IA Aplicada a Ingeniería con Python
2023

Trabajo Práctico III

Redes Neuronales Artificiales

Fernando Ezequiel Pose

Índice

1. Objetivos

Afianzar los conceptos respecto a las redes neuronales artificiales (RNA), mediante implementaciones prácticas. Interactuar con diferentes topologías y capas para comprender sus alcances y limitaciones.

Identificar el desempeño de los modelos de acuerdo al tipo de dataset. Evaluar el tamaño de una red en base a la complejidad de la tarea a realizar, determinar costos y tiempo de entrenamiento, establecer criterios de parada en base a métricas de desempeño.

Promover el interés respecto al estado del arte de las redes neuronales artificiales y sus aplicaciones prácticas.

2. Recomendaciones para la resolución del trabajo

Evitar copiar y/o modificar soluciones de pares (compañeros, sitios de Internet, etc.), en lugar de ello, esforzarse por elaborar una producción original propia a partir del análisis y reflexión de cada una de las consignas, teniendo a mano la teoría provista en clases, la bibliografía ofrecida y todo otro material complementario que juzgue necesario para enriquecer su producción.

Reflexionar sobre los conceptos o justificaciones que se ofrecen como solución a la consigna presentada. Es decir, pueden intercambiarse opiniones, debates o puestas en común respecto a un determinado punto, pero la producción entregada debe basarse en su concepción personal del marco teórico, experiencias generales e interpretación de las consignas.

3. Producción esperada para acreditar la actividad

Presentar un informe de estilo monográfico con formato libre en la tarea designada en el aula virtual del curso, incluyendo el contenido solicitado en cada punto de la guía. Realizarlo en tiempo y forma, dentro del plazo máximo de una semana desde la disponibilidad del presente documento.

Además del informe, adjuntar los códigos utilizados para llevar a cabo las experiencias. Los resultados presentados, deben poder ser replicables.

Priorizar la calidad por sobre la cantidad, cuidando la prolijidad general en la confección, incluyendo una portada debidamente identificatoria del trabajo.

4. Ejercicio 1

4.1. Enunciado

Se tiene un dataset con información relacionada a un conjunto de piezas mecánicas. Los datos se encuentran no estructurados, cómo imágenes. Cada categoría de piezas se corresponde con un subdirectorio distinto.

El conjunto de piezas de interés se conforma por 4 categorías, entre ellas: **tuercas**, **tornillos**, **pasadores** y **arandelas**. Al visualizar las observaciones en detalle, notará que pueden existir variantes dentro de una misma categoría (tornillos de diferentes formas, por ejemplo), pero todas computan como la misma clase.

Los datos se encuentran en el archivo adjunto denominado *"partes_mecanicas.zip"*. En la Figura 1 se presenta una muestra de cada una de las piezas de interés.

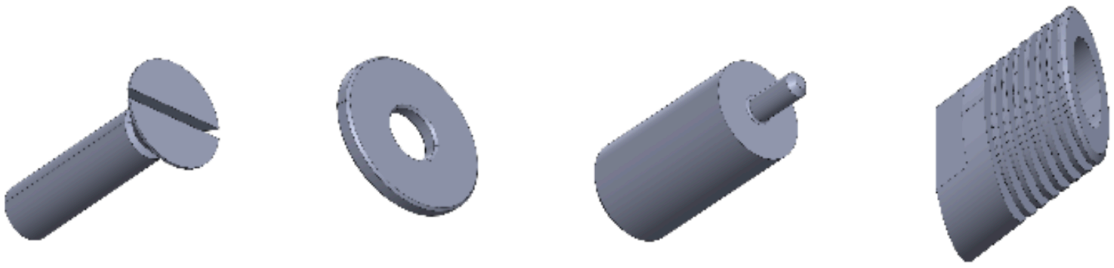


Figura 1: Muestra de las piezas de interés.

El objetivo es determinar si con las observaciones presentadas es posible obtener un modelo que permita **clasificar** cada una de las 4 piezas objetivo.

Puede asumir, que se está desarrollando un sistema de empaquetado automático en la línea de producción de una fábrica, y estas piezas necesitan derivarse desde una línea principal a la línea independiente de la pieza en cuestión. Por lo tanto, al ser todas las piezas del mismo material, el sensor debe ser una cámara que entrega imágenes como las del *dataset*.

Detallar y fundamentar cada aspecto de la solución propuesta, incluyendo los aspectos que crea conveniente y respondiendo como mínimo las siguientes premisas:

- ¿Qué tipo de capas son fundamentales en una RNA que procesa imágenes? ¿Por qué?
- Estas capas, ¿De qué manera consiguen extraer características de las clases objetivo?
- En base a los resultados obtenidos, ¿Sería posible utilizar el modelo para automatizar la separación de piezas en la línea o se debe contratar personal para dicha tarea?
- Con el hardware disponible, ¿Sería viable hacer inferencia en tiempo real con este modelo? (suponiendo una cámara que entrega 24 cuadros por segundos).
- De los resultados visualizados en la matriz de confusión, ¿Para todas las clases el modelo seleccionado presenta el mismo comportamiento?
- ¿Existen problemas evidentes en este dataset? Si es así, ¿Cuáles son y cómo los solucionaría?
- ¿Son necesarios sensores adicionales para que el modelo pueda identificar de manera eficiente las clases de interés?

4.2. Resolución

4.2.1. Conjunto de datos

El conjunto de datos utilizado consta de 4 tipos distintos de imágenes las cuales representan cuatro categorías de piezas mecánicas: tornillos, tuercas, pasadores y arandelas. La distribución de los tipos posibles de piezas en los conjuntos de entrenamiento y prueba se muestra en la siguiente tabla:

Cuadro 1: Conjunto de entrenamiento y prueba

	Muestras		
	Total	Conjunto de entrenamiento	Conjunto de prueba
Tornillo	1904	1523	381
Tuerca	1904	1523	381
Pasador	1904	1523	381
Arandela	1904	1523	381

Tras una primera observación del *dataset* se pudo observar que, dentro de los datos disponibles, en cada una de las categorías existen distintas piezas con diferentes formas y disposición. Por ejemplo, en la Figura 2, puede observarse 5 imágenes distintas correspondientes a la clase tuerca.

