

Detección de Edición en Imágenes

Daniel Abad Fundora C411

Anabel Benítez González C411

Enzo Rojas D'Toste C411

Objetivos del Trabajo

1. Abordar la detección de edición en imágenes:

Enfrentar este problema crítico en la era de la desinformación digital, considerando su impacto en seguridad, periodismo y redes sociales.

2. Desarrollar y comparar enfoques de Machine Learning:

Implementar diversas técnicas de aprendizaje automático y evaluarlas en la tarea de detección de ediciones en imágenes.

3. Identificar ventajas y desventajas:

Analizar los puntos fuertes y débiles de cada método, comprendiendo sus aplicaciones y limitaciones específicas.

4. Proponer mejoras y optimizaciones:

Sugerir mejoras basadas en los resultados obtenidos y buscar optimizaciones para aumentar la precisión y eficiencia de los modelos desarrollados.

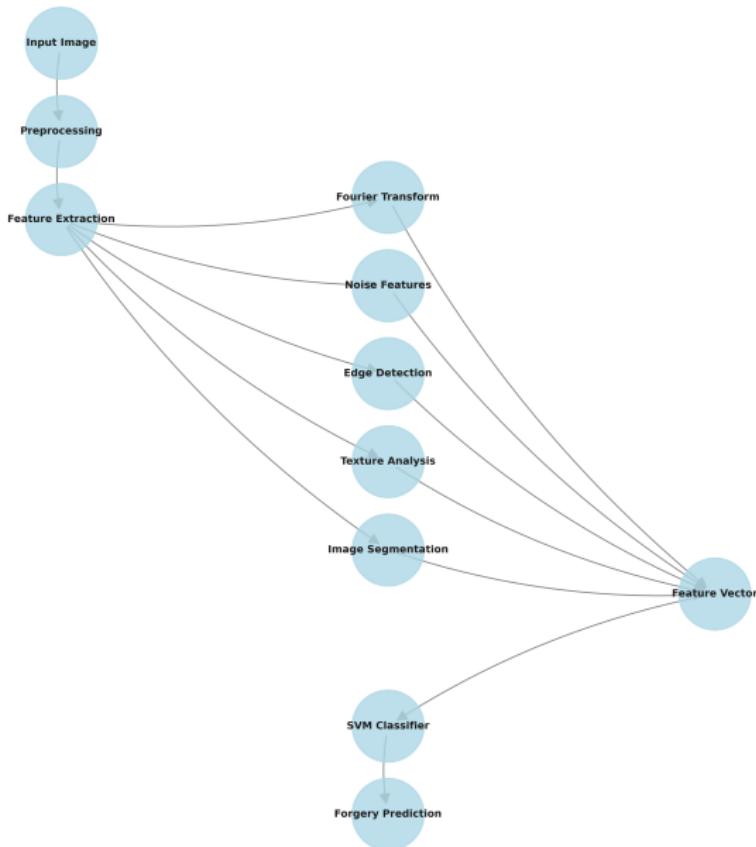
Enfoques Utilizados

- ▶ Extracción de características y utilización de un modelo SVM.
- ▶ Utilización de U-Net.
- ▶ Utilización de GAN.
- ▶ Utilización de técnicas de procesamiento de imágenes y una CNN.
- ▶ Utilización de Transfer Learning con extracción de características y Fine Tuning de un modelo preentrenado.
- ▶ Uso de Boosting para la creación de un Ensemble.

Dataset Utilizado: CASIA V2

- ▶ Variedad de Manipulaciones: Imágenes con diferentes tipos de manipulaciones.
- ▶ Volumen de Imágenes: 12,614 imágenes (7,491 auténticas y 5,123 manipuladas).
- ▶ Anotaciones Detalladas: Especifican las áreas alteradas.
- ▶ Diversidad de Escenarios: Asegura robustez y generalización de los modelos.

Entrenamiento de un modelo SVM



Extracción de Características

- ▶ Transformada de Fourier: Para obtener el espectro de magnitud de la imagen, revelando las frecuencias presentes.
- ▶ Características de Ruido: Se calculan la media y la desviación estándar del ruido en la imagen.
- ▶ Detección de Bordes: Se realiza utilizando el algoritmo de Canny.
- ▶ Características de Textura: Se utilizan matrices de co-ocurrencia de niveles de gris (GLCM) para calcular características de textura, específicamente el contraste.
- ▶ Segmentación de Imágenes: Se lleva a cabo mediante el algoritmo de k-means, dividiendo la imagen en k clústeres.

