UNIVERSIDAD NACIONAL DE SAN CRISTÓBAL DE HUAMANGA

FACULTAD DE INGENIERÍA DE MINAS, GEOLOGÍA Y CIVIL

ESCUELA PROFESIONAL DE INGENIERÍA CIVIL

TESIS:

"Desarrollo de un modelo de aprendizaje automático para evaluar daños en pavimentos flexibles mediante el procesamiento de imágenes en el tramo Socos - Licapa, 2025"

Para optar el título profesional de:
INGENIERO CIVIL

PRESENTADO POR:
Bach. Emergio ALANYA CUBA

ASESOR:
MSc. Ing. Saúl Walter RETAMOZO FERNÁNDEZ

AYACUCHO - PERÚ

2025

Resumen

La identificación de grietas en el tramo Socos-Licapa de la Vía Los Libertadores-Wari es fundamental para mantener la seguridad del tránsito vehicular. El uso de métodos tradicionales de inspección visual suele interrumpir el flujo normal del tráfico y presenta limitaciones en cuanto a objetividad y tiempo de ejecución. Por ello, se propone un algoritmo de detección basado en YOLOv12, diseñado para identificar grietas en carreteras mediante imágenes captadas con drones (UAV), cámaras convencionales y fotografías terrestres, empleando técnicas de visión por computadora. Este algoritmo busca alcanzar una detección precisa de grietas en distintos tamaños y escalas. Las pruebas realizadas con nuestro dataset, que contiene 1871 imágenes de escenas cercanas y lejanas captadas con cámara, evidencian la efectividad del modelo YOLOv12 utilizando Deep Learning. Se obtuvo una precisión del modelo para todas las clases de fallas con un mAP@50 de 70.4 % y un mAP@[50:95] de 41.9 %. Estos resultados confirman que el modelo mantiene un bajo peso computacional y ofrece un rendimiento de detección altamente eficaz, cumpliendo así con el objetivo trazado. La metodología utilizada es no experimental y transeccional, ya que los datos fueron recolectados en un solo momento, por lo que las imágenes obtenidas corresponden a un espacio y tiempo.

Palabras clave: detección de grietas, visión por computadora, aprendizaje profundo, YOLO V12 , Drones.

Abstract

The identification of cracks on the Socos-Licapa section of the Los Libertadores-Wari highway is essential for maintaining vehicular traffic safety. The use of traditional inspection methods often disrupts normal traffic flow and presents limitations in terms of objectivity and execution time. Therefore, a detection algorithm based on YOLOv12 is proposed, designed to identify road cracks using images captured by drones (UAVs), conventional cameras, and terrestrial photographs, employing computer vision techniques. This algorithm aims to achieve accurate crack detection at different sizes and scales. Tests conducted with our dataset, which contains 1,871 images of near and far scenes captured with a camera, demonstrate the effectiveness of the YOLOv12 model using Deep Learning. The model achieved a detection precision of 70.4 % mAP@50 and 41.9 % mAP@[50:95] for all crack classes. These results confirm that the model maintains low computational weight while providing highly effective detection performance, thus fulfilling the stated objective. The methodology used is non-experimental and cross-sectional, as data was collected at a single point in time; therefore, the images obtained correspond to a specific space and time.

Keywords: road crack detection, computer vision, deep learning, You Only Look Once, Unmanned Aerial Vehicle (UAV) .

Introducción

La presente investigación se estructura en cinco capítulos, los cuales se describen brevemente a continuación:

- **Capítulo 1: Planteamiento del problema.** Se describe el problema central de la investigación: la falta de seguridad vial debido a las condiciones actuales de la carretera y las limitaciones de los métodos tradicionales de inspección. Se justifica el uso de imágenes obtenidas en campo como dataset para la detección automatizada de fallas mediante inteligencia artificial.
- **Capítulo 2: Marco Teórico.**Se desarrollan los fundamentos conceptuales y teóricos que sustentan la investigación, incluyendo la clasificación de fallas en pavimentos flexibles, los conceptos de procesamiento digital de imágenes y los principios del aprendizaje profundo (Deep Learning), con énfasis en redes neuronales convolucionales y el algoritmo YOLOv12.
- **Capítulo 3: Metodologia de Investigación.** Se describe el enfoque cuantitativo de la investigación, el diseño no experimental y transeccional, así como los procedimientos de recolección, etiquetado y entrenamiento del dataset. Se explican las métricas de evaluación del modelo (precisión, recall, F1-score y mAP).
- **Capítulo 4: Análisis de resultado de la investigación.** Se presentan los resultados obtenidos con el modelo YOLOv12, incluyendo las métricas de precisión para cada clase de falla (D00, D10, D20, D40, REPAIR) y la interpretación de los mismos.
- **Conclusiones y recomendaciones.** Se sintetizan los hallazgos más importantes de la investigación y se formulan recomendaciones para futuros trabajos y aplicaciones prácticas.

  Finalmente, se incluyen las referencias bibliográficas y los anexos con evidencia fotográfica y el código fuente utilizado.
- **Referencias bibliográficas.**La presente investigación sobre todo YOLO V12, requiere de mucha información actual por lo que la fuente bibliográfica actual se obtiene de las revistas cientificas actuales sobre todo para YOLO V12 que se encuentra en su última versión.
- **Anexos.** Paneles fotográficos obtenidos en campo para el tramo de Socos Licapa.

Dedicatoria

A Dios: Porque cada logro es un eco de Su gracia. Esta tesis no es solo un trabajo académico, es un testimonio de fe y un agradecimiento por el don del conocimiento.

A mi madre, Cirila Cuba Ochoa: A tu abrazo que calma cualquier tormenta, a tus manos trabajadoras que construyeron mi futuro, a tu sabiduría sencilla que es la más profunda. Madre, en cada página de este trabajo hay una pizca de tu inmenso amor.

A mi padre, Julián M. Alanya Ventura: A tu ejemplo de integridad, a la fuerza tranquila que me impulsa, a las lecciones que no estaban en los libros pero que son las más importantes. Esta meta se alcanza también sobre los hombros de tu esfuerzo.

Agradecimientos

A la Universidad Nacional San Cristóbal de Huamanga, mi alma mater, Con profundo respeto y gratitud eterna, dedico los frutos de este esfuerzo intelectual a la institución que forjó mi pensamiento, alimentó mi curiosidad y me dotó de las herramientas para enfrentar los desafíos de la profesión.

Al Msc. Ing. Ernesto Estrada Cárdenas agradezco por su invaluable orientación, sus conocimientos técnicos precisos y su exigencia profesional, que fueron fundamentales para dar rigor y solidez a este trabajo. Su mentoría ha sido una pieza clave en la culminación de este proyecto..

Al Msc. Ing. Edmundo Canchari Gutiérrez, quien me enseñó que la ingeniería no es solo una ciencia de números y cálculos, sino un arte al servicio de la humanidad.

Al Msc. Ing. Saúl, RETAMOZO FERNÁNDEZ, por su rigurosidad técnica, su sabiduría y su inagotable paciencia para orientar cada paso de esta investigación fueron el pilar sobre el cual se construyó esta tesis. Más que un asesor, encontré en usted un maestro comprometido con la excelencia y el rigor científico.

Universidad Nacional de San Cristóbal de Huamanga

Ayacucho, 2025

Emergio, Alanya Cuba

Índice general

Índice de figuras

Índice de tablas

Glosario

Entrenamiento: La red neuronal artificial se entrena con la iteración de los parámetros de entrada hasta encontrar los pesos sinápticos ajustados al mejor modelo de la red con la salida de los datos estimados aproximados a los esperados.

Retropropagación: propagación de los datos hacia atras para actualizar los pesos sinápticos hasta encontrar el modelo deseado.

Entrenamiento de red: Iteración con parámetros de entrada hasta encontra mejor modelo en la salida de datos.

Generación sintética: Creación de datos utilizando modelos matemáticos o estadísticos.

Desempeño de la red: Es el modelo que más aprende en el entrenamiento para optimizar la red.

Deep Learning (Aprendizaje Profundo): Subcampo del machine learning que utiliza redes neuronales con múltiples capas (redes profundas) para aprender representaciones jerárquicas de datos.

Red Neuronal Convolucional (CNN): Volumen disponible de agua para poder satisfacer la demanda hídrica.

Convolución: Operación matemática que aplica un filtro (kernel) a una imagen para extraer características como bordes, texturas o patrones específicos..

Pooling (Agrupamiento): Reducción de dimensionalidad que retiene la información más importante, comúnmente mediante max-pooling (valor máximo) o average-pooling (valor promedio).

Función de Activación: Función no lineal aplicada a la salida de una neurona. Ejemplos: ReLU, Sigmoid, Tanh..

Backpropagation (Retropropagación): Algoritmo para ajustar los pesos de la red calculando el gradiente de la función de pérdida con respecto a cada peso..

Época (Epoch): Pase completo del conjunto de entrenamiento a través de la red neuronal..

Batch Size (Tamaño del Lote): Número de muestras procesadas antes de actualizar los pesos de la red..

Learning Rate (Tasa de Aprendizaje): Hiperparámetro que controla cuánto se ajustan los pesos de la red durante el entrenamiento..

Overfitting (Sobreajuste): Cuando el modelo aprende demasiado los detalles del conjunto de entrenamiento, perdiendo capacidad de generalización..

Underfitting (Subajuste): Cuando el modelo no captura adecuadamente los patrones de los datos.

Descenso por gradiente: Es un algoritmo que estima numericamente donde una función genera su valores mas bajos que tiende a converger la función.

YOLO: Arquitectura de detección de objetos que realiza detección y clasificación en una sola pasada por la red.

Bounding Box (Caja Delimitadora): Rectángulo que encierra un objeto detectado, definido por coordenadas (x, y, ancho, alto).

Acrónimos

AI	Inteligencia Artificial (Artificial Intelligence)
ANN	Artificial network neural
BPA	Back Propagation Algorith
CNN	Red Neuronal Convolucional (Convolutional Neural Network)
CSP	Etapa Parcial Convolucional (Cross Stage Partial)
CV	Visión por Computadora (Computer Vision)
DL	Aprendizaje Profundo (Deep Learning)
FFN	Feed forward network
GAN	Red Generativa Antagónica (Generative Adversarial Network)
Inception	Red Inception (GoogleNet)
ML	Aprendizaje Automático (Machine Learning)
MSE	Mean square error
NLP	Procesamiento de Lenguaje Natural (Natural Language Processing)
NMSE	Normalized mean square error
PAN	Red de Agregación de Ruta (Path Aggregation Network)
ReLU	Unidad Lineal Rectificada (Rectified Linear Unit)
ResNet	Red Residual (Residual Network)
RMSE	Root mean square error
RNN	Red Neuronal Recurrente (Recurrent Neural Network)
SSD	Detector de Una Sola Pasada (Single Shot MultiBox Detector)
VGG	Grupo de Geometría Visual (Visual Geometry Group)
Wi	Pesos sinápticos
YOLO	Sólo Miras Una Vez (You Only Look Once)

Símbolos

$\sum_{i=1}^{n} x_i$	Sumatoria
$\prod_{i=1}^{n} x_i$	Productoria
$\frac{\partial f}{\partial x}$	Derivada parcial
∇f	gradiente
α	Tasa de aprendizaje (learning rate)
λ	Término de regularización
μ	Media
σ	Desviación estándar / función sigmoid
x	vector de entrada
ϵ	Valor pequeño / error
W	Matriz de pesos
b	Vector de sesgo(bias)
z	Activación lineal pre-función
ε_t,ξ_t	Componentes estocástico
L	Función de pérdida(loss)
B	Número de bounding boxes por celda
C	Número de clases
TP	Verdaderos positivos
TN	Verdaderos negativos
FP	Falsos positivos
FN	Falsos negativos
$\hat{\beta}_i$	Coeficientes de regresión lineal esetimado
t_i	t de student
SCE	Suma de cuadrados explicativa
SCR	Suma de cuadrados residuales

SCT	Suma de cuadrados totales
C_L	Grietas longitudinales
C_t	Grietas transversales
C_p	Grietas en bloque

"Empieza por el principio," – dijo el Rey con gravedad – "y sigue hasta llegar al final; allí te paras."