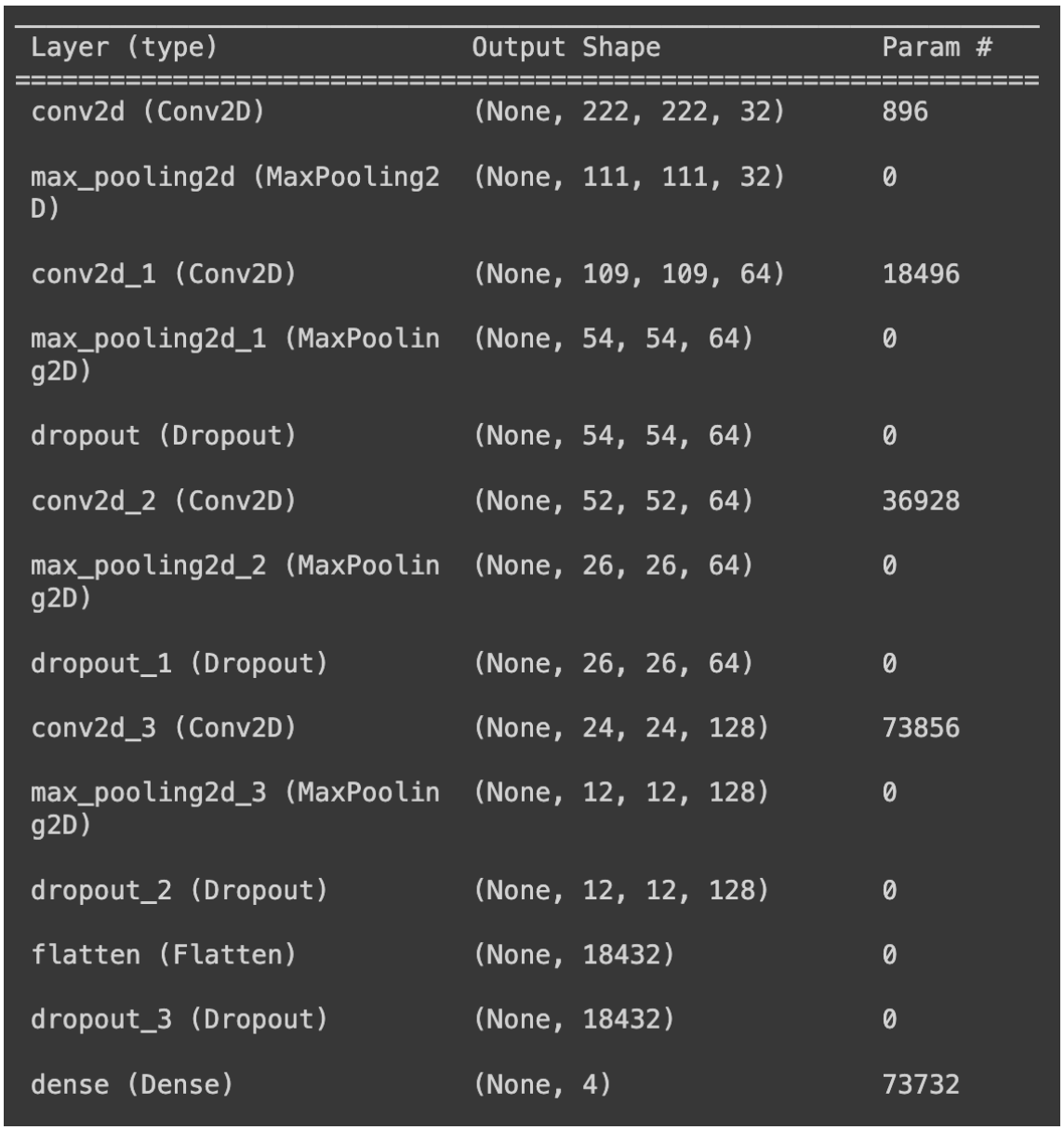
Figura 2: Ejemplo de 5 piezas de la misma clase en elc onjunto de test.

4.2.2. Implementación del modelo

Durante la implementación del modelo de red neuronal convolucional, han sido adoptadas diferentes estrategias a fin de optimizar tanto el tiempo de entrenamiento como su eficiencia. A continuación, se describe cada una de las decisiones tomadas:

1. Entrenamiento: Para el entrenamiento del modelo, se utilizarón 50 épocas con un tamaño de lote de 300. Además, se utilizó un conjunto de validación para la evaluación del modelo en cada iteración de entrenamiento.
2. Callbacks:
   a) Se implementó un criterio de parada temprana con *EarlyStopping*, deteniendo el entrenamiento si la pérdida no mejoraba durante 5 épocas.
   b) Se incorporó *ReduceLROnPlateau* para ajustar automáticamente la tasa de aprendizaje si la pérdida en el conjunto de validación no mejoraba durante 2 épocas.
   c) Se utilizó *ModelCheckpoint* para generar puntos de control durante el entrenamiento, permitiendo la reanudación del entrenamiento en caso de interrupciones o fallos.

La arquitectura de la red diseñada (Figura 3) incluyó cuatro capas *Conv2D* para la detección de patrones, tres capas *MaxPooling2D* para la reducción de la dimensionalidad, cuatro capas *Dropout* para la prevención del sobreajuste debido a la asignación de mayor importancia a un conjunto de neuronas sobre otras, una capa *Flatten* para redimensionar la salida a una única dimensión la cual permita ser acoplada a la última capa *Dense*, la cual se corresponde con la capa de clasificación. La Figura 3 resumen del modelo entrenado.



Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 222, 222, 32)	896
max_pooling2d (MaxPooling2D)	(None, 111, 111, 32)	0
conv2d_1 (Conv2D)	(None, 109, 109, 64)	18496
max_pooling2d_1 (MaxPooling2D)	(None, 54, 54, 64)	0
dropout (Dropout)	(None, 54, 54, 64)	0
conv2d_2 (Conv2D)	(None, 52, 52, 64)	36928
max_pooling2d_2 (MaxPooling2D)	(None, 26, 26, 64)	0
dropout_1 (Dropout)	(None, 26, 26, 64)	0
conv2d_3 (Conv2D)	(None, 24, 24, 128)	73856
max_pooling2d_3 (MaxPooling2D)	(None, 12, 12, 128)	0
dropout_2 (Dropout)	(None, 12, 12, 128)	0
flatten (Flatten)	(None, 18432)	0
dropout_3 (Dropout)	(None, 18432)	0
dense (Dense)	(None, 4)	73732

Figura 3: Red neuronal convolucional diseñada.

La red constó de 203908 parámetros entrenables finalizando su entrenamiento en la época 40 de las 50 posibles debído a que la pérdida en el conjunto de validación (0.06279) no tuvo una variación considerable luego de la época 40. En cuanto al tiempo de entrenamiento, la red necesito 717.6690 segundos para entrenarse haciendo uso el HW provisto por google colab.

4.2.3. Evaluación de resultados

La Figura 4 muestra las gráficas de pérdida y *accuracy* durante el proceso de entrenamiento y validación en cada época del entrenamiento.

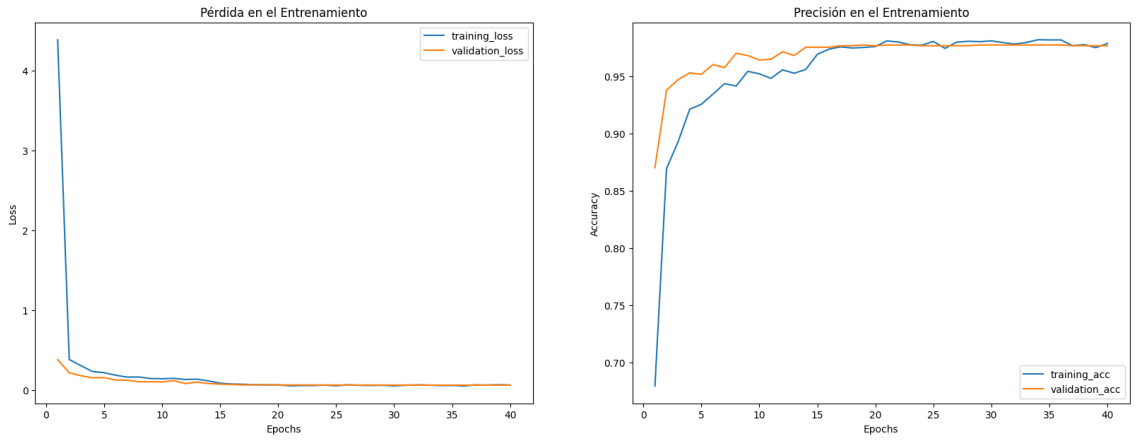


Figura 4: Muestra de las piezas de interés.

Si bien los gráficos reflejan una estabilización de las curvas de accuracy y pérdida a partir de la época 20 y el proceso de finalización de entrenamiento terminó en la época 40, puede observarse en el Código 6, que en la época 16 ya logró un resultado acorde con valores similares de pérdida (0.0768) y accuracy (0.9734) comparables con los resultados finales donde, en el conjunto de prueba, se obtuvo una pérdida de 0.0630 y accuracy de 0.9764.

El tiempo total de predicción en el conjunto de evaluación fue de 1.7765 segundos, mientras que la predicción de una imagen individual tomó 0.2474 segundos.

La Figura 5 ilustra la matriz de confusión obtenida luego de realizar la clasificación sobre el conjunto de datos de prueba.

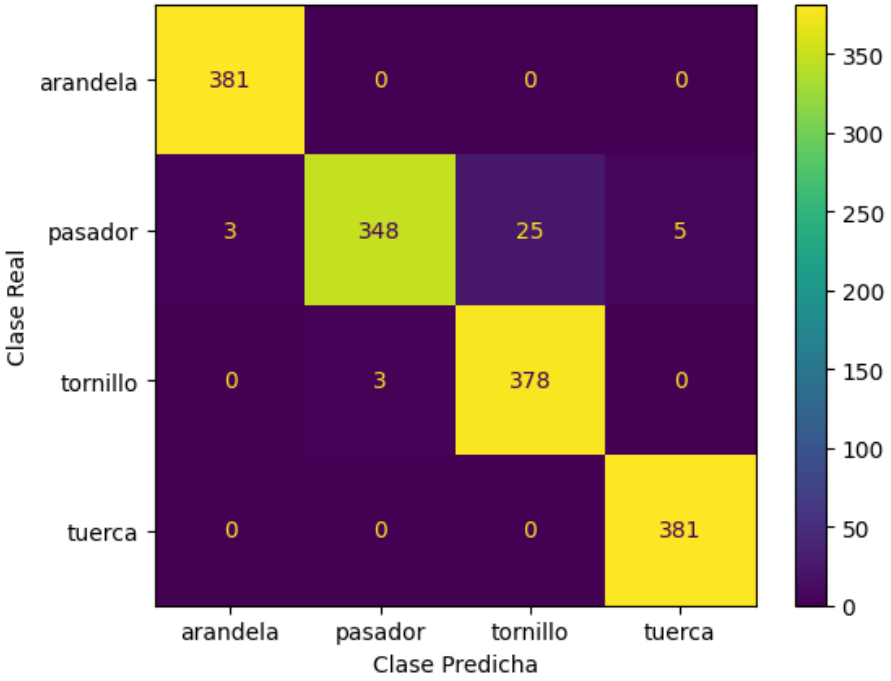


Figura 5: Matriz confusión.

Se observó un sólido rendimiento en general. Sin embargo, la categoría ”Pasador”presentó mayores desafíos

con 25 predicciones incorrectas, siendo principalmente confundida con la clase tornillos.