Resultados del modelo SVM

Característica Removida	Precisión (%)
Ninguna (base)	72
Transformada de Fourier	70
Ruido	72
Detección de Bordes	77
Texturas	72
Segmentación	58

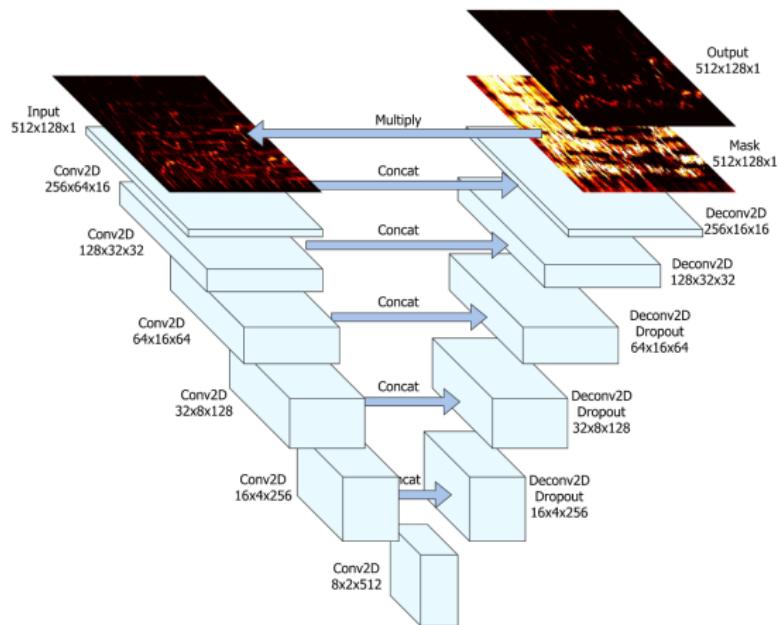
Table: Impacto de la eliminación de características individuales en la precisión del modelo SVM.

Resultados del modelo SVM

- ▶ **Prueba adicional:**
 - ▶ Modelo simplificado: solo Transformada de Fourier y Segmentación
 - ▶ Resultado: 77% de precisión
- ▶ **Hallazgos clave del análisis:**
 - ▶ Transformada de Fourier: componente crítico para la precisión
 - ▶ Segmentación: contribuye sustancialmente, pero con alta carga computacional
 - ▶ Otras características (ruido, bordes, texturas): impacto mínimo o potencialmente negativo
- ▶ **Implicaciones para la optimización:**
 - ▶ Potencial para simplificar significativamente el modelo
 - ▶ Oportunidad de mejorar eficiencia sin sacrificar efectividad

Utilización de U-Net

Las U-Nets son una clase de redes neuronales convolucionales diseñadas originalmente para tareas de segmentación de imágenes biomédicas, su arquitectura se caracteriza por una estructura en forma de "U", que consta de una etapa de contracción y una etapa de expansión.



Arquitectura U-Net: Componentes Clave

► Etapa de contracción:

- ▶ Similar a una CNN tradicional
- ▶ Capas de convolución seguidas de pooling
- ▶ Reduce resolución espacial
- ▶ Incrementa número de características detectadas

► Etapa de expansión:

- ▶ Recupera resolución espacial
- ▶ Utiliza upsampling y convolución
- ▶ Combina características de contracción y expansión
- ▶ Proporciona información más rica y detallada

► Skip Connections:

- ▶ Característica distintiva de U-Net
- ▶ Transfiere directamente características entre etapas
- ▶ Preserva información detallada
- ▶ Mejora la precisión de la segmentación

Resultados con U-Net. Limitaciones y Desafíos

- ▶ **Velocidad de entrenamiento insuficiente**
 - ▶ Restringió las pruebas a conjuntos de datos reducidos
 - ▶ Resultó en un claro problema de *overfitting*
- ▶ **Resultados insatisfactorios**
 - ▶ El rendimiento quedó por debajo de las expectativas para aplicaciones prácticas
- ▶ **Necesidad de revisión significativa**
 - ▶ El modelo, en su estado actual, requiere una revisión profunda
 - ▶ Arquitectura y métodos de entrenamiento deben ser mejorados para superar las limitaciones y alcanzar un desempeño aceptable

Utilización de GAN

Estructura de una Generative Adversarial Network

- ▶ Generador
- ▶ Discriminador

Utilización de GAN

Rol del Generador

El **generador** se entrenó para crear imágenes que parecieran lo más reales posible, con el objetivo de engañar al discriminador.