— Lewis Carroll, *Alice in Wonderland*

1 Planteamiento del problema

1.1 Descripción del problema

El seguimiento periódico del estado superficial de los pavimentos flexibles es esencial para programar acciones de mantenimiento apropiadas y oportunas, evitando así un desgaste prematuro, disminuyendo costos operativos excesivos y optimizando las condiciones de seguridad para los usuarios viales. Por ello, existe una necesidad creciente de disponer de herramientas y enfoques más eficaces que faciliten la evaluación y detección de fallas en este tipo de pavimentos, buscando reducir tanto el tiempo como los gastos asociados a dichos procesos.

Actualmente, la detección de daños se realiza de forma artesanal, a través de una inspección visual directa y el registro en formatos técnicos impresos. Este enfoque presenta diversas desventajas:

- Obliga a trasladar personal especializado hasta el lugar de evaluación.
- interrupción del tránsito vehicular mientras se realiza la inspección.
- Resulta subjetivo, ya que depende del criterio y la experiencia del inspector.
- Implica una inversión considerable de tiempo y recursos económicos.

Ante esta situación, la presente investigación propone desarrollar un modelo basado en Deep Learning, apoyado en el procesamiento digital de imágenes, para la valoración automatizada de deterioros en pavimentos flexibles. Dicha solución representa un aporte significativo, dado que disminuye la necesidad de contar con personal calificado en campo, se traduce en un ahorro de tiempo y recursos humanos, y permite unificar los criterios de evaluación mediante un software entrenado con métodos de Inteligencia Artificial, garantizando así mayor coherencia y objetividad en los diagnósticos.

1.2 Delimitación del problema

1.2.1 Espacial (Geográfica)

El estudio se circunscribe específicamente al tramo carretero Socos-Licapa, comprendido dentro de la Vía Los Libertadores-Wari, en la región Ayacucho, Perú.

1.2.2 Temporal

La captura de las imágenes se efectuó durante enero de 2025. El procesamiento y examen de los datos se desarrolló entre enero y marzo del mismo año.

1.2.3 Temática y unidad de análisis

Las categorías centrales de análisis abarcan cinco tipos de fallas en pavimentos flexibles: piel de cocodrilo (D00), parches (D10), fisuras en bloque (D20), fisuras de borde (D40) y reparaciones mayores (REPAIR). La detección y tipificación se ejecutan de forma automatizada empleando un modelo YOLOv12 entrenado específicamente para esta función.

1.3 Formulación del problema

1.3.1 Problema general

¿De qué manera el campo de la inteligencia artificial permite crear un modelo de Deep Learning que pueda evaluar diferentes tipos de daños en pavimentos flexibles mediante el procesamiento digital de imágenes obtenidas en el tramo Socos-Licapa?

1.3.2 Problemas específicos

1. ¿ Cómo los modelos de Deep Learning basados en entrenamiento con imágenes pueden reducir los tiempos en la evaluación de diferentes tipos de daños en pavimentos flexibles?
2. ¿De qué manera los datos de entrenamiento obtenidos del tramo Socos-Licapa y la configuración de la arquitectura de red influyen en el desempeño del modelo de Deep Learning?
3. ¿Es posible identificar y evaluar los daños en pavimentos flexibles mediante un modelo de Deep Learning a partir de datos de imágenes?

1.4 Justificación e importancia

Este trabajo se fundamenta en la carencia de una metodología eficiente (en términos de tiempo y costo) para efectuar inspecciones visuales y catalogar los daños en las vías. El desarrollo del estudio posibilita construir un modelo fundamentado en redes neuronales con capacidad de identificar de forma autónoma cinco tipos de fallas en pavimentos flexibles, constituyendo un sistema extrapolable a otros segmentos viales. Esta herramienta permite diagnosticar los deterioros de manera más ágil, favoreciendo una intervención rápida y planificada en la red de carreteras. Como beneficio final, se ven directamente favorecidos los conductores y la población residente en el área de influencia de la carretera, quienes cuentan con un servicio de tránsito adecuado y seguro.

1.5 Limitaciones o restricciones

- Restricción por escasez de datos: El conjunto de imágenes se reduce a un único tramo vial (Socos-Licapa) y a una ventana de captura determinada (junio de 2022).
- Limitación por condiciones ambientales: Las fotografías fueron tomadas bajo iluminación diurna y en clima seco, lo cual podría limitar la capacidad de generalización del modelo.
- Restricción por capacidad computacional: El entrenamiento del modelo exige recursos informáticos considerables (GPU con al menos 8 GB de RAM).
- Limitación en el número de categorías: Únicamente se contemplaron 5 tipos de fallas, de las 16 descritas en la bibliografía especializada.

1.6 Objetivos

1.6.1 Objetivo general

Definir un modelo de Deep Learning con parámetros optimizados para identificar y evaluar cinco tipos de daños en pavimentos flexibles mediante el procesamiento digital de imágenes obtenidas en el tramo Socos-Licapa.

1.6.2 Objetivos específicos

1. Analizar los parámetros del modelo de Deep Learning basado en entrenamiento con imágenes para reducir los tiempos de evaluación de daños en pavimentos flexibles.
2. Recolectar y etiquetar un dataset de imágenes del tramo Socos-Licapa para entrenar un modelo de Deep Learning que permita evaluar cinco tipos de daños en pavimentos flexibles.
3. Desarrollar e implementar un modelo YOLOv12 para la detección automatizada de daños en pavimentos flexibles mediante procesamiento de imágenes.

"El aspecto más triste de la vida actual es que la ciencia gana en conocimiento más rápidamente que la sociedad en sabiduría."

— Isaac Asimov

2 Marco teórico

2.1 Antecedentes

2.1.1 Investigaciones internacionales.

En el estudio titulado "Predicción de la vida útil remanente de carreteras mediante el índice de condición del pavimento", Setyawana et al. (2015) evaluaron el estado del pavimento para estimar su vida remanente. Dicha investigación, ejecutada en cinco segmentos de una vía localizada al sureste de Sumatera, utilizó el Índice de Condición del Pavimento (PCI) junto con mediciones de deflexión para efectuar los pronósticos. Se halló una correlación entre el valor del PCI y la vida remanente, derivando el modelo matemático y = 4.1872ln(x) - 14.728, el cual exhibió un coeficiente de correlación de 0.88, lo que evidencia una relación estadísticamente fuerte.

Por su parte, Shahnazari et al. (2012) desarrollaron la investigación .ªplicación de la computación blanda para la predicción del Índice de Condición del Pavimento", cuyo objetivo fue proponer un método alternativo para predecir el PCI empleando técnicas de optimización como las redes neuronales artificiales (RNA) y la programación genética (PG). Utilizaron una base de datos con valores de PCI provenientes de más de 1250 km de vías en Irán. Los enfoques propuestos lograron estimar el PCI de manera fiable, arrojando valores muy cercanos a los observados en campo. Los investigadores concluyeron que el modelo basado en RNA fue más exacto que el basado en PG, con coeficientes de correlación de 0.9986 y 0.9887, respectivamente.

En la actualidad, a nivel global se emplean sistemas automatizados de inspección vial que integran diversos equipos especializados para asegurar la fiabilidad y eficiencia del proceso.

2.1.2 Investigaciones nacionales

En su tesis, Leguía y Pacheco (2016) realizaron la Evaluación superficial del pavimento flexible mediante el método pavement condition index (PCI) en las vías arteriales: Cincuentenario, Colón y Miguel Grau (Huacho-Huaura-Lima). El propósito central de esta investigación fue diagnosticar el estado de las citadas vías aplicando el método del Índice de Condición del Pavimento (PCI), con el fin de determinar la condición real del pavimento flexible. Tras la implementación de esta metodología, los hallazgos indicaron que la Av. Cincuentenario presenta un nivel de conservación

regular", con un PCI de 51.84, mientras que las avenidas Colón y Miguel Grau muestran un estado "bueno", con un PCI de 59.29.

Por otro lado, Paytán (2018) desarrolló en su tesis .Estimación del índice de regularidad internacional en pavimentos flexibles usando redes neuronales artificialesün estudio orientado a comparar el valor del Índice de Regularidad Internacional (IRI) estimado mediante un modelo de redes neuronales artificiales con el valor obtenido directamente a través de un perfilómetro láser en la carretera PE-1S. El modelo neuronal fue construido empleando el software Matlab y, para su validación, se utilizó una base de datos distinta a la empleada durante la etapa de entrenamiento. Los valores de IRI generados por el modelo fueron confrontados con las mediciones realizadas con el perfilómetro láser. El análisis arrojó un coeficiente de correlación de R = 0.365, lo cual refleja una relación débil entre ambas mediciones. Con base en esto, el autor concluyó que, para esta vía en particular, el modelo basado en redes neuronales artificiales no tuvo un rendimiento aceptable.

2.2 Bases teóricas

2.2.1 Pavimento flexible

Los pavimentos flexibles son sistemas estructurales formados por capas granulares (subbase y base) y una superficie de rodadura elaborada con materiales asfálticos. Reciben la denominación de "flexibles"debido al mecanismo mediante el cual distribuyen las cargas vehiculares desde la capa de rodadura hacia la subrasante. En este esquema, el asfalto actúa principalmente como un medio de transmisión de dichas cargas. La mayoría de los pavimentos flexibles requieren la incorporación de varias capas intermedias dentro de su estructura, conformando un conjunto estructural que se extiende desde la carpeta de rodadura hasta la subrasante (Becerra Salas, 2012; Ministerio de Transportes y Comunicaciones, 2013).

2.2.2 Clasificación de fallas de pavimento flexible

Existen múltiples tipos de fallas en pavimentos flexibles. De acuerdo con el manual de INVIAS (2006) y la norma ASTM D6433, se pueden identificar las siguientes categorías principales:

Código	Tipo de falla	Característica principal
D00	Piel de cocodrilo	Fisuras interconectadas por fatiga estructural
D10	Parche	Zona reparada con material nuevo
D20	Fisura en bloque	Grietas que dividen el pavimento en bloques
D40	Fisura de borde	Grietas cerca del borde de la calzada
REPAIR	Reparación extrema	Intervención mayor en la superficie

Justificación de la selección de clases

Para esta investigación, se eligieron las 5 clases mencionadas (D00, D10, D20, D40, REPAIR) por ser las más representativas y recurrentes en el tramo de estudio (Socos-Licapa), conforme a la inspección visual preliminar. Las otras 11 clases descritas en la literatura (ondulaciones, abultamiento, ahuellamiento, etc.) no fueron incorporadas debido a su baja o nula aparición en el tramo analizado.

2.2.2.1 Fisuras longitudinales y transversales

Estas discontinuidades aparecen en la carpeta asfáltica, pudiendo alinearse con el sentido del tráfico o disponerse de manera perpendicular a este. Su presencia indica la existencia de tensiones de tracción en alguna de las capas que componen el pavimento, las cuales han excedido la resistencia admisible del material (INVIAS, 2006).

Figura 1 Fisura transversal (FT, Unidad de medida: m)

Fuente: INVIAS (2006)

2.2.2.2 Fisuras en juntas de construcción

Este tipo de fisuras, ya sean longitudinales o transversales, surgen por una mala ejecución de las juntas de construcción en la capa asfáltica o en las conexiones de las zonas ensanchadas. Suelen ubicarse en el eje de la carretera, alineadas con el ancho de los carriles, en áreas ampliadas y en los puntos donde confluyen dos fases distintas de tendido del pavimento asfáltico (INVIAS, 2006).

Figura 2 Fisuras en juntas de construcción

Fuente: Adaptado de (INVIAS, 2006).

2.2.2.3 Fisuras por reflexión de juntas

Este daño se produce cuando se extiende una capa de concreto asfáltico sobre una base constituida por losas de concreto rígido. Las fisuras aparecen fundamentalmente por dos razones: la reflejo en la superficie de las juntas entre losas, generando un agrietamiento sistemático, o la extensión hacia arriba de fisuras preexistentes en el concreto rígido, lo que origina un patrón de agrietamiento desordenado en la capa asfáltica superficial (INVIAS, 2006).

Figura 3 Fisuras por reflexión de juntas

Fuente: Adaptado de (INVIAS, 2006).

2.2.2.4 Fisuras en media luna

Se trata de grietas de geometría parabólica vinculadas al desplazamiento de la banca, por lo que frecuentemente van acompañadas de hundimientos.