A continuación, la Figura 6, presenta el reporte de clasificación en el cual se proporciona detalles adicionales sobre precision, recall y f1-score para las clases: arandela (clase 1), pasador (clase 2), tornillo (clase 3) y tuerca (clase 4)

	precision	recall	f1-score	support
0	0.99	1.00	1.00	381
1	0.99	0.91	0.95	381
2	0.94	0.99	0.96	381
3	0.99	1.00	0.99	381
accuracy			0.98	1524
macro avg	0.98	0.98	0.98	1524
weighted avg	0.98	0.98	0.98	1524

Figura 6: Reporte con las principales métricas de clasificación para el modelo.

4.2.4. Conclusiones

Si bien no fueron identificados problemas evidentes en el conjunto de datos propuesto, es importante considerar diversas variables externas en un entorno de producción que podrían afectar la eficacia del modelo. Por ejemplo, factores como la iluminación al capturar las imágenes de las piezas o la orientación de las mismas sobre la cinta transportadora podrían influir en la capacidad de la red para realizar clasificaciones precisas.

Adicionalmente, podría explorarse la incorporación de otros sensores los cuales proporcionen información complementaria sobre las piezas, como sus dimensiones o peso. De este modo, la combinación de una red neuronal convolucional con modelos de aprendizaje automático basados en datos de sensores adicionales podría llegar a mejorar de forma significativa la precisión del sistema completo mediante un sistema de votación entre ambos tipos de modelos. Esta estrategia permitiría abordar de manera más robusta la variabilidad inherente a las condiciones de producción.

En relación con los tiempos que requirió la red, es importante señalar que, aunque el tiempo de entrenamiento fue considerable, el tiempo de predicción se sitúo en 0.2474 segundos para una imagen individual. La evaluación de este parámetro como aceptable o no dependerá de la distancia entre las piezas en la línea de producción. Si esta distancia es lo suficientemente amplia, el sistema podría realizar la clasificación, es decir, el tiempo de adquisición de la imagen y la predicción, en unidades de tiempo deberían ser inferior a la separación en unidad de tiempo entre las piezas en la cinta transportadora.

5. Ejercicio 2

5.1. Enunciado

En el trabajo práctico anterior se abordó un problema referido a la resistencia a la compresión del concreto. Al implementar el modelo diseñado en producción, la dispersión de ciertas variables no se comportó como la de las muestras; por lo tanto, en determinadas ocasiones, la resistencia obtenida por el modelo divergía de la real.

Este problema no pasó desapercibido, ya que comenzaron a aparecer grietas y fisuras en las estructuras realizadas con el concreto mal estimado. Con el objetivo de cuantificar el daño, la empresa registra un conjunto de imágenes con algunas de las zonas afectadas. El objetivo actual, es diseñar un modelo que permita identificar los daños (regiones agrietadas) y, de esta forma, poder inferir de manera automática el porcentaje de daño sobre cientos de metros cuadrados.

Los datos se encuentran en el archivo adjunto denominado *"concreto_agrietado_segmentacion.zip"*. Para cada muestra, un becario capacitado ha realizado un demarcado sobre las zonas afectada. En la Figura 7 se presentan unas muestras de lo mencionado.

