tiempo, medir su rendimiento de manera efectiva en un conjunto de datos independiente. Pero quizás uno de los aspectos más cruciales en este proceso es la normalización de las imágenes y su

conversión en tensores. Esto se hace para asegurarse de que todas las imágenes tengan la misma escala y formato, lo que facilita que el modelo las procese de manera uniforme. La normalización también contribuye a mejorar la eficiencia del proceso de entrenamiento, ya que evita que los valores de los píxeles sean demasiado grandes o pequeños, lo que podría dificultar la convergencia del modelo.

- **Construcción del Modelo:** Aquíse procede a diseñar la arquitectura de una Red Neuronal Convolucional (CNN) que será responsable de clasificar las imágenes como auténticas o falsificadas. En esta fase, se considera la inclusión de capas completamente conectadas al final de la CNN, lo que permite realizar la clasificación final de las imágenes. Dado que la definición de una arquitectura ïdeal"puede ser un desafío, se ha llevado a cabo un proceso experimental que involucra la exploración de diversas arquitecturas con el objetivo de determinar la más adecuada para la tarea de detección de rostros. Este enfoque experimental es esencial para garantizar que el modelo logre un rendimiento óptimo en la clasificación de imágenes de rostros reales y falsificados.

- **Entrenamiento:** En esta etapa, se configura una función de pérdida y un optimizador que permiten al modelo aprender durante el proceso de entrenamiento. El modelo se entrena utilizando el conjunto de entrenamiento, ajustando sus parámetros para hacer predicciones más precisas. El rendimiento del modelo se evalúa periódicamente en el conjunto de validación después de cada época de entrenamiento y haciendo uso del método de remuestreo llamado k-folds.

- **Evaluación:** La evaluación del modelo implica medir su desempeño en un conjunto de datos independiente utilizando métricas como la precisión (Accuracy), matriz de confusión y curva ROC entre otras.

3. EXPERIMENTOS

A. Construcción de modelos desde cero

En el inicio del proceso, se propuso la construcción de modelos desde cero, explorando diversas arquitecturas con el objetivo de encontrar la configuración óptima. Estas arquitecturas incluyeron:

- **Primer modelo:** Este modelo consta de tres capas convolutivas con *MaxPooling* seguidas de tres capas completamente conectadas con función de activación ReLU.

- **Segundo modelo:** En esta variante, se emplearon dos capas convolutivas con *BatchNormalization* y *MaxPooling*, seguidas de tres capas completamente conectadas con función de activación ReLU.

- **Tercer modelo:** Esta arquitectura incorpora tres capas convolutivas con .ªveragePool,"seguidas de dos capas completamente conectadas con función de activación tangente hiperbólica.

- **Cuarto modelo:** Consiste en tres capas convolutivas con "BatchNormalizationz *AveragePool*, seguidas de tres capas completamente conectadas con función de activación ReLU.

- **Quinto modelo:** En esta variante, se utilizan tres capas convolutivas con *BatchNormalization* y *AveragePool*, seguidas de cuatro capas completamente conectadas con función de activación tangente hiperbólica.

Se entrenaron todos los modelos anteriores con los datos sin tamaños de lotes (tamaño de lote = 1). Luego, se aplicó un tamaño de lote de 32 y, posteriormente, un tamaño de lote de 64. Los resultados obtenidos se presentan en la gráfica, que muestra que el segundo modelo logró la mayor precisión. Además, se observa que la aplicación del tamaño de lote de 64 en el modelo 2 condujo a una mejora en los resultados en comparación con los tamaños de lote más pequeños.

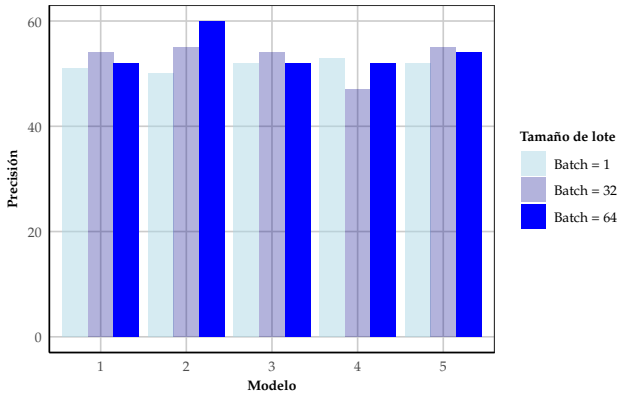


Fig. 1. Comparativa de los modelos según el tamaño de lote.