Figura 4 Fisuras en media luna

Fuente: Adaptado de (INVIAS, 2006).

2.2.2.5 Fisuras de borde

Estas fisuras, con orientación longitudinal o semicircular, se sitúan próximas al límite de la calzada. Su origen se atribuye principalmente a la ausencia de berma o a un desnivel acusado entre esta y la superficie de rodadura (INVIAS, 2006). Generalmente, se localizan dentro de una franja paralela al borde, cuyo ancho puede llegar hasta 0.6 metros cuadrados.

Figura 5 Fisuras de borde

Fuente: Adaptado de (INVIAS, 2006).

2.2.2.6 Fisuras en bloque

Según lo señalado por INVIAS (2006), esta patología consiste en la fragmentación de la superficie asfáltica en porciones de forma aproximadamente rectangular, cuyo lado medio supera los 0,30 m³. Este daño se distingue del agrietamiento tipo piel de cocodrilo por su localización: mientras que el segundo afecta zonas directamente expuestas a las cargas del tráfico, el agrietamiento en bloques suele darse en áreas no sometidas a dichas cargas.

2.2.2.7 Piel de cocodrilo

Este daño se manifiesta como una red de fisuras conectadas que dibujan formas irregulares, localizadas habitualmente en zonas expuestas a cargas vehiculares reiteradas. El proceso de fisuración suele comenzar en la cara inferior de las capas asfálticas, punto donde las tensiones de tracción alcanzan su valor máximo bajo el efecto de dichas cargas (INVIAS, 2006).

Figura 6 Fisura en bloque

Fuente: Elaboración propia

Figura 7 Falla por piel de cocodrilo

Fuente: Elaboración propia

2.2.2.8 Fisuras por deslizamiento de capas

Estas fisuras, que adquieren forma semicircular o de medialuna, presentan curvaturas características en respuesta a los esfuerzos de tracción generados por los neumáticos sobre el pavimento durante maniobras de aceleración o frenado. Su formación obedece principalmente a la acción de deslizamiento y deformación en la superficie del pavimento cuando los vehículos inician la marcha o detienen su avance bruscamente. Es habitual observar este deterioro en zonas montañosas, curvas cerradas o cruces viales (INVIAS).

Figura 8 Fisuras por deslizamiento de capas

Fuente: Adaptado de (INVIAS, 2006).

2.2.2.9 Fisuración incipiente

La fisuración incipiente se define como una agrupación de grietas pequeñas, cercanas entre sí y por lo general no conectadas. Estas afectan únicamente la capa superficial de concreto asfáltico. Por tratarse de un daño de muy baja severidad, este tipo de fisuras carece de niveles de gravedad tipificados (INVIAS, 2006).

Figura 9 Fisuración incipiente

Fuente: Adaptado de (INVIAS, 2006).

2.2.2.10 Ondulaciones

También denominado corrugación o rizado, este daño se define por la aparición de ondulaciones en la superficie del pavimento. Estas ondas suelen disponerse de manera perpendicular al flujo del tráfico y presentan una distancia entre crestas generalmente inferior a 1,0 metro (INVIAS, 2006).

Figura 10 Ondulaciones

Fuente: Adaptado de (INVIAS, 2006).

2.2.2.11 Abultamiento

Este daño hace referencia a elevaciones o protuberancias que emergen en la superficie del pavimento. Tales deformaciones pueden presentarse de forma súbita, afectando áreas reducidas, o evolucionar paulatinamente en zonas más extensas. En ciertas ocasiones, aparecen acompañadas de fisuras (INVIAS, 2006).

Figura 11 Abultamiento

Fuente: Adaptado de (INVIAS, 2006).

2.2.2.12 Hundimiento

Los hundimientos consisten en depresiones puntuales en la superficie del pavimento, caracterizadas por ser desplazamientos verticales súbitos y de pequeña extensión (American Society for Testing and Materials, 2020).

Figura 12
Hundimiento

Fuente: Adaptado de (INVIAS, 2006).

2.2.2.13 Ahuellamiento

El ahuellamiento es una depresión superficial que aparece en la rodada de los neumáticos. En los bordes de dicha depresión puede producirse un levantamiento del pavimento. Este daño es consecuencia de una deformación permanente en alguna de las capas del pavimento o en la subrasante, provocada generalmente por la consolidación o el desplazamiento lateral de los materiales debido a las cargas repetidas del tránsito (American Society for Testing and Materials, 2020).

Figura 13
Ahuellamiento

Fuente: Adaptado de (INVIAS, 2006).

2.2.2.14 Parche

Un parche es una porción de la capa de rodadura donde el material original ha sido reemplazado por uno nuevo con el fin de reparar un sector dañado. Se considera un defecto sin importar su condición, ya que tanto la zona intervenida como sus alrededores generalmente no alcanzan el mismo nivel de comportamiento estructural y superficial que una sección original e intacta del pavimento (American Society for Testing and Materials, 2020).

Figura 14 Parche

Fuente: Adaptado de (INVIAS, 2006).

2.2.2.15 Baches

Este daño corresponde a la disgregación completa de la capa asfáltica, exponiendo los materiales granulares y las capas subyacentes de la estructura del pavimento. Una vez que aparece, el área afectada tiende a ensancharse notablemente con el tiempo debido al efecto erosivo del tráfico, que causa la pérdida progresiva de material. Este tipo de falla es fácilmente identificable e incorpora también los llamados .ºjos de pescado", que son baches de contorno redondeado (INVIAS, 2006).

Figura 15 Baches

Fuente: Adaptado de (INVIAS, 2006).

2.2.2.16 Descascaramiento

Este deterioro consiste en el desprendimiento o pérdida de material de la capa superficial de asfalto, sin afectar a las capas asfálticas inferiores (INVIAS, 2006).

Figura 16 Descascaramiento

Fuente: Adaptado de (INVIAS, 2006).

2.2.3 Procesamiento digital de imágenes

2.2.3.1 Imagen digital

Una imagen digital está formada por un número finito de elementos fundamentales llamados **píxeles**, cada uno de los cuales tiene un valor numérico asociado (típicamente en el intervalo de 0 a 255 para imágenes en escala de grises, o por canal en imágenes de color) y ocupa una localización única dentro de la estructura matricial que conforma la imagen (Gonzalez & Woods, 2018). Desde un punto de vista matemático, una imagen digital se expresa como una función bidimensional:

$$I(x, y) = \text{nivel de intensidad en la coordenada } (x, y) \tag{2.1}$$

donde x e y corresponden a las coordenadas espaciales, mientras que $I(x, y)$ representa un valor numérico que indica la intensidad o el color en dicha posición.

2.2.3.2 Técnicas del procesamiento digital de imágenes

El procesamiento digital de imágenes abarca varias fases esenciales, las cuales se detallan a continuación:

a) **Adquisición:** Esta fase inicial consiste en obtener la imagen digital a través de dispositivos como cámaras fotográficas convencionales, vehículos aéreos no tripulados (UAV) o teléfonos móviles. En esta instancia se determina la resolución, el formato y la calidad de la imagen.

b) **Preprocesamiento:** Esta fase tiene por objeto realzar la calidad de la imagen con el fin de incrementar las posibilidades de éxito en la meta final del análisis. Dicho proceso abarca técnicas como:
   - Supresión de ruido (filtros gaussianos, filtros de mediana)
   - Mejora del contraste (ecualización del histograma)
   - Enfatización de rasgos particulares
   - Estandarización de la iluminación

c) **Segmentación:** Consiste en particionar la imagen en sus partes constituyentes o en los elementos que la integran. Una segmentación apropiada favorece notablemente la solución del problema abordado, mientras que una segmentación deficiente puede ocasionar directamente el fracaso del análisis. Entre las técnicas más habituales se encuentran:
   - Umbralización (thresholding)
   - Detección de contornos (Sobel, Canny)
   - Crecimiento de regiones
   - Algoritmos de agrupamiento (K-means)

d) **Descripción:** El producto del proceso de segmentación es un conjunto de datos que puede incluir los límites de una región o los puntos que la constituyen. Para que esta información pueda ser procesada por el sistema computacional, resulta necesario convertirla a un formato apropiado, procedimiento que se denomina descripción. En esta etapa se extraen atributos como:
   - Superficie, perímetro y geometría de las regiones
   - Momentos invariantes
   - Descriptores de textura
   - Histogramas de orientación de bordes

e) **Reconocimiento:** La etapa de reconocimiento se ocupa de identificar y extraer atributos distintivos que aporten información cuantitativa relevante, o bien, rasgos fundamentales que posibiliten distinguir una categoría de objetos de otra. Para el caso particular de detección de fisuras, se reconocen características como:
   - Dirección de la fisura (longitudinal, transversal, en bloque)
   - Largo y ancho de la grieta
   - Patrón de agrietamiento (piel de cocodrilo, fisuras aisladas)
   - Contexto espacial (emplazamiento en la calzada)

f) **Interpretación:** La fase final consiste en otorgar un significado al conjunto de elementos que han sido detectados y reconocidos previamente. En el marco de esta investigación, la

interpretación asigna a cada región identificada una categoría de daño específica (D00, D10, D20, D40 o REPAIR) y brinda una valoración del grado de severidad.

La Figura 17 presenta de forma esquemática las fases del procesamiento digital de imágenes aplicado a la identificación de grietas en pavimentos.

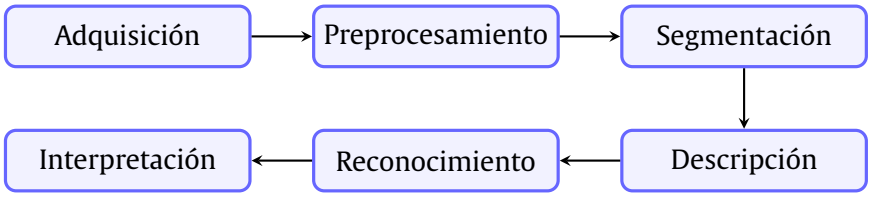


Figura 17: Etapas del procesamiento digital de imágenes

2.2.4 Elementos básicos de inteligencia artificial

2.2.4.1 Inteligencia Artificial

La **Inteligencia Artificial (IA)** constituye la rama que capacita a los sistemas tecnológicos para percibir su entorno, relacionarse con él, solucionar inconvenientes y ejecutar acciones con un fin determinado. Estos sistemas reciben información del exterior, la procesan y elaboran una respuesta. Un rasgo distintivo de la IA es su habilidad para modificar su conducta mediante el análisis de las consecuencias de sus actuaciones previas, lo que le permite funcionar con cierto grado de autonomía (Alba, 2019; Russell & Norvig, 2021).

En esencia, la IA persigue automatizar labores intelectuales que tradicionalmente desempeñan los seres humanos. Como campo de estudio amplio, abarca tanto el **Aprendizaje Automático (Machine Learning)** como el **Aprendizaje Profundo (Deep Learning)**. La Figura **??** exhibe la jerarquía de estos conceptos.

2.2.4.2 Analogía de la neurona biológica y artificial

Se establece una correspondencia entre una neurona biológica y una neurona artificial, observándose que ambas presentan una estructura similar: disponen de entradas (dendritas en el caso biológico), asignan pesos o coeficientes a dichas señales (sinapsis) y generan una salida resultante (axón). La Tabla 1 expone esta comparación de manera ordenada.

Tabla 1: Analogía entre neurona biológica y neurona artificial

Neurona biológica	**Neurona artificial**
Dendritas (reciben señales de otras neuronas)	Entradas $(x_1, x_2, \ldots, x_n)$
Sinapsis (conexiones que modulan la señal)	Pesos sinápticos $(w_1, w_2, \ldots, w_n)$
Cuerpo celular o soma (integra las señales)	Regla de propagación $(\sum w_i x_i)$
Umbral de activación del potencial de acción	Función de activación $(g(\cdot))$
Axón (transmite la señal de salida)	Salida de la neurona (y)

A pesar de que una neurona individual posee una capacidad de cómputo restringida, la interconexión de centenares, miles o incluso millones de ellas en una red permite enfrentar y resolver problemas de gran complejidad (Matich, 2001). La Figura 18 muestra gráficamente esta comparación.