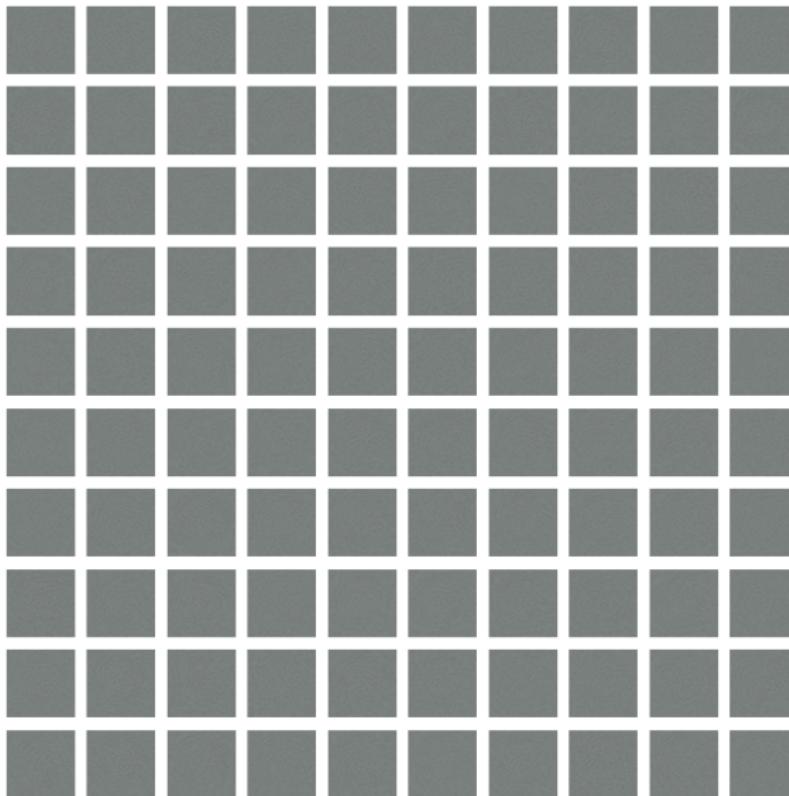
Rol del Discriminador

Por su parte, el **discriminador** se enfocó en aprender a identificar estas imágenes generadas como falsas.

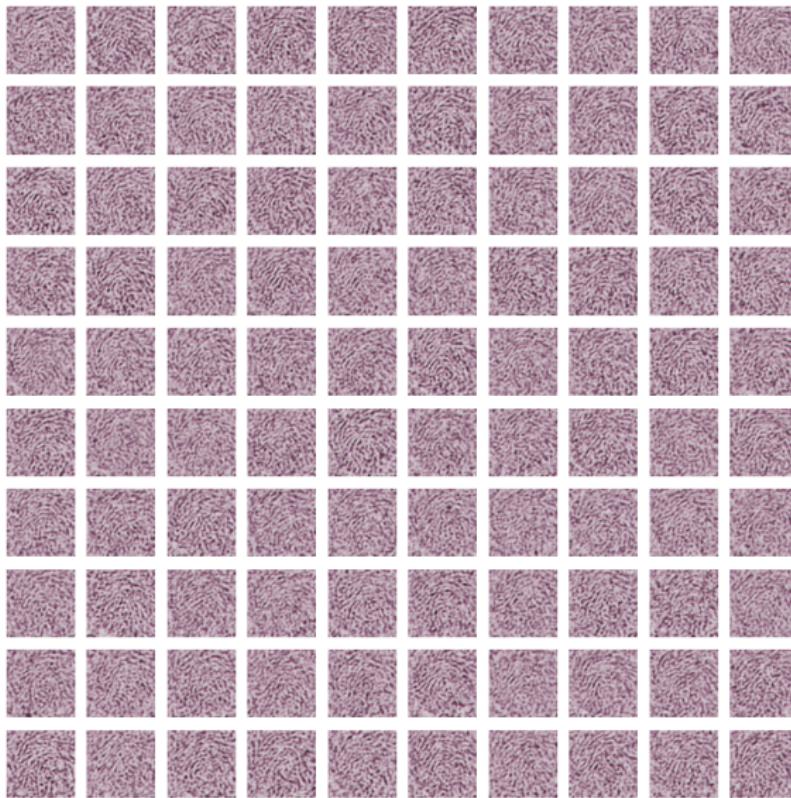
Objetivo Final

Este proceso de entrenamiento adversario tenía como propósito mejorar continuamente la capacidad del discriminador para detectar manipulaciones.