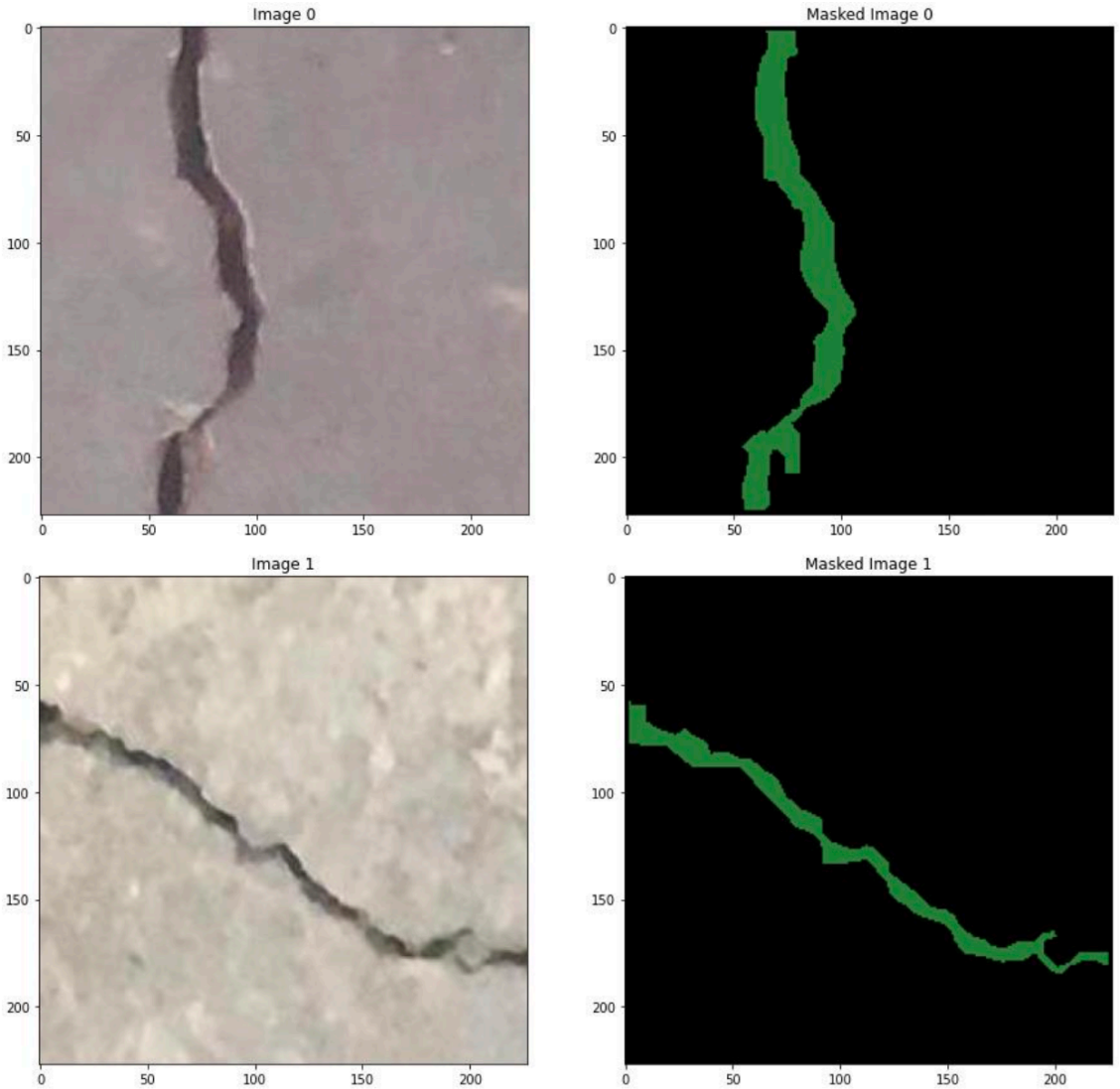


Figura 7: Muestra de las zonas afectadas y su señalización.

El objetivo es determinar si con estas observaciones y anotaciones es posible obtener un modelo que permita **segmentar** las regiones agrietadas en las zonas afectadas. Finalmente, sería ideal contrastar si el modelo funciona correctamente con datos reales nuevos, para ello se disponen de muchas muestras nuevas en:

https://data.mendeley.com/datasets/5y9wdsg2zt/2.

Detallar y fundamentar cada aspecto de la solución propuesta, incluyendo los aspectos que crea conveniente y respondiendo como mínimo las siguientes premisas:

- En base a los resultados obtenidos, ¿Sería posible utilizar el modelo para automatizar la tarea de identificación de daños?
- El modelo obtenido, ¿Sería implementable en algún dispositivo portátil para uso in situ? (Podría hipotetizarse algo respecto a la cantidad de parámetros del modelo).
- ¿Qué métricas utiliza para evaluar el desempeño de un modelo frente a un problema de este tipo?
- ¿Existen problemas evidentes en este dataset? Si es así, ¿Cuáles son y cómo los solucionaría?
- El preprocesamiento de los datos, ¿Afecta de alguna manera el desempeño del modelo?
- Después de presentar su solución a la empresa, ¿Cree que volverían a contratarlo?

5.2. Resolución

El conjunto de datos empleado para el entrenamiento y validación del modelo está compuesto por 822 imágenes, de las cuales 411 representan grietas y las otras 411 corresponden al contorno de dichas grietas en cada imágen. Estas imágenes tienen una resolución de 227 píxeles de ancho, 227 píxeles de alto y 3 canales (RGB). La distribución de las muestras en los conjuntos de entrenamiento y prueba se presenta en la Tabla siguiente:

Cuadro 2: Conjunto de entrenamiento y prueba

	Muestras		
	Total	Conjunto de entrenamiento	Conjunto de prueba
Imagenes	411	329	82

Al realizar una primera observación del conjunto de datos, fue evidenciada la presencia de grietas con diferentes formas y disposición. Por ejemplo, en la Figura **??**, se presentan cuatro imágenes distintas que se corresponden a tres grietas, cada una con su respectiva máscara que indica la ubicación de la grieta, marcada por el becario asignado.