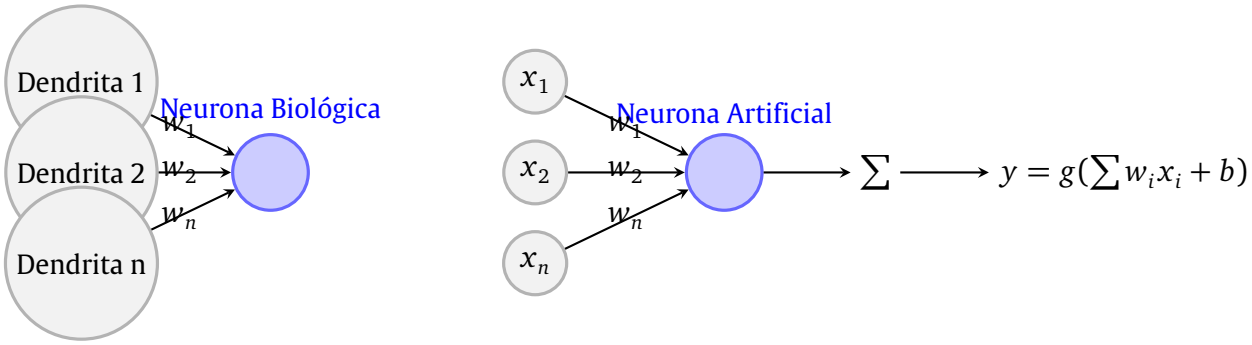


Figura 18: Analogía entre neurona biológica y neurona artificial

2.2.4.3 Definición de una red neuronal artificial

Una **Red Neuronal Artificial (RNA)** es un modelo computacional que imita el comportamiento de las neuronas biológicas y la organización del cerebro, concebido para resolver una extensa variedad de problemas. Gracias a su flexibilidad, una misma red puede adecuarse para llevar a cabo distintos tipos de tareas (Haykin, 2009; Goodfellow et al., 2016).

De forma análoga a los sistemas neuronales biológicos, los componentes esenciales de una RNA son las neuronas artificiales. Cada una de estas unidades actúa como un dispositivo de cálculo elemental, que recibe un conjunto de datos de ingreso y produce una única salida o respuesta como resultado.

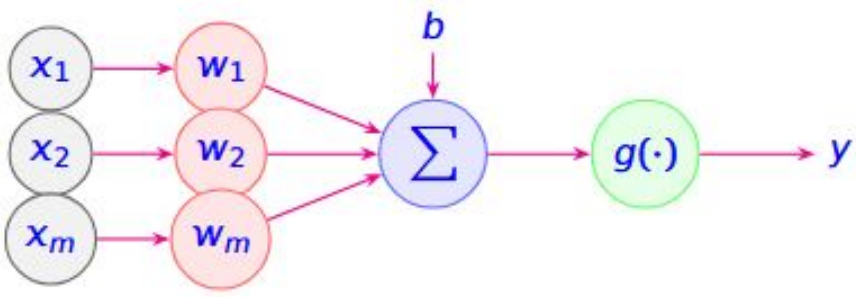
2.2.4.4 Elementos de una red neuronal artificial

Los componentes que participan en una neurona artificial son los siguientes:

1. **Conjunto de entradas:**
$$x_i \quad \text{con} \quad i = 1, 2, \ldots, m \tag{2.2}$$
2. **Pesos sinápticos:**
$$w_i \quad \text{con} \quad i = 1, 2, \ldots, m \tag{2.3}$$
3. **Regla de propagación:** Integra las entradas y los pesos sinápticos. Habitualmente se emplea la sumatoria ponderada:
$$s = \sum_{i=1}^{m} w_i x_i \tag{2.4}$$
4. **Función de activación:** Es una función de s y de una constante b (llamada umbral o sesgo), que entrega la salida y de la neurona:
$$y = g(s + b) = g\left(\sum_{i=1}^{m} w_i x_i + b\right) \tag{2.5}$$

La Figura presenta el modelo esquemático de una neurona artificial con todos sus componentes.

Figura 19 Modelo esquemático de una neurona artificial



Fuente: Elaboración propia

Las funciones de activación más frecuentemente empleadas en redes neuronales se muestran en la Tabla 2.

Tabla 2: Funciones de activación más utilizadas en redes neuronales

Función	Ecuación	Rango de salida
Sigmoide	$\sigma(x) = \frac{1}{1+e^{-x}}$	$(0, 1)$
Tangente hiperbólica (tanh)	$\tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$	$(-1, 1)$
ReLU	$\text{ReLU}(x) = \text{máx}(0, x)$	$[0, \infty)$
Leaky ReLU	$\text{LReLU}(x) = \text{máx}(0.01x, x)$	$(-\infty, \infty)$
Softmax	$\text{softmax}(z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}}$	$(0, 1)$ (probabilidades)

2.2.5 Arquitectura o tipología de una red neuronal artificial

Las redes neuronales pueden agruparse según su arquitectura, la cual define el modo en que se conectan las neuronas entre sí. Las principales arquitecturas se describen a continuación.

2.2.5.1 Perceptrón

Esta corresponde a la disposición más elemental: suele estar compuesta por dos unidades de entrada y una de salida. Los datos ingresan por las unidades de entrada, se dirigen hacia la unidad de salida, donde se les aplica una sumatoria ponderada y posteriormente una función de activación, para finalmente producir el valor resultante. La Figura **??** muestra esta arquitectura.

2.2.5.2 Redes feed forward

Estas redes constituyen una evolución del perceptrón básico. Se caracterizan por estar dispuestas en múltiples capas de neuronas, donde cada neurona de una capa se conecta con todas las neuronas de la capa subsiguiente, conformando una conexión **"totalmente conectada"** o *densa*. Incorporan al menos una capa oculta y el flujo de información es unidireccional, desde la entrada hacia la salida (**alimentación hacia adelante** o *feed-forward*).

La Figura **??** ilustra la arquitectura de una red *feed-forward* con una capa oculta.

El entrenamiento de estas redes se realiza habitualmente mediante el algoritmo de **retropropagación** (*back-propagation*), que modifica los pesos sinápticos minimizando el error de predicción. El algoritmo se expresa en la Ecuación 2.6.

$$\Delta w_{ij} = -\eta \frac{\partial E}{\partial w_{ij}} \tag{2.6}$$

donde:

- Δw_{ij} es el ajuste del peso que vincula la neurona i con la neurona j
- η es la tasa de aprendizaje (*learning rate*)
- E es la función de error o pérdida

2.2.5.3 Redes Neuronales Convolucionales (CNN)

Las **Redes Neuronales Convolucionales** (CNN, por sus siglas en inglés *Convolutional Neural Networks*), también llamadas **Redes Neuronales Convolucionales Profundas** (DCNN), están concebidas primordialmente para el procesamiento de imágenes. Su arquitectura se divide en dos bloques funcionales.

1. **Bloque de extracción de características:** Integrado fundamentalmente por capas convolucionales y de *pooling*, se ocupa de extraer y detectar patrones visuales en la imagen.
2. **Bloque de clasificación:** Tiene como propósito categorizar la información procesada que recibe del primer bloque, usualmente mediante capas totalmente conectadas (*fully connected*).

Las operaciones fundamentales en una CNN se definen mediante las siguientes ecuaciones:

- **Operación de convolución:**

$$(f * g)(x, y) = \sum_{i=-k}^{k} \sum_{j=-k}^{k} f(i, j) \cdot g(x - i, y - j) \tag{2.7}$$

- **Operación de pooling (max-pooling):**

$$p_{i,j} = \max_{m,n \in \text{ventana}} a_{i+m,j+n} \tag{2.8}$$

La Tabla 3 describe las capas más habituales en una CNN y su cometido.

Tabla 3: Capas principales en una arquitectura CNN

Tipo de capa	Función
Capa Convolucional	Aplica filtros (kernels) a la imagen de entrada para extraer características como bordes, texturas y patrones
Capa de Activación (ReLU)	Introduce no-linealidad a la red, reemplazando valores negativos por cero
Capa de Pooling	Reduce la dimensión espacial de los mapas de características, disminuyendo la cantidad de parámetros
Capa Totalmente Conectada (FC)	Conecta todas las neuronas de la capa anterior con todas las de la capa siguiente, realizando la clasificación final
Capa Dropout	Desactiva aleatoriamente un porcentaje de neuronas durante el entrenamiento para evitar el sobreajuste

2.2.6 Mecanismos de aprendizaje

El aprendizaje en una red neuronal es el proceso mediante el cual esta modifica los valores de sus pesos como respuesta a los datos de ingreso que recibe. Este proceso implica fundamentalmente la alteración de las conexiones entre neuronas, lo que puede traducirse en la supresión, ajuste o generación de nuevos enlaces sinápticos artificiales (Matich, 2001).

La Figura 20 clasifica los principales tipos de aprendizaje en redes neuronales.

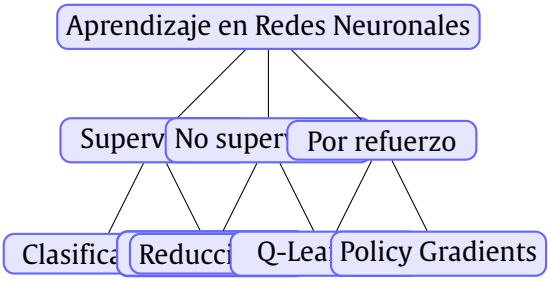


Figura 20: Tipos de aprendizaje en redes neuronales

2.2.6.1 Aprendizaje supervisado

Esta modalidad de aprendizaje se caracteriza por la intervención de un agente externo (supervisor o instructor) que orienta el proceso de entrenamiento. Dicho supervisor indica cuál debe ser la respuesta correcta de la red ante cada dato de entrada suministrado. El sistema confronta la

salida generada por la red con la salida esperada y, si detecta una diferencia, ajusta los pesos de las conexiones internas para disminuir el error y acercar progresivamente la salida de la red al resultado deseado (Matich, 2001).

Las funciones de error más empleadas en aprendizaje supervisado se presentan en la Tabla 4.

Tabla 4: Funciones de error más comunes en aprendizaje supervisado

Función de error	**Ecuación**
Error Cuadrático Medio (MSE)	$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$
Error Absoluto Medio (MAE)	$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}\lvert y_i - \hat{y}_i\rvert$
Entropía Cruzada Binaria	$L = -\frac{1}{n}\sum_{i=1}^{n}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1-\hat{y}_i)]$
Entropía Cruzada Categórica	$L = -\sum_{i=1}^{n} y_i \log(\hat{y}_i)$

2.2.6.2 Aprendizaje no supervisado

En el aprendizaje no supervisado, también llamado autosupervisado, las redes neuronales modifican sus pesos internos sin la participación de una guía externa. Durante este proceso, la red no recibe ninguna retroalimentación acerca de si la salida que produce ante una entrada determinada es acertada o errónea.

En algunos enfoques, la salida refleja el nivel de semejanza o familiaridad entre la información de ingreso actual y los datos procesados con anterioridad. En otros casos, la red lleva a cabo una tarea de *clustering* o agrupamiento, donde su objetivo es identificar y asignar la entrada a una categoría particular. Es la propia red la que debe descubrir dichas categorías y sus patrones distintivos a partir de las correlaciones que encuentra en los datos suministrados (Matich, 2001).

"La inteligencia consiste no solo en el conocimiento, sino también en la destreza de aplicar los conocimientos en la práctica."

— Aristóteles

3 Método de la investigación

El enfoque metodológico adoptado en esta investigación es el deductivo.

3.1 Enfoque.

La investigación posee un carácter cuantitativo, dado que se sustenta en la recopilación de datos provenientes de un muestreo representativo. La variable central del estudio, correspondiente a los deterioros presentes en pavimentos flexibles, será cuantificada empleando instrumentos tales como registros videográficos y fotografías tomadas en un entorno determinado. El examen de dichas imágenes permitirá obtener evidencias para, posteriormente, formular conclusiones vinculadas a las hipótesis planteadas.

3.2 Alcance

El alcance de una investigación determina los resultados que se prevé alcanzar y, consecuentemente, define la estrategia metodológica que se aplicará para lograrlos.