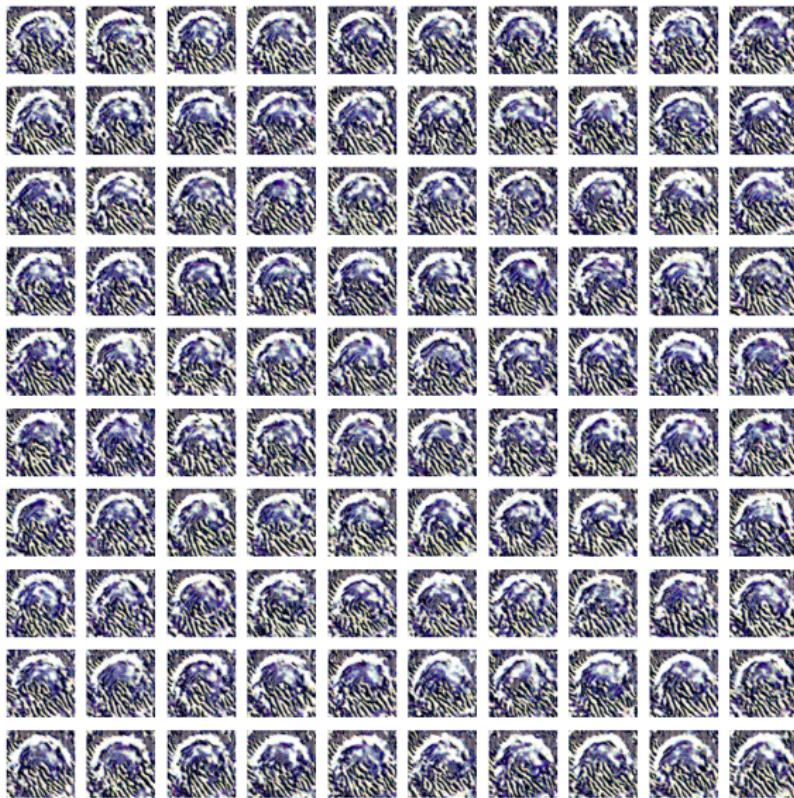
Resultados y Limitaciones del Enfoque GAN



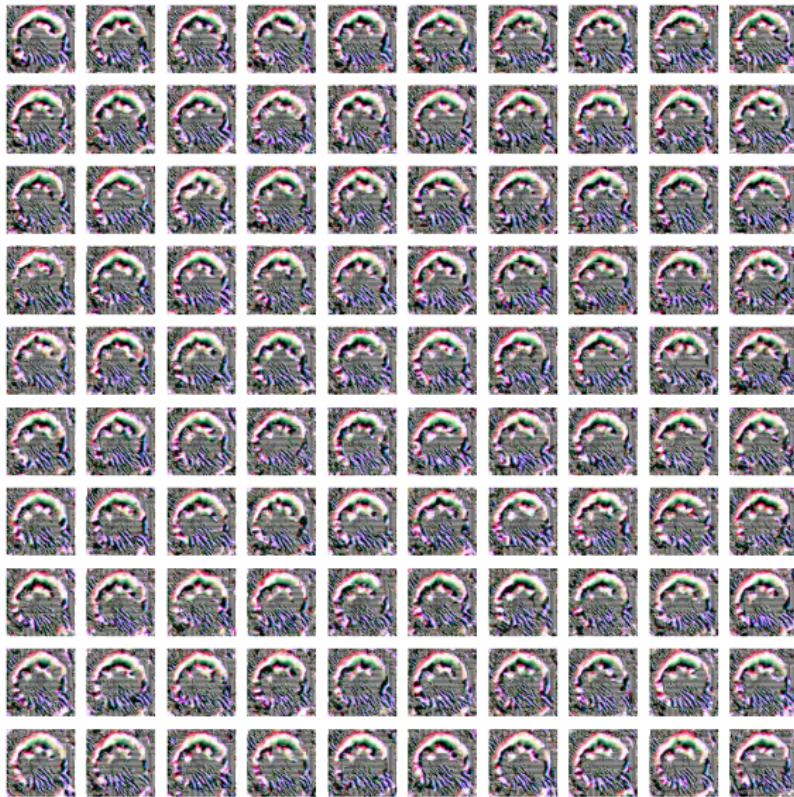
Resultados y Limitaciones del Enfoque GAN



Resultados y Limitaciones del Enfoque GAN



Resultados y Limitaciones del Enfoque GAN



Resultados y Limitaciones del Enfoque GAN

Observaciones

A pesar de observar patrones en las imágenes generadas, estos no contribuyeron a una mejora significativa en la detección de manipulaciones por parte del discriminador.