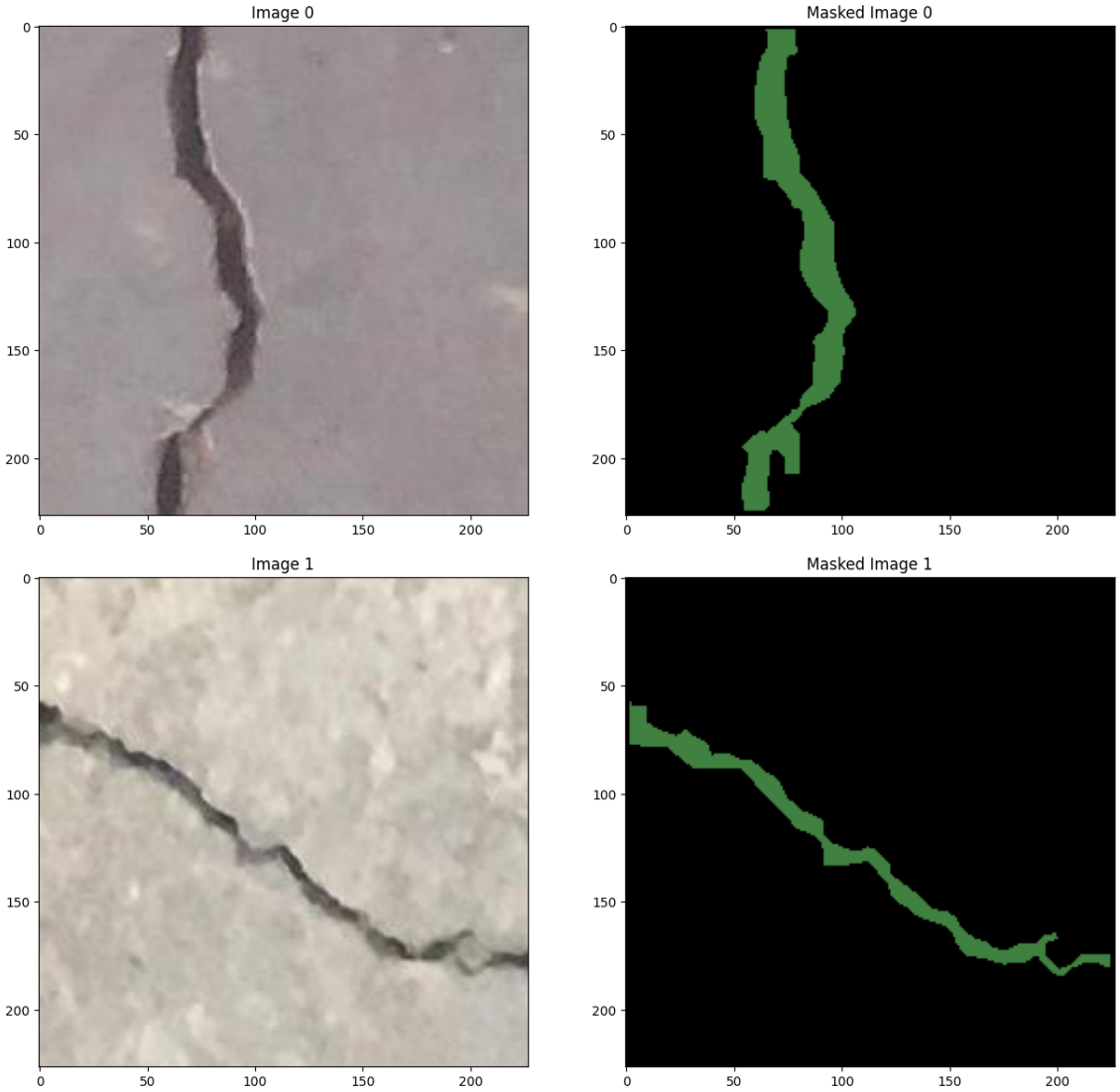


Figura 8: Ejemplo de 3 grietas.

5.2.1. Preprocesamiento de los datos

En la etapa de preprocesamiento, se optó por una nueva resolución. Cada imagen, originalmente de dimensiones (227, 227, 3), se redujo a (128, 128, 3) píxeles, y cada máscara se ajustó a (128, 128, 1) píxeles, donde los valores de la máscara son 1 o 0. La Figura 9 muestra un ejemplo de una imágen y su correspondiente máscara luego de este proceso.

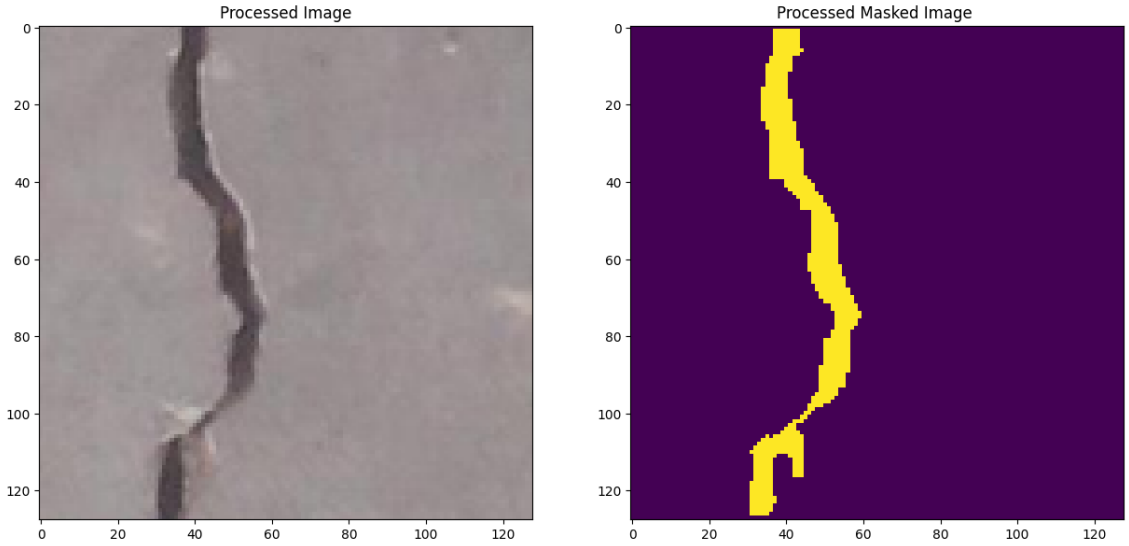


Figura 9: Ejemplo de imagen preprocesada (128, 128, 3) y máscara (128, 128, 1).

Esta reducción de resolución contribuye a la eficiencia computacional durante el entrenamiento del modelo disminuyendo su costo, sin comprometer significativamente la calidad de la imágen.

5.2.2. Implementación del modelo

La arquitectura elegida para la segmentación de grietas en las imágenes es el modelo U-NET (Figura 10).

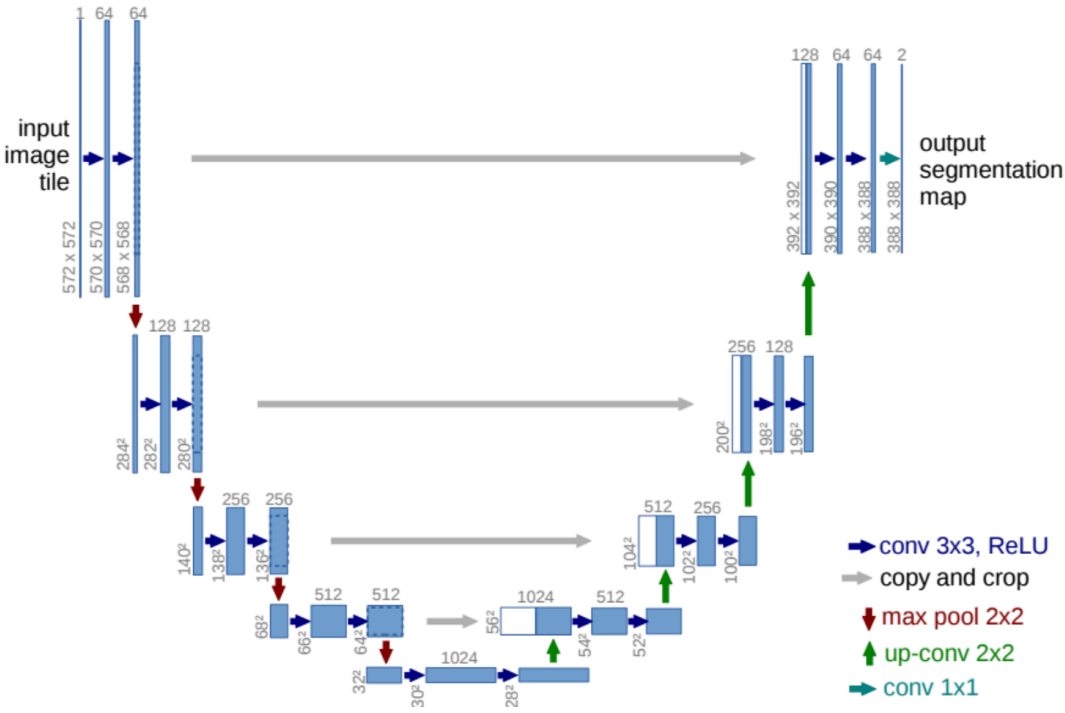


Figura 10: Arquitectura del modelo U-NET para la segmentación de grietas.