Este trabajo presenta un enfoque correlacional, ya que su propósito central consiste en determinar el grado de relación o asociación (no causal) entre dos o más variables. Se distingue por medir inicialmente las variables involucradas y, posteriormente, mediante pruebas de hipótesis correlacionales y herramientas estadísticas, estimar el nivel de correlación existente entre ellas.

3.3 Diseño de investigación

La investigación es de tipo no experimental, dado que los acontecimientos y las variables analizadas ya han sucedido con anterioridad, y el estudio se limita a observar dichas variables y sus interrelaciones dentro de su ámbito natural. Este tipo de diseño se define como aquel en el que no se manipulan deliberadamente las variables; es decir, no se alteran intencionalmente las variables independientes para examinar sus efectos sobre otras variables. En lugar de ello, se analizan los fenómenos tal como se manifiestan en la realidad.

3.4 Población y muestra

3.4.1 Población

En este estudio, la población está conformada por 1871 imágenes, correspondientes al total de fotografías capturadas en el tramo Socos-Licapa para el desarrollo de la presente investigación.

3.4.2 Muestra

Se emplea un muestreo intencional, utilizando la totalidad de la población disponible. De las 1871 imágenes, todas constituyen la muestra de la investigación.

3.5 Hipótesis

3.5.1 Hipótesis general

Un modelo YOLOv12 entrenado con 1871 imágenes es capaz de evaluar cinco tipos de daños en pavimentos flexibles del tramo Socos-Licapa con una precisión mAP@50 superior al 70 %.

3.5.2 Hipótesis específicas

1. El modelo YOLOv12 reduce el tiempo de evaluación de daños en al menos un 80 % en comparación con la inspección visual manual.
2. La calidad y cantidad del dataset (1871 imágenes etiquetadas) y la arquitectura YOLOv12-s influyen significativamente en la precisión final del modelo.
3. Es posible identificar y clasificar las cinco clases de daños (D00, D10, D20, D40, REPAIR) con una precisión mAP@50 superior al 65 % para cada clase.

3.6 Operacionalización de variables

3.6.1 Variables

Las variables constituyen elementos de datos que, una vez organizados y presentados como información, sirven de base para elaborar descripciones preliminares. Dichas descripciones son posteriormente evaluadas e interpretadas en el análisis, integrándose y sintetizándose en conclusiones que, utilizadas como premisas, permiten contrastar cada subhipótesis con aquellas con las que guardan una relación directa en el marco de la investigación.

En todo proceso investigativo participan fundamentalmente dos tipos de variables: las independientes y las dependientes.

3.6.2 Indicadores

Debido a que las variables no son directamente observables, resulta necesario establecer procedimientos que posibiliten su medición de manera indirecta a través de manifestaciones externas, empíricas y susceptibles de ser observadas. Estas manifestaciones concretas son lo que denominamos indicadores.

Tabla 5: Operacionalización de variables

VARIABLE		Definición conceptual	Dimensiones	Indicadores	Unidad de medida
Tipo	**Nombre**				
Independiente (Causa)	Modelo YOLOv12 con Deep Learning	Algoritmo de deep learning de última generación para detección automatizada de objetos en una sola pasada (You Only Look Once, Versión 12)	Arquitectura de red	Número de capas	Und.
				Parámetros (M)	Millones
			Hiperparámetros	Tasa de aprendizaje (learning rate)	Valor numérico
				Número de épocas	Und.
Dependiente (Efecto)	Detección de daños en pavimentos flexibles	Identificación, localización y clasificación de fallas en pavimentos flexibles mediante procesamiento de imágenes		mAP@50	%
			Precisión de detección	mAP@[50:95]	%
				Precisión (Precision)	%
			Clasificación por tipo de daño	Recall (Sensibilidad)	%
				F1-Score	%

3.7 Técnicas e instrumentos

La fase inicial contempla el montaje de una cámara convencional en la parte delantera de un vehículo, garantizando que la grabación se realice de manera centrada sobre la calzada. Luego, durante la etapa de filtrado y procesamiento de los vídeos, el propósito fundamental consiste en extraer fotogramas individuales a partir de la secuencia registrada. Estas imágenes son posteriormente sometidas a un tratamiento para suprimir las zonas que carecen de información útil para la detección y valoración de los daños presentes en el pavimento flexible. Los equipos empleados en esta investigación comprenden una cámara dotada de sistema de posicionamiento GPS.

3.7.1 Técnicas

Revisión documental, procesamiento digital de imágenes.

3.7.2 Instrumentos

Formatos de registro, dispositivo fotográfico digital.

3.8 Desarrollo del trabajo de tesis

3.8.1 Recolección y obtención de la dataset

Para la recolección de imágenes destinadas a la detección de grietas en pavimentos se requiere contar con materiales básicos como una cámara digital de alta resolución o un teléfono móvil de buena calidad, así como un dron (UAV) en caso de captar superficies extensas desde vista aérea. También es recomendable disponer de un trípode o estabilizador para obtener imágenes nítidas, además de una laptop o computadora para almacenar y organizar la información recolectada. El procedimiento consiste en planificar el tramo a inspeccionar donde que para la presente investigación se recolecto la dataset del tramo Socos Licapa de la vía los Libertadores Wari, se verificó las condiciones de seguridad en la zona, configurando adecuadamente los equipos de captura y registrando fotografías a diferentes ángulos, alturas y distancias para obtener variedad en el conjunto de datos las fotografías se organizan en carpetas clasificadas y se depuran eliminando tomas borrosas o inservibles, dejando listo el material para su posterior etiquetado y uso en modelos de detección automática como YOLO V12, donde posteriormente procederá al etiquetado de las imágenes tomadas en campo y filmada cámara de buena calidad utilizando una camioneta Hilux donde se recorrió el tramo Socos Licapa hasta el puente Rumichaca límite entre Huancavelica y Ayacucho.

Figura 21 Recolección de dataset en el tramo Socos Licapa de la Vía los Libertadores Wari

Fuente: Elaboración propia

Figura 22 Recolección de dataset en el tramo Socos Licapa de la Vía los Libertadores Wari

Fuente: Elaboración propia

Figura 23
Recolección de dataset en el tramo Socos Licapa de la Vía los Libertadores Wari

Fuente: Elaboración propia

Figura 24
Recolección de dataset en el tramo Socos Licapa de la Vía los Libertadores Wari

Fuente: Elaboración propia

Figura 25
Recolección de dataset en el tramo Socos Licapa de la Vía los Libertadores Wari

Fuente: Elaboración propia

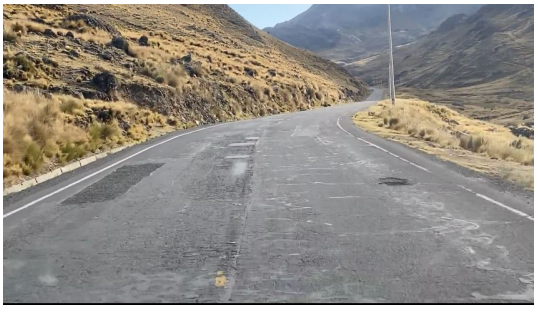
Figura 26
Recolección de dataset en el tramo Socos Licapa de la Vía los Libertadores Wari

Fuente: Elaboración propia

Figura 27
Recolección de dataset en el tramo Socos Licapa de la Vía los Libertadores Wari

Fuente: Elaboración propia

Figura 28
Recolección de dataset en el tramo Socos Licapa de la Vía los Libertadores Wari



Fuente: Elaboración propia

Figura 29
Recolección de dataset en el tramo Socos Licapa de la Vía los Libertadores Wari

Fuente: Elaboración propia

Figura 30
Recolección de dataset en el tramo Socos Licapa de la Vía los Libertadores Wari

Fuente: Elaboración propia

Figura 31
Recolección de dataset en el tramo Socos Licapa de la Vía los Libertadores Wari

Fuente: Elaboración propia

Figura 32
Recolección de dataset en el tramo Socos Licapa de la Vía los Libertadores Wari

Fuente: Elaboración propia

Figura 33
Recolección de dataset en el tramo Socos Licapa de la Vía los Libertadores Wari

Fuente: Elaboración propia

Figura 34
Recolección de dataset en el tramo Socos Licapa de la Vía los Libertadores Wari

Fuente: Elaboración propia

Figura 35 Recolección de dataset en el tramo Socos Licapa de la Vía los Libertadores Wari

Fuente: Elaboración propia

Figura 36 Recolección de dataset en el tramo Socos Licapa de la Vía los Libertadores Wari

Fuente: Elaboración propia

3.8.2 Etiquetado y labeling de dataset para entrenamiento

En este estudio se utilizaron tres conjuntos de datos de detección de imágenes del tramo de Socos Licapa, para entrenamiento para validación y para la prueba, modelar con precisión e iterar necesita mayor entrenamiento para el aprendizaje del modelo para posteriormente validarlo y utilizarlo en la prueba para encontrar la precisión de mPA del modelo, este enfoque se adoptó para evitar que el modelo se vuelva sesgado hacia entornos específicos, abordando así el posible problema de una dependencia excesiva en un conjunto de datos particular. Al integrar estos conjuntos de datos distintos, recopilados bajo diversas condiciones y entornos, el modelo fue entrenado para detectar escenarios de caídas de una manera más generalizada.

Todos los conjuntos de datos fueron etiquetados de forma dicotómica, diferenciando entre el estado normal y el estado de caída. Sin embargo, dado que el objetivo principal de este estudio fue mejorar la precisión en la detección de caídas, se determinó que entrenar el modelo con imágenes del estado normal no contribuiría significativamente a este objetivo. En consecuencia, para minimizar posibles errores de entrenamiento y optimizar el rendimiento del modelo, la estructura de etiquetado se modificó para enfocarse exclusivamente en una 5 clases, la clase D00, D10, D20, D40 y la clase REPAIR. Esta reorganización busca mejorar la capacidad del modelo para identificar incidentes de caída con mayor precisión y reducir las tasas de clasificación errónea, mejorando así la precisión global de la detección.

Este estudio utilizó un total de 1871 imágenes, de las cuales 1582 se destinaron al entrenamiento, 197 a la validación y las 92 para la evaluación final de precisión. Se utilizó el programa de ROBOFLOW para el etiquetado y labeling de las imágenes para el procesamiento y entrenamiento del modelo

3.8.2.1 Clases de fallas

Según ASTM D6433 (PCI – Pavement Condition Index), el Ministerio de Transportes y Comunicaciones (MTC Perú), Administración Federal de Carreteras de EE.UU. (FHWA) y Asociación Americana de Oficiales Estatales de Carreteras y Transporte (AASHTO)

Las fallas están clasificadas en :

Pavimento rígido (concreto)

Fallas típicas:

- Grietas por contracción
- Fauces por bombeo
- Falla de juntas
- Spalling
- Descascaramiento

Pavimento flexible (asfalto)

Fallas típicas:

- Grietas longitudinales
- Grietas en bloque
- Ahuellamiento
- Piel de cocodrilo
- Exudación
- Desintegración (potholes)

Identificar la morfología de la falla

Fallas típicas:

Tabla 6: Morfología de la falla

Causa	Tipos de falla
Fatiga estructural	Piel de cocodrillo
Mal drenaje	Baches, pérdida de soporte
Tráfico pesado	ahuellamiento
envejecimiento	grietas en bloque
Deficiente compactación	Hundimiento

Fuente: Elaboración propia

Ubicación de la falla

La ubicación ayuda a decidir qué tipo es:

Tabla 7: Clases de falla en el tramo Socos Licapa

Ubicación	sospecha
en huella de rodadura	ahuellamiento
Cerca de bordes	grietas de borde
entre juntas	falla de junta
superficie completa	rigidez o envejecimiento

***Fuente**:* Elaboración propia

Elección de las clases de falla para el tramo Socos Licapa

Tabla 8: Clases de falla en el tramo Socos Licapa

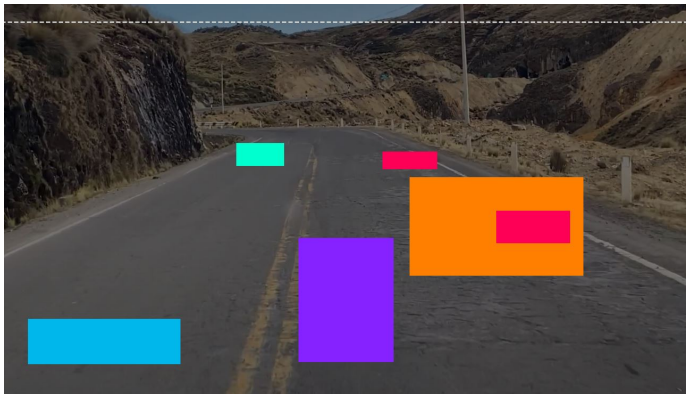
Clases	Tipo de Falla
D00	Falla de Piel de cocodrillo
D10	Parches
D20	Falla de fisura en bloque
D40	Falla de fisura en borde
REPAIR	Reparación de falla extrema