Implicaciones

Este resultado sugiere:

- ▶ Posibles limitaciones inherentes al enfoque utilizado
- ▶ Necesidad de ajustes en la arquitectura del modelo
- ▶ Potencial revisión de las estrategias de entrenamiento

Utilización de Técnicas de Procesamiento de Imágenes y una CNN

- ▶ Implementamos este modelo basados en los resultados del paper "Image Forgery Detection Using Deep Learning by Recompressing Images" (2022).
- ▶ Las redes neuronales convolucionales (CNNs) se inspiran en el sistema visual humano y son efectivas para tareas de visión por computadora.
- ▶ La falsificación de imágenes deja artefactos sutiles que las CNNs pueden detectar, aunque sean imperceptibles para el ojo humano.

Puntos clave del enfoque

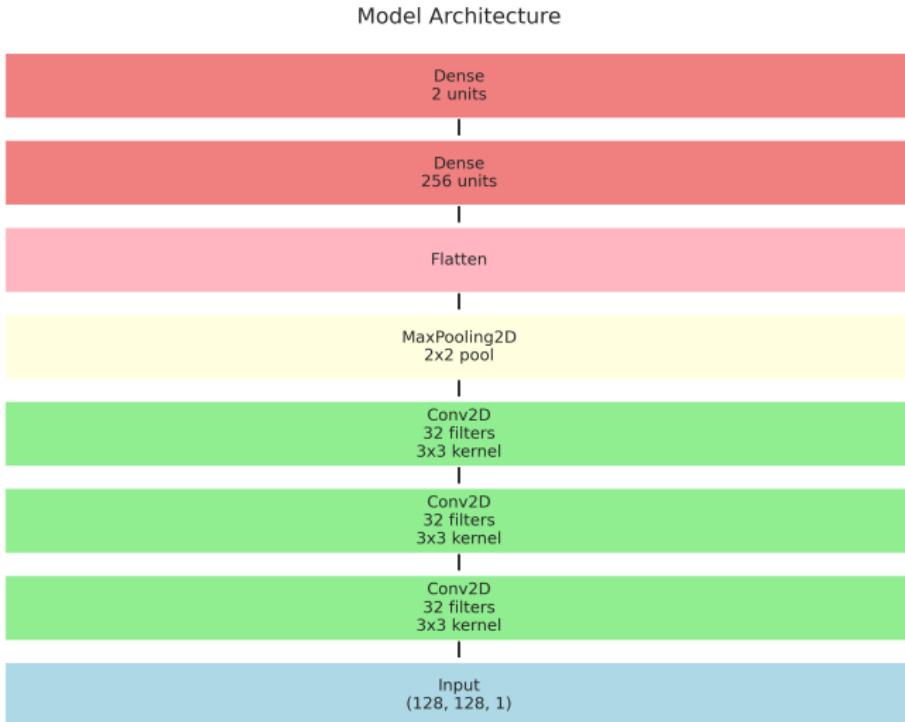


- ▶ Cuando se recomprime una imagen falsificada, la parte alterada se comprime de manera diferente al resto debido a su origen distinto.
- ▶ Se aprovecha el hecho de que las regiones falsificadas tienen una distribución estadística diferente de coeficientes DCT (Discrete Cosine Transform) comparadas con las regiones originales.

Proceso de Detección de Falsificaciones con CNN

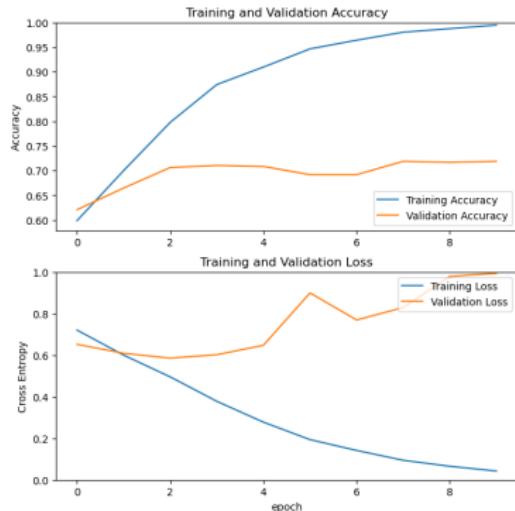
1. Se toma una imagen potencialmente falsificada
2. Se recomprime la imagen
3. Se calcula la diferencia entre la imagen original y la recomprimida
4. En esta imagen diferencia se destaca la parte falsificada
5. Se usa esta imagen como entrada para entrenar la CNN