La estructura de la red se compone de una capa de entrada, 19 capas de convolución (*Conv2D*) para la detección de patrones, 4 capas de submuestreo (*MaxPooling2D*) para reducir la dimensionalidad, 2 capas *Dropout* para prevenir el sobreajuste, 3 capas de concatenación (*concatenate*) para combinar información entre capas, 3 capas de convolución inversa (*conv2d_transpose*) para la reconstrucción de la información, y 3 capas de normalización (*batch_normalization*) para mejorar la eficiencia del entrenamiento.

El modelo fue entrenado con un tamaño de lote (*batch_size*) de 32 y 20 épocas. La cantidad total de parámetros de la red fue de 8643746, de los cuales 8641762 fueron parámetros entrenables. El entrenamiento concluyó al finalizar la época 20. En cuanto al tiempo de entrenamiento, el modelo necesitó 86.32 segundospara entrenarse haciendo uso del *hardware* proporcionado por Google Colab. En el código del ejercicio (Sección 6) se puede encontrar un resumen del modelo entrenado.

5.2.3. Evaluación del modelo

En la Figura 11, se presentan las curvas de pérdida, *accuracy* y el Índice de Intersección sobre Unión (IoU) durante el proceso de entrenamiento y validación en cada época.

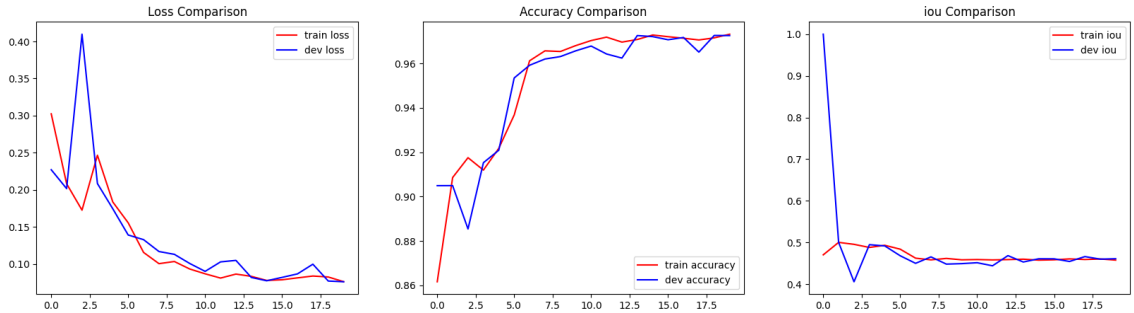


Figura 11: Curvas de pérdida, *accuracy* y IoU durante el entrenamiento.

En los gráficos, puede observarse que la métrica IoU en validación tiende rápidamente a su valor final en las primeras épocas de entrenamiento. En cuanto a las métricas de pérdida y *accuracy*, se estabilizan alrededor de la época 17, alcanzando valores cercanos a los finales después de las 20 épocas de entrenamiento. En términos cuantitativos, se obtuvieron valores de pérdida, IoU y *accuracy* de 0.0763, 0.4574 y 0.9732 para el conjunto de entrenamiento, y 0.0762, 0.4607, 0.9725 para el conjunto de validación. Es importante destacar que la métrica IoU evalúa la precisión en la ubicación y el tamaño de la máscara predicha por el modelo en comparación con la máscara real.

Como ejemplo, la Figura 12 (izquierda) muestra una imágen a segmentar junto con su máscara correspondiente. La misma figura, parte derecha, muestra la máscara predicha por el modelo y una superposición de esta máscara en la imágen real. Para esta predicción, la red neuronal requirió 0.9728 segundos en predecir.

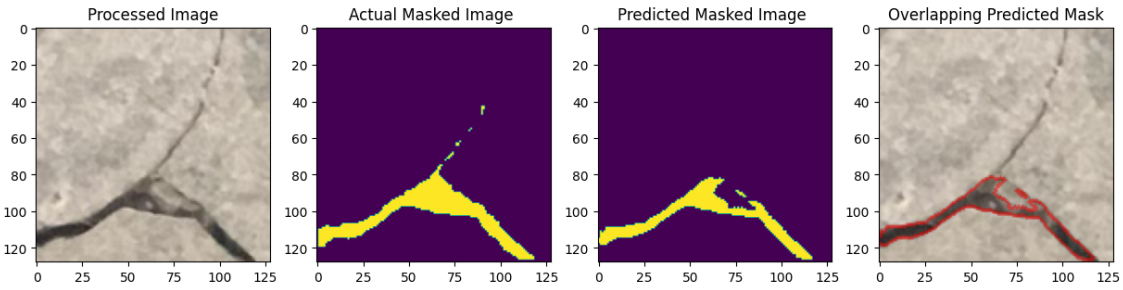


Figura 12: Ejemplo de segmentación de grietas.

Respecto al bajo valor de IoU podría ser explicado por el hecho de que, aunque la red detecta la grieta, la máscara predicha tiene una superficie menor que la máscara real. Por lo tanto, a nivel cuantitativo, para este conjunto de datos, la métrica IoU puede no ser realmente efectiva, aunque el modelo cumple el objetivo de detectar las grietas, incluso, si el valor de IoU, es bajo.