***Fuente**:* Elaboración propia

Figura 37 Clases de fallas mas resaltantes elegidas para el tramo Socos Licapa

Fuente: Elaboración propia

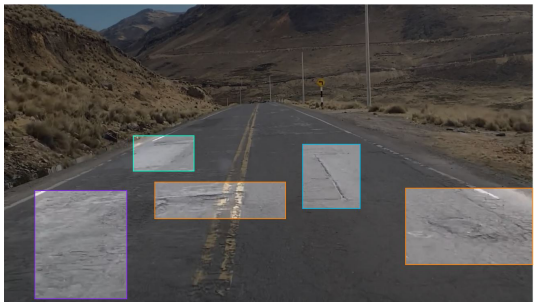
Figura 38 Capas para las clases de fallas en el tramo Socos Licapa



Fuente: Elaboración propia

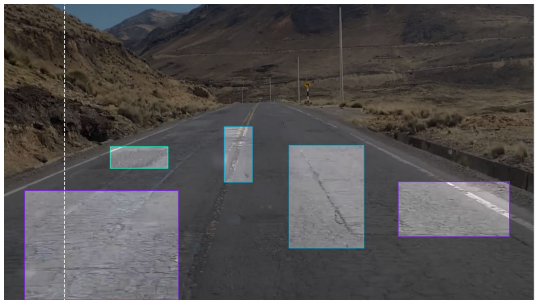
3.8.2.2 Clases para el tramo Socos Licapa

Figura 39 Etiquetado de las clases de fallas



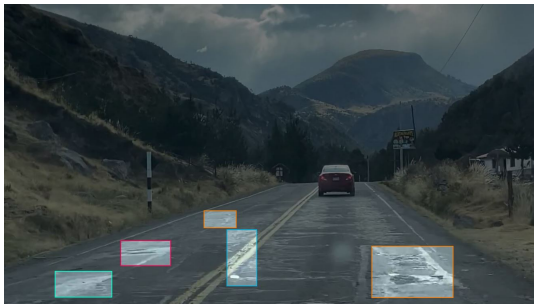
Fuente: Elaboración propia

Figura 40 clasificacion de fallas tramo Socos Licapa



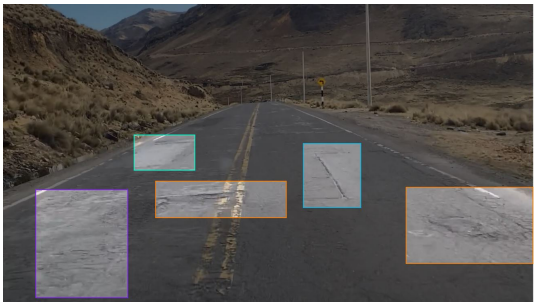
Fuente: Elaboración propia

Figura 41 Etiquetado de las clases de fallas



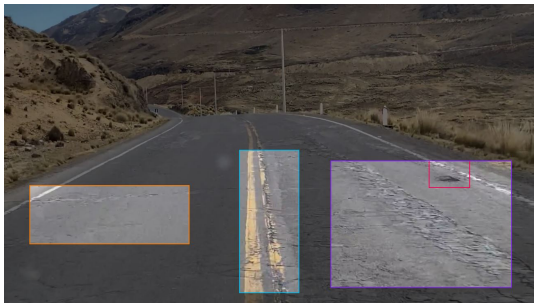
Fuente: Elaboración propia

Figura 42 clasificacion de fallas tramo Socos Licapa



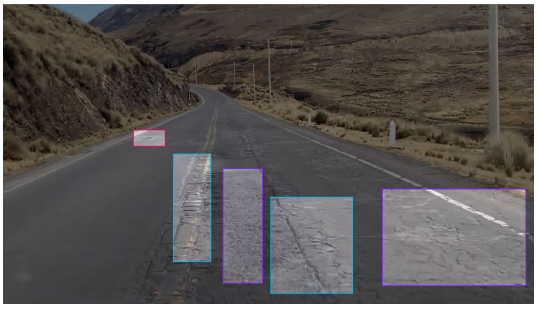
Fuente: Elaboración propia

Figura 43 Etiquetado de las clases de fallas



Fuente: Elaboración propia

Figura 44 clasificacion de fallas tramo Socos Licapa

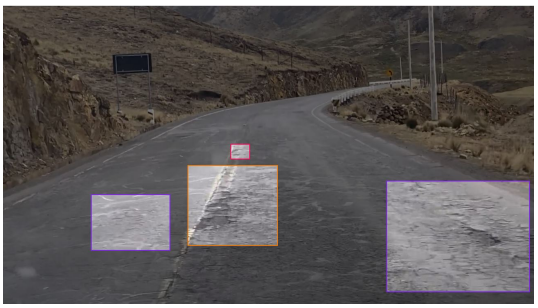


Fuente: Elaboración propia

Figura 45 Etiquetado de las clases de fallas

Fuente: Elaboración propia

Figura 46 clasificacion de fallas tramo Socos Licapa

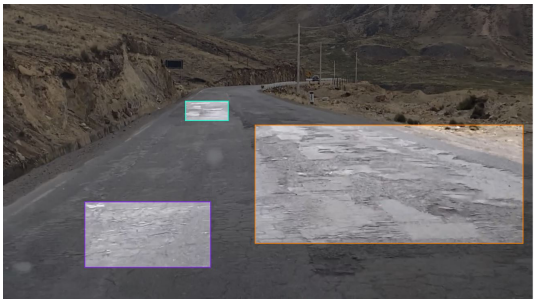


Fuente: Elaboración propia

Figura 47 Etiquetado de las clases de fallas

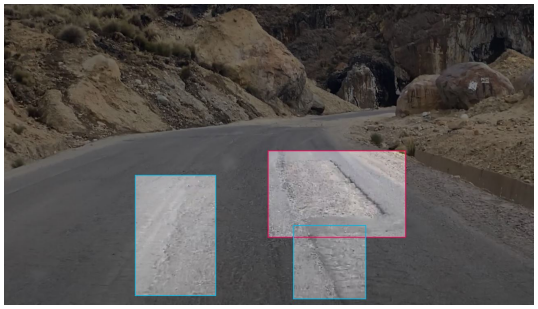
Fuente: Elaboración propia

Figura 48 clasificacion de fallas tramo Socos Licapa



Fuente: Elaboración propia

Figura 49 Etiquetado de las clases de fallas



Fuente: Elaboración propia

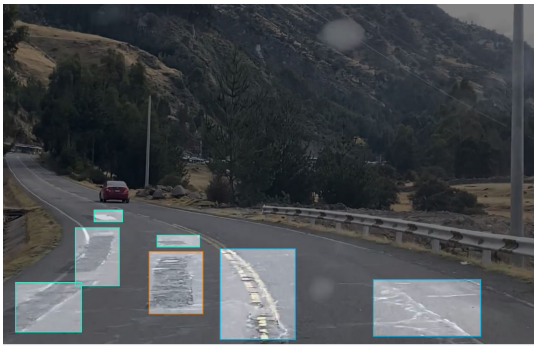
Figura 50 clasificacion de fallas tramo Socos Licapa

Fuente: Elaboración propia

Figura 51 Etiquetado de las clases de fallas

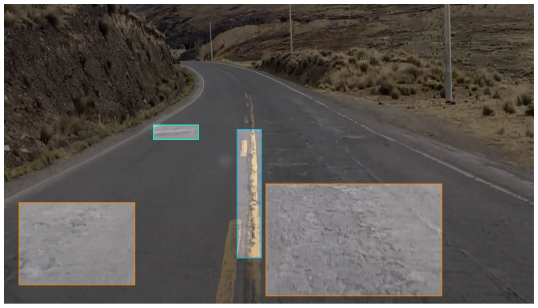
Fuente: Elaboración propia

Figura 52 clasificacion de fallas tramo Socos Licapa



Fuente: Elaboración propia

Figura 53 Etiquetado de las clases de fallas



Fuente: Elaboración propia

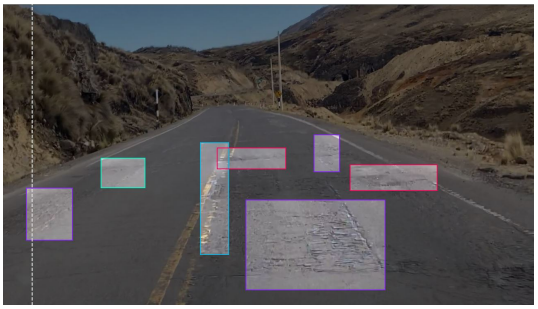
Figura 54 clasificacion de fallas tramo Socos Licapa

Fuente: Elaboración propia

Figura 55 Etiquetado de las clases de fallas

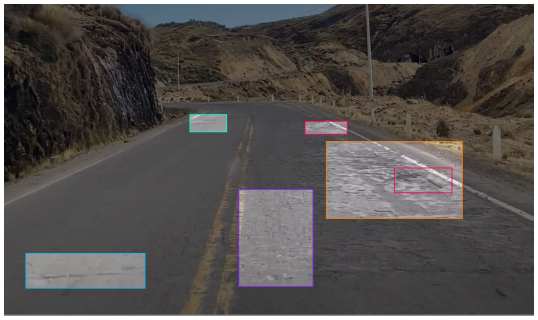
Fuente: Elaboración propia

Figura 56 clasificacion de fallas tramo Socos Licapa



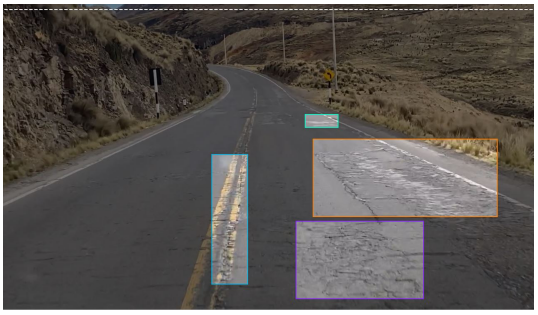
Fuente: Elaboración propia

Figura 57 Etiquetado de las clases de fallas



Fuente: Elaboración propia

Figura 58 clasificacion de fallas tramo Socos Licapa



Fuente: Elaboración propia

3.8.3 Evaluación de medida del modelo

En las tareas de detección de objetos, las cuatro métricas principales para evaluar el rendimiento del modelo son la Precisión Media (mPrecision), la Sensibilidad Media (mRecall), la Puntuación F1 Media (mF1) y la Precisión Media Promedio (mAP). Estas métricas se utilizan comúnmente para evaluar la capacidad del modelo para detectar diversas clases de objetos, incluyendo el fondo y las clases de grietas.