Estructura de la CNN



Resultados con CNN

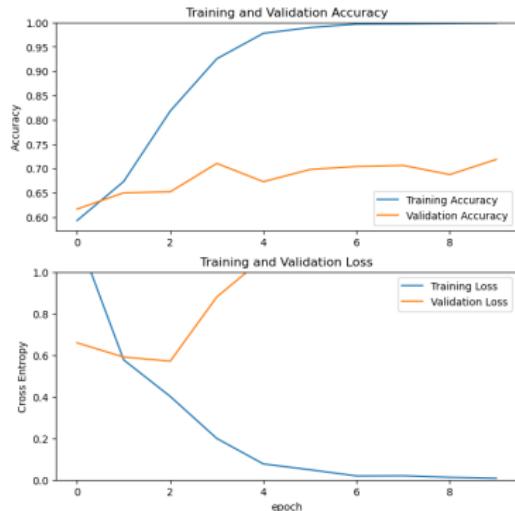
- ▶ Epochs: 10
- ▶ Calidad de compresión: 98
- ▶ Tamaño de paso: 0.0001



Metric	Value
False Positives (FP)	207.00
False Negatives (FN)	161.00
True Positives (TP)	319.00
True Negatives (TN)	513.00
Precision	0.61
Recall	0.66
F1-score	0.63
Overall Accuracy	0.69

Resultados con CNN

- ▶ Epochs: 10
- ▶ Calidad de compresión: 80
- ▶ Tamaño de paso: 0.001



Metric	Value
False Positives (FP)	192.000
False Negatives (FN)	194.000
True Positives (TP)	286.000
True Negatives (TN)	528.000
Precision	0.598
Recall	0.596
F1-score	0.597
Overall Accuracy	0.678

Resultados y desafíos del modelo

Problema de Overfitting

El modelo tiende a hacer overfitting:

- ▶ Resultados cercanos a 1 en el conjunto de entrenamiento
- ▶ Resultados mediocres en el conjunto de validación

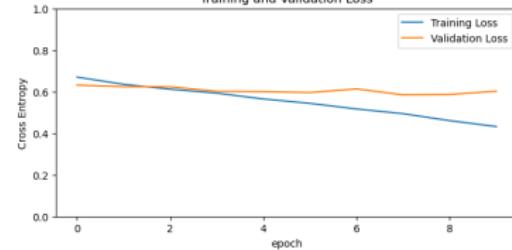
Intento de Solución

Para evitar el sobreajuste de parámetros:

- ▶ Se probó disminuir el tamaño de paso
- ▶ Se probó agregar capas de dropout

Resultados con CNN variando hiperparámetros

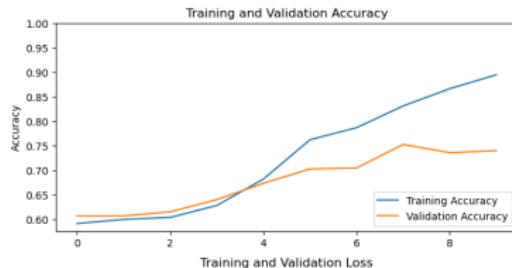
Disminuyendo el tamaño de paso a 0.00001.



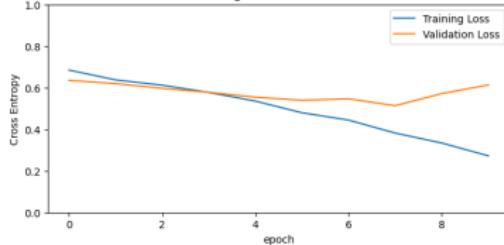
Metric	Value
False Positives (FP)	235.00
False Negatives (FN)	181.00
True Positives (TP)	299.00
True Negatives (TN)	485.00
Precision	0.56
Recall	0.62
F1-score	0.59
Overall Accuracy	0.65

Resultados con CNN

Añadiendo capas de dropout



Metric	Value
False Positives (FP)	47.00
False Negatives (FN)	304.00
True Positives (TP)	176.00
True Negatives (TN)	673.00
Precision	0.79
Recall	0.37
F1-score	0.50
Overall Accuracy	0.71



Utilización de Transfer Learning

El Transfer Learning (aprendizaje por transferencia) es una técnica en machine learning donde un modelo entrenado en una tarea se reutiliza como punto de partida para una nueva tarea relacionada. Esta técnica es especialmente útil cuando se dispone de un conjunto de datos limitado para la nueva tarea, pero se cuenta con un modelo preentrenado en un conjunto de datos grande y diverso. Las dos formas principales de hacer transfer learning son la extracción de características y el fine-tuning (ajuste fino).

Ventajas

- ▶ Eficiencia Computacional
- ▶ Desempeño Competitivo
- ▶ Escalabilidad

Proceso de Extracción de Características

1. Carga del modelo preentrenado:

Se carga un modelo que ha sido preentrenado en un conjunto de datos grande.

2. Congelación de las capas:

Se congelan todas las capas del modelo, excepto la última capa de clasificación, para que los pesos de estas capas no se actualicen durante el entrenamiento.