B. Función optimizadora

La función optimizadora en una red neuronal es un componente esencial del proceso de entrenamiento de la red. Su objetivo principal es ajustar los pesos y sesgos de la red de manera que la función de pérdida (loss function) se minimice, lo que lleva a una mejora en el rendimiento de la red en tareas específicas, como la clasificación o la regresión. Por tanto, se ha experimentado con las siguientes funciones de optimización, así como sus correspondientes hiperparámetros en el entrenamiento del segundo modelo destacando el optimizador Adadelta.

Cuadro 1. Experimentación del optimizador

Optimizador	Tasa de aprendizaje	Precisión (%)
Adadelta	1	62
Adadelta	0.1	60
Adadelta	0.01	66
Adadelta	0.001	45
SGD	0.1	59
SGD	0.01	59
SGD	0.001	50
RMSPop	0.1	53
RMSProp	0.001	45
Adam	0.1	55
Adam	0.01	52
AdamW	0.1	53
AdamW	0.01	56

C. Ajuste fino y aumento masivo de datos

El *ajuste fino* y el *aumento masivo de datos* son dos estrategias clave en el aprendizaje profundo. El ajuste fino implica adaptar una red pre-entrenada a una tarea específica, aprovechando el conocimiento previo de la red. Por otro lado, el aumento masivo de datos implica enriquecer artificialmente el conjunto de entrenamiento mediante transformaciones de datos, lo que mejora la capacidad del modelo para generalizar. Ambas estrategias son cruciales para mejorar el rendimiento y la generalización de los modelos de redes neuronales en tareas específicas.

La elección del las técnicas anteriores se justifica por su capacidad para aprovechar el conocimiento previo de redes pre-entrenadas y mejorar la generalización del modelo. El ajuste fino permite adaptar una red previamente entrenada a una tarea específica, lo que es esencial para ahorrar tiempo y recursos, especialmente cuando los datos son limitados. Por otro lado, el *aumento masivo de datos* enriquece el conjunto de entrenamiento mediante la aplicación de transformaciones a los datos, reduciendo el riesgo de sobreajuste y aumentando la diversidad de ejemplos. Combinar ambas estrategias mejora la robustez y la capacidad de adaptación del modelo, lo que es fundamental para construir modelos de aprendizaje profundo efectivos y robustos en una variedad de aplicaciones.

En un principio, se optó por realizar el proceso de *ajuste fino* en modelos pre-entrenados, específicamente en AlexNet, ResNet18 y ResNet34. Estos modelos han sido previamente entrenados para extraer características abstractas de imágenes y clasificar entre 1000 objetos diferentes. Luego, haciendo uso de la librería facenet-pytorch se intentó reentrenar los modelos entrenados para el reconocimiento de caras, aunque esto no obtuvo mejora.

Finalmente, se decidió utilizar el modelo ResNet18 debido a que logró una precisión de validación del 74 % durante el proceso de entrenamiento, lo que lo convirtió en la elección óptima para la tarea de detección de rostros reales y falsos.

D. Técnicas de regularización aplicadas.

Durante el proceso de reentrenamiento del modelo ResNet18, se identificó un problema de sobreajuste. Este fenómeno se manifestó con un aumento en la precisión en los datos de entrenamiento a medida que el modelo se ajustaba más a estos datos, pero la precisión en los datos de validación disminuía. Esto sugiere que el modelo estaba memorizando los datos de entrenamiento en lugar de generalizar patrones, lo que podría afectar la precisión en nuevos datos. El sobreajuste es común en el aprendizaje automático, y su mitigación implica la aplicación de técnicas de regularización y la selección cuidadosa de hiperparámetros.

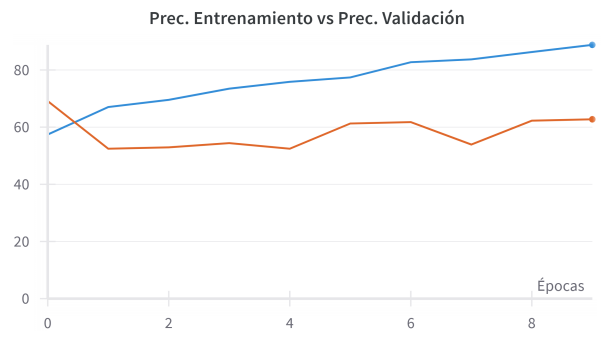


Fig. 2. Fenómeno del sobreajuste.