3.8.3.1 Precisión

La precisión indica la proporción de muestras detectadas correctamente y clasificadas como clase positiva (por ejemplo, grietas) respecto al total de muestras que el modelo predijo como clase positiva. TP (Verdaderos Positivos) son los casos positivos identificados correctamente. FP (Falsos Positivos) son los casos negativos identificados incorrectamente como positivos. FN (Falsos Negativos) son los casos positivos que el modelo no logró detectar. La fórmula para calcular la precisión es la siguiente:

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{3.1}$$

Donde:

TP = Verdaderos positivos

FP = Falsos positivos

3.8.3.2 Recall

La sensibilidad (Recall) se refiere a la proporción de muestras correctamente detectadas como clase positiva entre todas las muestras que realmente pertenecen a la clase positiva. La fórmula para calcular la sensibilidad es la siguiente:

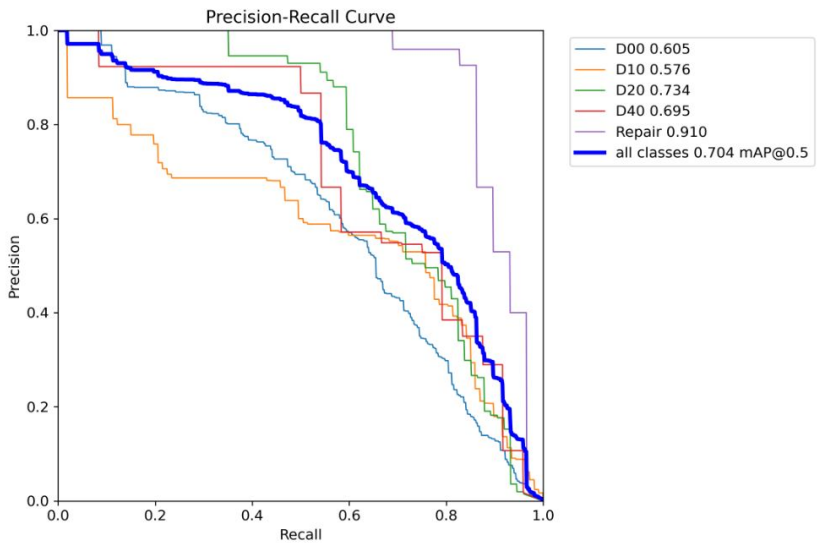
$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3.2}$$

Donde:

TP = Verdaderos positivos

FN = Falsos negativos

Figura 59 Mean Average Precisión (mAP)



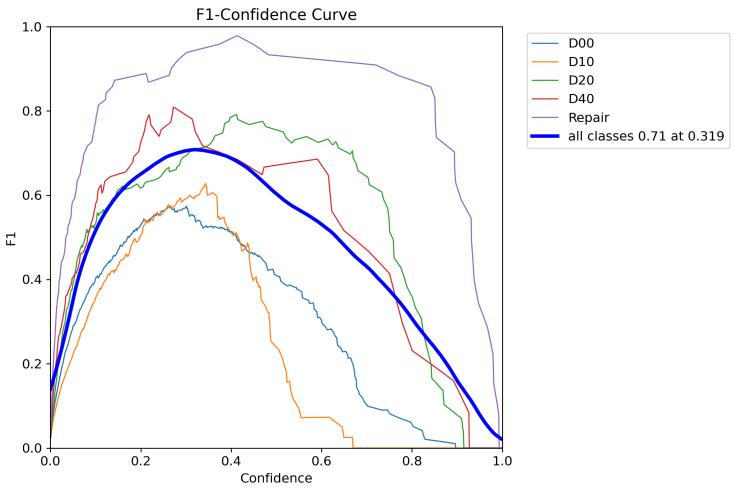
Fuente: Elaboracion propia con Python.

3.8.3.3 F1 score

El puntaje F1 (F1 score) es la media armónica de la precisión y la sensibilidad, y proporciona una evaluación integral del rendimiento del modelo al considerar ambas métricas. La fórmula para calcular el puntaje F1 es la siguiente:

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} \tag{3.3}$$

Figura 60 Mean Average Precisión (mAP)



Fuente: Elaboracion propia con Python

3.8.3.4 Mean Average Precisión (mAP)

La Precisión Media Promedio (mAP) es una métrica de evaluación ampliamente utilizada en tareas de detección de objetos. Representa el promedio de la Precisión Promedio (AP) calculada para todas las clases. La AP cuantifica el rendimiento de detección del modelo para una sola clase, mientras que la mAP proporciona una evaluación promedio del rendimiento de detección del modelo a través de múltiples clases:

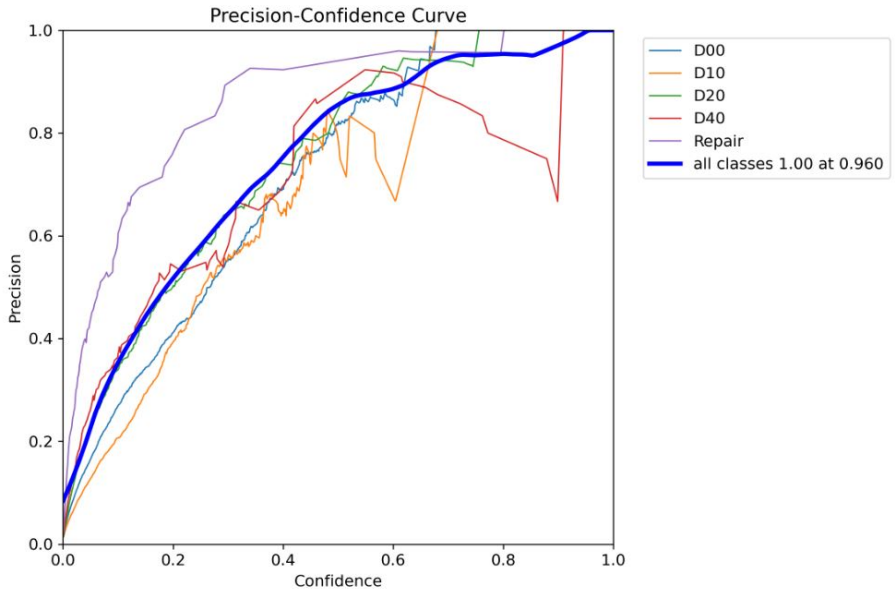
$$\text{mAP} = \frac{1}{N}\sum_{i=1}^{N} \text{AP}_i \tag{3.4}$$

Donde:

N= Número de clases

$AP_{(i)}$ representa la precisión promedio de la i-ésima clase.

Figura 61 Mean Average Precisión (mAP)



Fuente: Elaboracion propia con Python

3.8.3.5 Hallando la confusión matriz

la confusión matriz compara las predicciones del modelo con las clases reales de los objetos para evaluar su rendimiento y si acerto detectar una clase.

Figura 62 Confusión matriz

Fuente: Elaboracion propia con Python

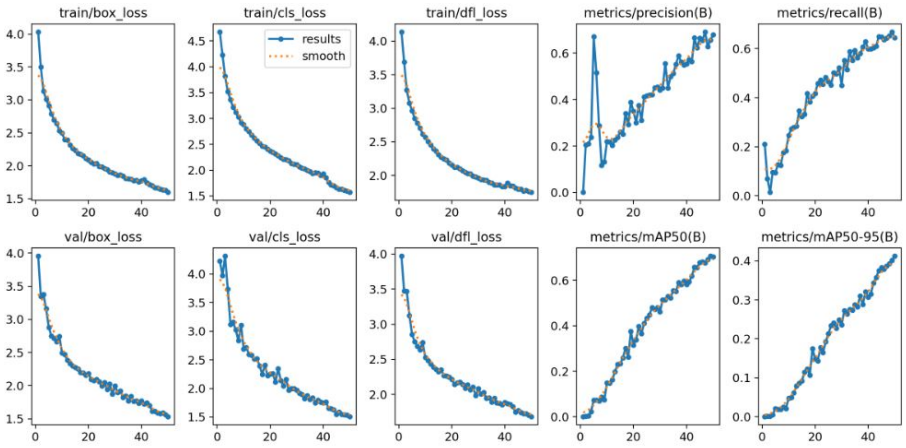
Figura 63 Confusión matriz normalizada

Fuente: Elaboracion propia con Python

3.8.3.6 Hallando resultados de entrenamiento, validación y precisión

las gráfica de resultados nos indica, el desempeño del entrenamiento, los resultados de la validación y los resultados de la precisión con la cual cuenta el modelo de YOLO V12.

Figura 64
Resultados de entrenamiento, validación y precisión del modelo YOLO V12

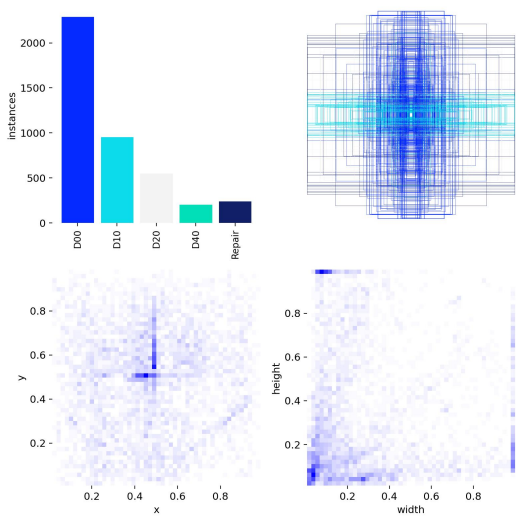


Fuente: Elaboracion propia con Python

3.8.3.7 instancias versus clases

El gráfico Instances vs Classes muestra cuántas instancias (objetos anotados) hay para cada clase dentro del dataset usado para entrenar o evaluar el modelo. Es decir, compara la cantidad de ejemplos reales disponibles por cada tipo de objeto que el modelo debe aprender a detectar.

Figura 65
Instancias versus clases

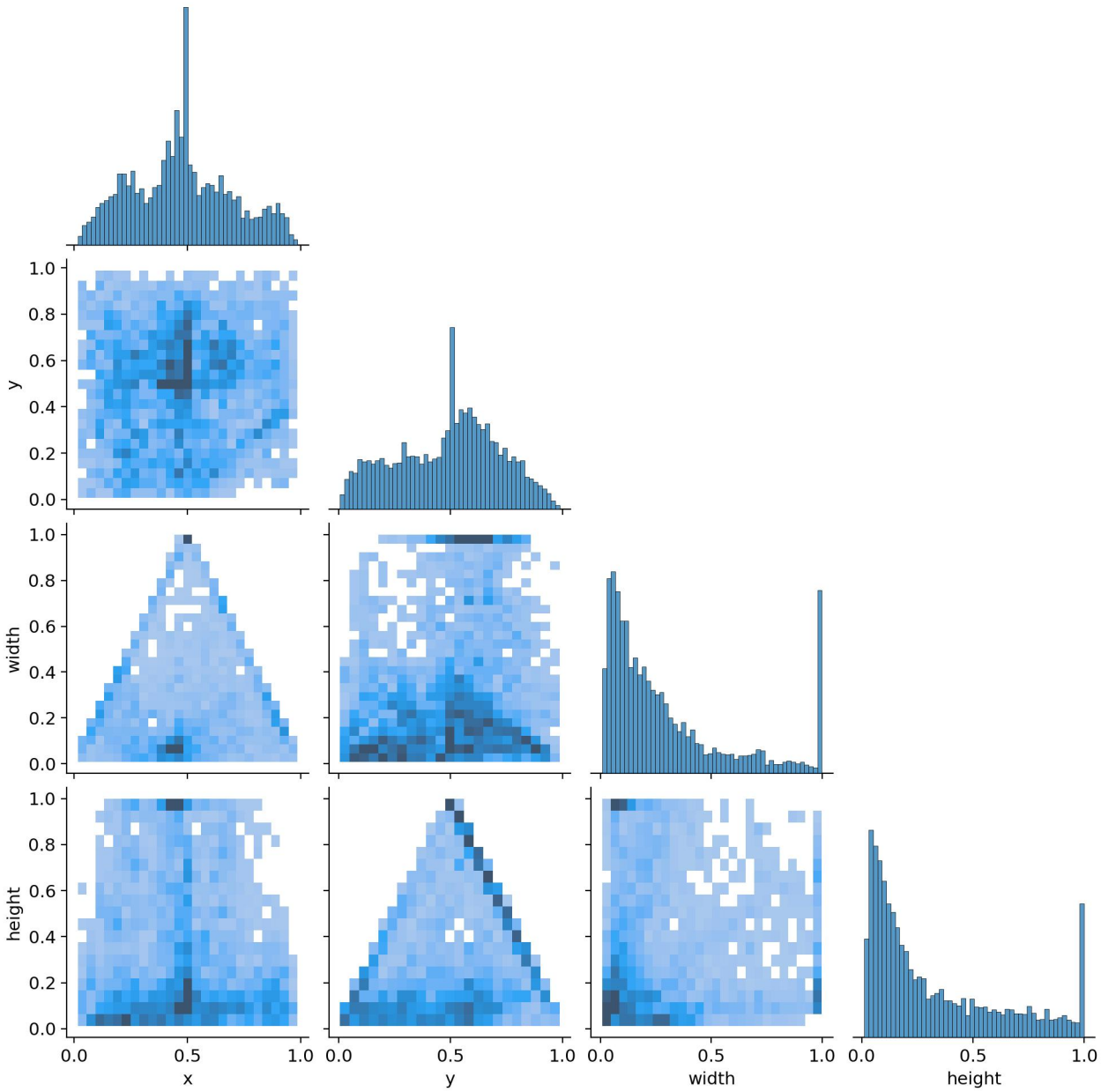


Fuente: Elaboracion propia con Python

3.8.3.8 Distribución espacial de las cajas anotadas

Estos gráficos muestran la distribución de las coordenadas y tamaños de las cajas anotadas (bounding boxes) de tu dataset. Son usados para verificar cómo están distribuidos espacialmente los objetos dentro de las imágenes.