3. Añadir nuevas capas de clasificación:

Se eliminan las capas de clasificación originales del modelo preentrenado y se añaden nuevas capas densas adecuadas para la nueva tarea.

4. Entrenamiento del nuevo clasificador:

Se entrena solo la nueva parte de clasificación con el conjunto de datos específico de la nueva tarea.

Proceso de Fine-Tuning

1. Carga del modelo preentrenado:

Se carga un modelo preentrenado en un conjunto de datos grande, como ImageNet.

2. Congelación inicial de las capas:

Inicialmente se congelan todas las capas del modelo para entrenar solo las nuevas capas de clasificación.

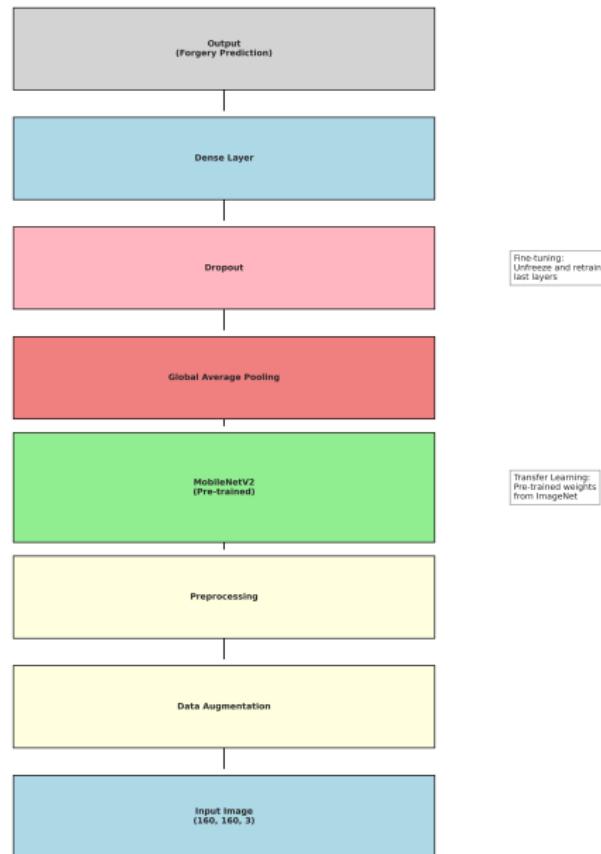
3. Descongelación de algunas capas:

Después de entrenar las nuevas capas de clasificación, se descongelan las últimas capas convolucionales del modelo preentrenado.

4. Entrenamiento conjunto:

Se entrena el modelo completo, ajustando tanto las nuevas capas de clasificación como las capas descongeladas del modelo preentrenado.

Estructura del modelo



Resultados del Entrenamiento

▶ Configuración inicial:

- ▶ 10 epochs de fine-tuning
- ▶ 10 epochs de entrenamiento en capas agregadas
- ▶ Optimizador: RMSprop (tamaño del paso 0.0001)
- ▶ **Resultado:** 70% de rendimiento en entrenamiento y validación

▶ Observaciones:

- ▶ Crecimiento lento y constante en la precisión

▶ Ajustes y resultados:

- ▶ Aumento del tamaño del paso: peores resultados
- ▶ Aumento a 15 epochs: precisión aumenta a 72%
- ▶ **Advertencia:** indicios de overfitting

Uso de Boosting

El boosting es una técnica de ensemble que crea un modelo fuerte a partir de una serie de modelos débiles, donde cada modelo débil es un clasificador que apenas supera la precisión de un clasificador aleatorio. A diferencia de otros métodos de ensemble como el bagging, donde los modelos se entran de manera independiente, en el boosting los modelos se entran secuencialmente.

Proceso de Boosting

1. Inicialización:

Se comienza entrenando un modelo débil (por ejemplo, un árbol de decisión pequeño) utilizando el conjunto de datos original.

2. Pesado de Errores:

Se evalúa el modelo y se aumentan los pesos de las muestras mal clasificadas. Esto hace que el siguiente modelo ponga más atención en estas muestras difíciles.

3. Entrenamiento Secuencial:

Se entrena un nuevo modelo débil utilizando los datos ponderados, y este proceso se repite. Cada nuevo modelo intenta corregir los errores de los modelos anteriores.

4. Combinación de Modelos:

Los modelos entrenados se combinan mediante un método de votación ponderada o una combinación aditiva, donde los modelos que tienen mejor rendimiento tienen un peso mayor en la decisión final.