Es en esta gráfica donde nos damos cuenta que mientras la precisión de entrenamiento aumenta (línea azul) , la precisión de validación disminuye (línea naranja). Por tanto, con tal de evitar el fenómeno se ha optado por lo siguiente:

1. **Regularización L2.** La regularización L2 implica agregar un término de penalización a la función de pérdida del modelo que es proporcional a la suma de los cuadrados de todos los pesos del modelo. Esto tiene el efecto de penalizar los valores de peso grandes y, en última instancia, puede ayudar a prevenir el sobreajuste al mantener los pesos del modelo en un rango más pequeño.

2. **Dropout.** Consiste en apagar (o *abandonar*) de manera aleatoria una fracción de las unidades (neuronas) en una capa durante el entrenamiento. En otras palabras, durante cada paso de entrenamiento, algunas neuronas se excluyen temporalmente de la red con una probabilidad dada

3. **K-folds.** La idea principal detrás de K-Fold Cross-Validation es dividir el conjunto de datos en K particiones o *pliegues* aproximadamente del mismo tamaño. Luego, se entrena y evalúa el modelo K veces, utilizando K-1 de los pliegues como datos de entrenamiento y el pliegue restante como datos de prueba en cada iteración. Al evaluar el modelo en múltiples particiones de datos de prueba, se obtiene una estimación más confiable de cómo se generalizará a datos nuevos y no vistos previamente. Esto evita que el modelo se ajuste demasiado a un conjunto particular de datos de entrenamiento.

E. Utilización de la plataforma "Weights & Biases".

WandB es una plataforma de seguimiento y visualización de experimentos diseñada para rastrear y gestionar los proyectos de manera más efectiva. Wandb ofrece una serie de características y herramientas que facilitan el proceso de entrenamiento y experimentación con modelos de aprendizaje automático, incluyendo el registro de experimentos y la visualización en tiempo real de los resultados.

4. RESULTADOS

La evaluación de modelos es esencial en el aprendizaje automático. Permite medir la capacidad de generalización de un modelo, es decir, su precisión en datos no vistos previamente. Evaluar un modelo brinda una visión objetiva de su rendimiento, crucial para decisiones en aplicaciones prácticas.

A. Precisión del conjunto de test.

El conjunto de prueba consiste en un conjunto de imágenes que la red neuronal no ha tenido la oportunidad de observar durante su proceso de entrenamiento. La evaluación de la precisión en este conjunto se rige por la siguiente fórmula:

$$\text{Precisión(Accuracy)} = \frac{\text{TP} + \text{TN}}{\text{Total de Predicciones}} \qquad \textbf{(1)}$$

Donde:

- TP (Verdadero positivos) corresponde a los casos en los que el modelo predijo correctamente una instancia positiva. Es decir, la cara falsa clasificada como falsa.

- TN (Verdaderos Negativos) corresponde a los casos en los que el modelo predijo correctamente una instancia negativa. es decir, la cara real clasificada como real.

Finalmente, la precisión de nuestro modelo, expresada en términos porcentuales, asciende al **81,95 %**. Este resultado denota una precisión altamente satisfactoria, especialmente considerando la naturaleza compleja del conjunto de datos.

B. Matriz de confusión

Una matriz de confusión muestra de manera resumida la cantidad de predicciones correctas e incorrectas realizadas por el modelo. En particular, muestra el número de verdaderos positivos (TP), verdaderos negativos (TN), falsos positivos (FP) y falsos negativos (FN). La matriz presentada en la figura 3 proporciona una representación visual de las clasificaciones realizadas por el modelo. Se observa que, en su mayoría, el modelo logra clasificar correctamente ambas clases, con excepción de 7 muestras que originalmente pertenecían a la clase caras reales y fueron erróneamente clasificadas como pertenecientes a la clase caras falsas. Además, se registra un gran error de la red en forma de caras falsas que han sido incorrectamente clasificadas como reales.

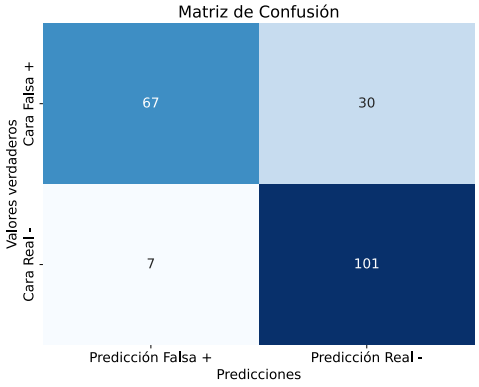


Fig. 3. Matriz de confusión para el conjunto de Test.