5.2.4. Evaluación del modelo con nuevos datos

En esta sección, se somete el modelo entrenado en 5.2.2) a una prueba utilizando un nuevo conjunto de imágenes, compuesto por imágenes con grietas y sin grietas. Del total de las imágenes, fueron seleccionadas 500 imágenes de cada tipo para llevar a cabo el análisis.

En el primer análisis, tras la aplicación del modelo entrenado, se observó que de las 500 imágenes con grietas, 394 (78.8 %) fueron correctamente clasificadas, mientras que 106 (21.2 %) fueron clasificadas erróneamente como imágenes con grietas cuando no lo eran. Mientras que, de las 500 imágenes sin grietas, 394 (78.8 %) fueron correctamente clasificadas, pero 106 (21.2 %) fueron clasificadas incorrectamente.

Tras una inspección de los resultados, se identificó que algunas imágenes sin grietas presentaban manchas negras que podían ser interpretadas como grietas por el algoritmo. Por este motivo, se introdujo un umbral de 20 píxeles en el proceso de clasificación.

En este nuevo enfoque, cuando el modelo detectó en una imagen sin grietas una grieta con un tamaño de máscara predicha inferior a 20 píxeles, se reclasificó como sin grietas. Con este ajuste, se logró una clasificación final para el conjunto de datos de imágenes sin grietas, un total de 463 (92 %) imágenes correctamente clasificadas y 37 (7.39 %) imágenes mal clasificadas.

Aplicando el mismo umbral en el conjunto de imágenes con grietas, se obtuvo un total de 461 (92.2 %) imágenes con grietas correctamente clasificadas y 39 (7.8 %) imágenes clasificadas incorrectamente.

Es relevante tener en cuenta que en imágenes de (128, 128) píxeles (total: 16384 píxeles), 20 píxeles representan un 0.12 % del total de píxeles de la imágen. Por lo tanto, no se esperaría que se estén pasando por alto grietas significativas con una cantidad mínima de píxeles. La Figura 13 ejemplifica el caso de una imágen clasificada inicialmente como con grieta, pero tras la reclasificación debido al tamaño de la máscara predicha, la clasificación fue corregida.

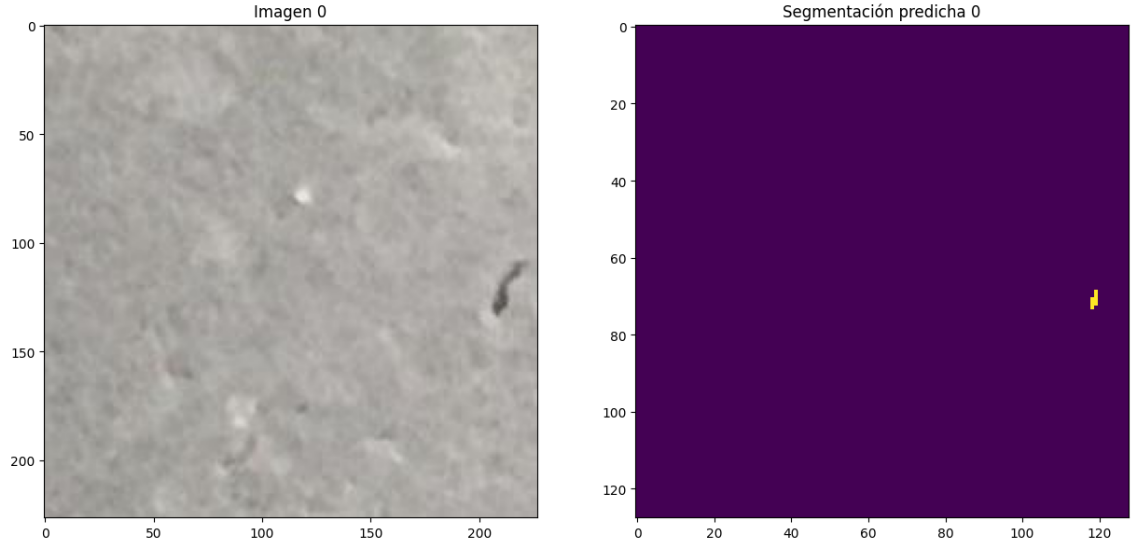


Figura 13: Ejemplo de reclasificación de una imagen sin grietas.

5.2.5. Conclusiones

En el transcurso de este trabajo práctico, se llevo a cabo la evaluación de un modelo U-Net con el objetivo de identificar grietas en estructuras de concreto. A pesar de que la métrica IoU arrojó un valor bajo, el modelo demostró eficacia al detectar y señalar las áreas afectadas por grietas en el conjunto de datos proporcionado.

En cuanto a la viabilidad de implementar este modelo en dispositivos portátiles, destaca la eficiencia del U-Net. A pesar de su robustez, no impone una cantidad de parámetros excesiva, lo que permitiría su integración en dispositivos portátiles, facilitando su utilización en tiempo real.

El análisis del conjunto de datos, especialmente durante la segunda evaluación del modelo (Sección 5.2.4), reveló la presencia de manchas oscuras interpretadas como grietas en imágenes sin daños. Si bien la introducción de un umbral durante la clasificación contribuyó a mitigar este problema, se enfatiza la importancia de mejorar la calidad y limpieza de los datos durante la etapa de preprocesamiento a fin de fortalecer la robustez del modelo. Por lo tanto, como una sugerencia para futuras mejoras, podrían considerarse técnicas avanzadas de preprocesamiento de imágenes, como lo son las técnicas de maquillaje.

Desde una perspectiva empresarial, la solidez del modelo sugiere que podría desempeñar un papel significativo en la identificación automatizada de daños en estructuras de concreto. La combinación de un umbral de clasificación adecuado junto con un preprocesamiento de los datos podría ampliar aún más la utilidad y confiabilidad de esta herramienta en el ámbito práctico.

6. Material complementario

6.1. Notebooks

- Ejercicio 1
- ejercicio 2