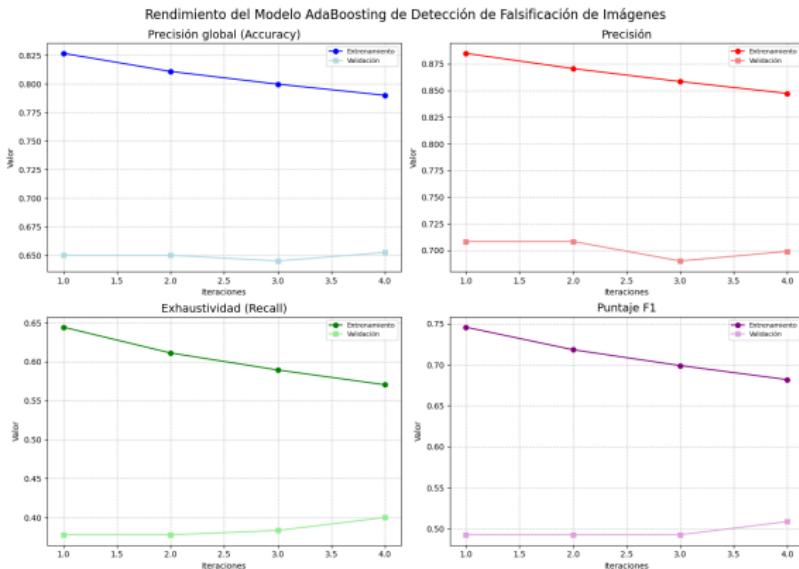
Resultados con Boosting

Durante el entrenamiento de este modelo nos enfrentamos a significativas limitaciones computacionales que restringieron nuestra capacidad para entrenarlo con el conjunto de datos completo. Estas restricciones nos obligaron a utilizar solo una pequeña fracción del dataset disponible, lo cual tuvo un impacto directo en la calidad de los resultados obtenidos generalmente insatisfactorios.

Resultados con Boosting

Modelos Base

4 SVM con el mejor rendimiento que obtuvimos

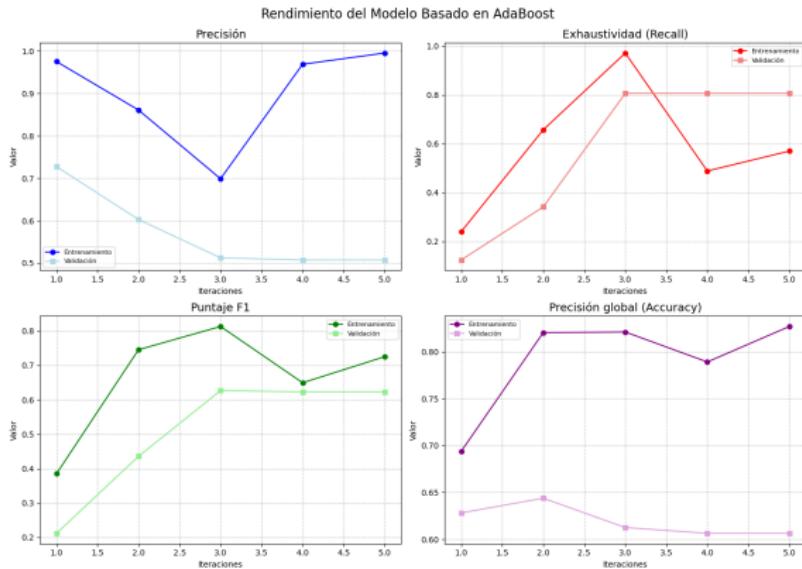


Resultados con Boosting

Modelos Base

5 CNN con los siguientes hiperparámetros:

- ▶ Calidad de compresión: 80
- ▶ Tamaño de paso: 0.0001
- ▶ Epochs: 4



Resultados con Boosting

Modelos Base

5 CNN con los siguientes hiperparámetros:

- ▶ Calidad de compresión: 80
- ▶ Tamaño de paso: 0.0001
- ▶ Epochs: 4

Evaluación

Metric	Value
False Positives (FP)	251
False Negatives (FN)	56
True Positives (TP)	264
True Negatives (TN)	229
Precision	0.51
Recall	0.82
F1-score	0.63
Overall Accuracy	0.62

Conclusiones

- ▶ Las limitaciones computacionales impactaron el rendimiento de modelos complejos.
- ▶ SVM mostró resultados prometedores a pesar de las restricciones.
- ▶ Base establecida para futuras investigaciones.

Propuestas Futuras

- ▶ Ampliación y diversificación del dataset.
- ▶ Mejora en la extracción de características.
- ▶ Optimización de recursos computacionales.
- ▶ Mejora de la interpretabilidad del modelo.
- ▶ Evaluación en escenarios del mundo real.