C. Visualización de predicciones erróneas

Se han detectado errores en la red neuronal para ambas clases, lo que ha llevado a un análisis detallado de casos específicos para una mejor comprensión.

Caras reales predichas como falsas

La matriz de confusión previamente presentada (Figura 3) ha revelado que se produjeron siete instancias en las que caras reales fueron incorrectamente clasificadas como falsas por el modelo. Un análisis más detallado de dos de estas imágenes específicas revela que ciertas características, como el maquillaje o la apariencia de una piel perfecta, podrían haber sido factores contribuyentes a esta clasificación errónea por parte de la red.

Fig. 4. Ejemplos de caras reales clasificadas como falsas.

Caras falsas predichas como reales

Un dato de mayor relevancia que se desprende de nuestro análisis es la significativa proporción de errores cometidos por nuestra red al clasificar caras falsas como reales. Al examinar detenidamente las dos imágenes inferiores, resulta evidente que la imagen de la izquierda es claramente falsa, lo que sugiere que la normalización aplicada a nuestras imágenes podría estar contribuyendo a estas confusiones. No obstante, la imagen de la derecha presenta un desafío incluso para la percepción humana, ya que la apariencia facial resulta extremadamente realista y difícil de discernir como falsa.

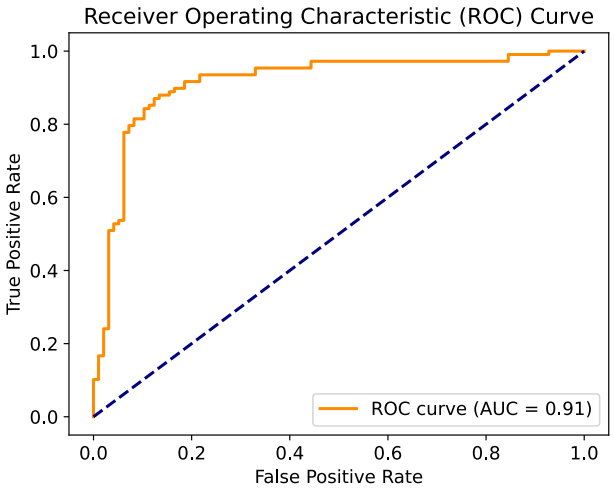
Fig. 5. Ejemplos de caras falsas clasificadas como reales.

D. Curva ROC

La curva ROC (Receiver Operating Characteristic) muestra cómo cambia la tasa de verdaderos positivos (sensibilidad) en función de la tasa de falsos positivos (1 - especificidad) a medida que varía el umbral de decisión del modelo. Cada punto en la curva ROC representa un umbral de decisión diferente que se aplica al modelo para clasificar las instancias.

Por un lado, el área bajo la curva (AUC) es de 0.91, lo que indica un excelente nivel de discriminación. Esto significa que hay una alta probabilidad de que el modelo clasifique correctamente entre las clases positivas y negativas.

Por otro lado, la curva se eleva rápidamente hacia la tasa de verdaderos positivos de 1, lo que sugiere que el modelo tiene una alta sensibilidad y puede identificar correctamente la mayoría de los casos positivos.



E. Medidas formales de eficacia de la clasificación

En esta subsección, se presentan diversas medidas formales de eficacia utilizadas para evaluar la calidad y el rendimiento de un modelo de clasificación. Estas métricas son esenciales para comprender cómo se desempeña un clasificador en la tarea de separar las clases de datos. La tabla que se muestra a continuación detalla algunas de las métricas más comunes utilizadas en la evaluación de modelos de clasificación. Cada una de estas métricas ofrece una perspectiva única sobre el desempeño del modelo, ya sea en términos de sensibilidad, especificidad, precisión, exactitud, F-Measure, correlación de Matthews y otras medidas de similitud.

Métrica	Valor
Sensibilidad (TPR)	0.935185
FPR	0.309278
Especificidad	0.690722
Predictivo Positivo (PPV)	0.770992
Predictivo Negativo (NPV)	0.905405
F-Measure	0.845188
Correlación de Matthews	0.650663
Relación Positiva de Similitud	3.023765
Relación Negativa de Similitud	0.093836

Cuadro 2. Medidas de eficacia de la clasificación

De la tabla anterior podemos sacar las siguientes conclusiones:

Sensibilidad (TPR). Con un valor de 0.935185, esta métrica indica que el modelo tiene una alta capacidad para identificar verdaderos positivos en comparación con el total de casos positivos. Esto sugiere que el modelo es efectivo para detectar falsas cuando realmente lo son.

FPR (False Positive Rate). Con un valor de 0.309278, esta métrica mide la tasa de falsos positivos (. Un valor más bajo es deseable, ya que indica que el modelo comete menos errores al clasificar caras reales como falsas.

Especificidad. Con un valor de 0.690722, esta medida indica la capacidad del modelo para identificar verdaderas caras reales en comparación con el total de caras reales. Un valor más alto indica una mejor capacidad para distinguir negativos correctamente.

Predictivo Positivo (PPV). Con un valor de 0.770992, esta métrica representa la proporción de verdaderas caras falsas con respecto a todas las predicciones de caras falsas realizadas por el modelo. Un PPV alto sugiere que las predicciones positivas son confiables.

Predictivo Negativo (NPV). Con un valor de 0.905405, esta métrica indica la proporción de verdaderas caras reales con respecto a todas las predicciones de caras reales realizadas por el modelo. Un alto NPV sugiere que las predicciones negativas son confiables.

F-Measure. Con un valor de 0.845188, esta métrica combina precisión y sensibilidad en una sola medida. Un valor alto indica un equilibrio entre la capacidad de identificar positivos y la precisión de las predicciones positivas.

Correlación de Matthews. Con un valor de 0.650663, esta métrica evalúa la calidad general de las predicciones, teniendo en cuenta tanto los verdaderos positivos como los verdaderos negativos. Un valor positivo indica una relación estadística significativa entre las predicciones y las observaciones reales.

Relación Positiva de Similitud. Con un valor de 3.023765, esta métrica sugiere una fuerte similitud entre las predicciones positivas y los verdaderos positivos. Cuanto mayor sea este valor, mejor será la capacidad del modelo para identificar caras falsas.

Relación Negativa de Similitud. Con un valor de 0.093836, esta métrica indica una relación menos fuerte entre las predicciones negativas y los verdaderos negativos. Un valor más bajo puede indicar que el modelo tiene margen de mejora en la identificación de caras reales.

5. CONCLUSIONES

En este estudio, se desarrolla un modelo de clasificación de caras reales y falsas mediante técnicas de aprendizaje profundo. Las conclusiones clave incluyen un impresionante accuracy del 81 %, lo que refleja el éxito del enfoque utilizado. Además, las métricas de precisión, la curva ROC y la matriz de confusión destacan la eficacia del modelo en la detección de caras falsas. Se aprendió la importancia del aprendizaje por transferencia en situaciones con datos limitados, lo que permitió adaptar un modelo preentrenado, ahorrando tiempo y recursos.

Además, se ha adquirido una comprensión más profunda de cómo detectar y evitar el sobreajuste, una preocupación común cuando se trabajan con conjuntos de datos pequeños. Mediante la aplicación de técnicas de regularización, se pudo mejorar la generalización del modelo y evitar problemas de sobreajuste que podrían haber afectado negativamente el rendimiento.

Finalmente, se destacan como los desafíos más significativos la implementación inicial de modelos convolucionales desde cero, dado que estos requieren una cantidad considerable de datos para detectar características abstractas en las imágenes de manera efectiva. Además, se encontró una complicación en la búsqueda de técnicas de regularización adecuadas con el objetivo de mejorar el rendimiento del modelo en el conjunto de prueba, lo cual se volvió especialmente relevante al enfrentar una precisión del 60 %.

6. TRABAJO FUTURO

Como una mejora potencial para el modelo, sería recomendable explorar una variedad de arquitecturas adicionales y considerar la utilización de modelos preentrenados adicionales. Asimismo, se podría contemplar el aumento del tamaño del conjunto de datos de entrenamiento con el propósito de mejorar aún más el rendimiento del modelo.

Adicionalmente, se podría haber llevado a cabo la optimización de hiperparámetros utilizando la librería Optuna; sin embargo, es importante destacar que este enfoque probablemente habría implicado un aumento significativo en el tiempo de entrenamiento de los modelos debido a la búsqueda exhaustiva de parámetros, lo cual no fue considerado en esta etapa del estudio.

7. REFERENCIAS

- Repositorio de Github.