Figura 66 Distribución de la cajas anotadas

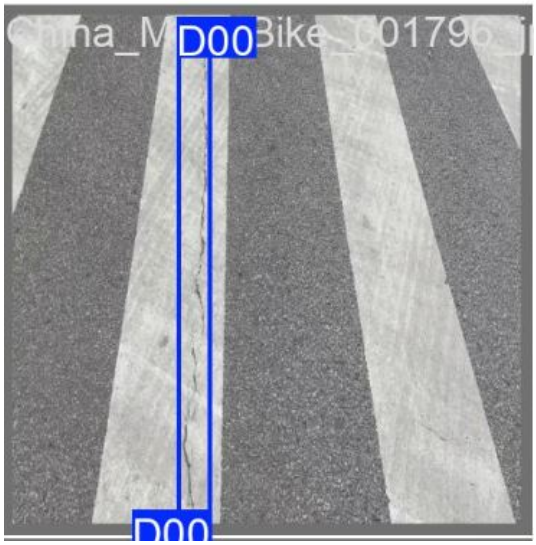


Fuente: Elaboracion propia con Python

3.8.3.9 Entrenamiento del modelo de la Red Neuronal

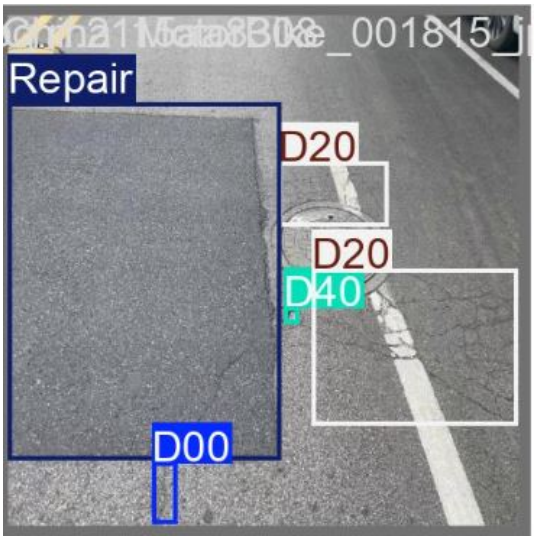
Para el entrenamiento de la red Neuronal y para el diseño de al arquitectura del modelo se utilizaron 1871 imagenes seleccionados clasificados en 5 clases de fallas en pavimentos para el desarrollo de la investigación

Figura 67 Imágenes utilizadas para el entrenamiento del modelo YOLO V12



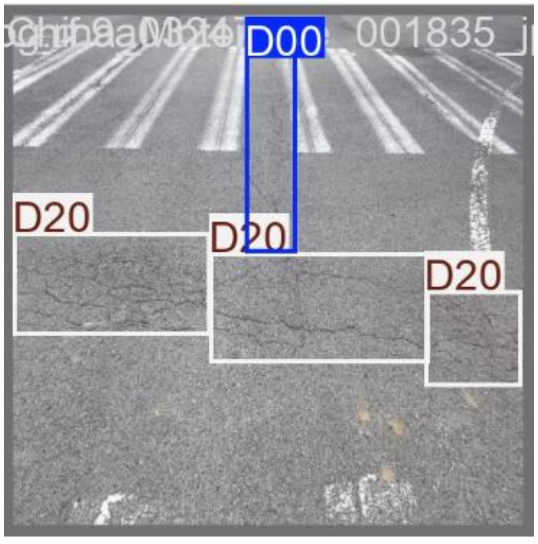
Fuente: Elaboración propia

Figura 68 Imágenes utilizadas para el entrenamiento del modelo YOLO V12



Fuente: Elaboración propia

Figura 69 Imágenes utilizadas para el entrenamiento del modelo YOLO V12

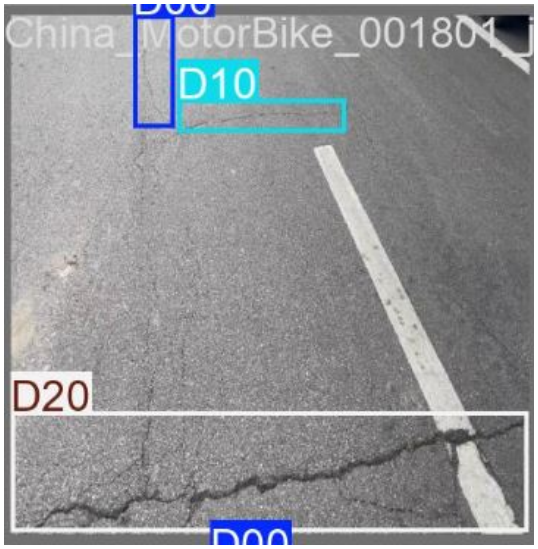


Fuente: Elaboración propia

Figura 70 Imágenes utilizadas para el entrenamiento del modelo YOLO V12

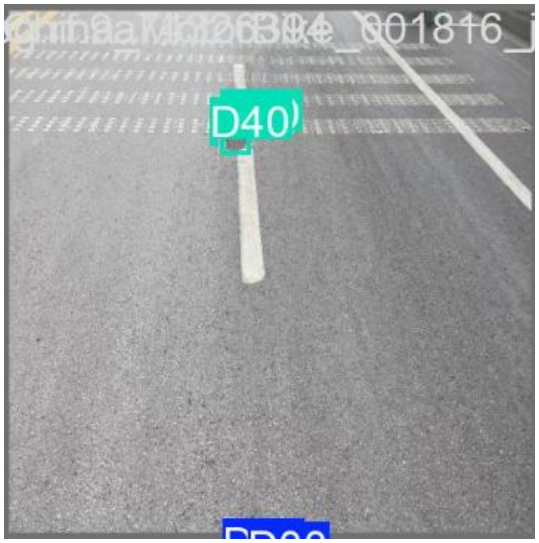
Fuente: Elaboración propia

Figura 71
Imágenes utilizadas para el entrenamiento del modelo YOLO V12



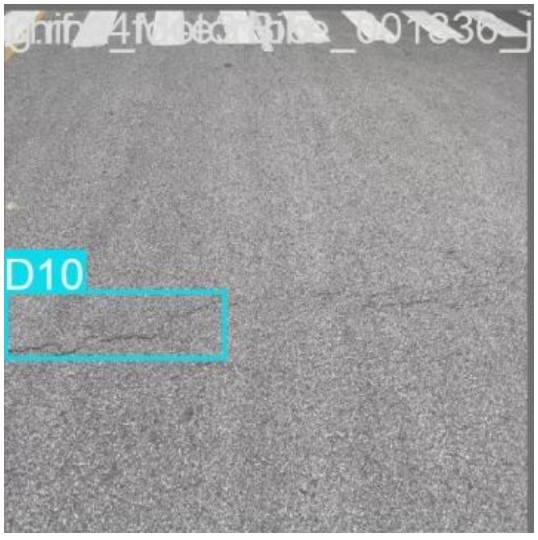
Fuente: Elaboración propia

Figura 72
Imágenes utilizadas para el entrenamiento del modelo YOLO V12



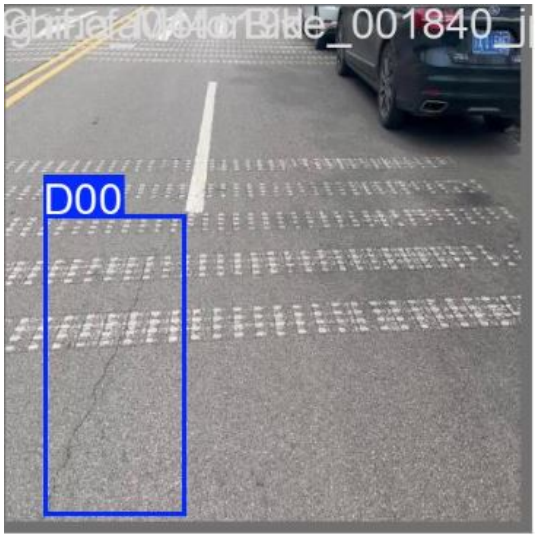
Fuente: Elaboración propia

Figura 73
Imágenes utilizadas para el entrenamiento del modelo YOLO V12



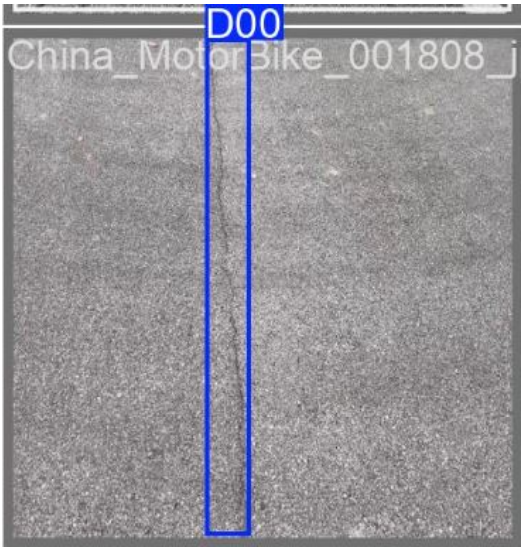
Fuente: Elaboración propia

Figura 74
Imágenes utilizadas para el entrenamiento del modelo YOLO V12



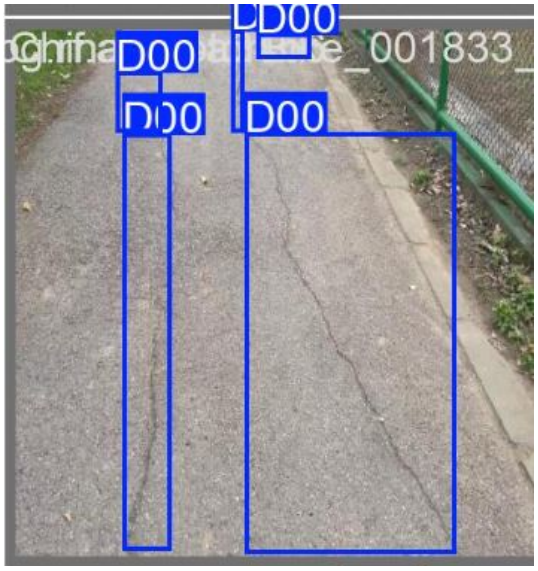
Fuente: Elaboración propia

Figura 75
Imágenes utilizadas para el entrenamiento del modelo YOLO V12



Fuente: Elaboración propia

Figura 76
Imágenes utilizadas para el entrenamiento del modelo YOLO V12



Fuente: Elaboración propia

Para entrenar el modelo YOLOv12 se empleó un conjunto de imágenes específicamente recolectadas para representar diversas condiciones del pavimento, incluyendo zonas con grietas longitudinales, transversales, en bloque y superficies sin deterioros. Las imágenes fueron capturadas con cámaras de alta resolución y, en algunos casos, mediante drones, con el fin de obtener perspectivas variadas y un mayor nivel de detalle. Cada fotografía fue seleccionada considerando iluminación adecuada, ausencia de sombras fuertes y nitidez suficiente para permitir una correcta anotación. Posteriormente, todas las imágenes fueron organizadas y etiquetadas manualmente siguiendo el formato requerido por YOLO, identificando cada instancia de grieta mediante cuadros delimitadores (bounding boxes). Este dataset, compuesto por ejemplos variados en ángulos, distancias y tamaños de grietas, permitió generar un conjunto de datos balanceado y representativo para mejorar la precisión del modelo durante el entrenamiento.

“Debo reconocer que un hombre que concluye que un argumento no tiene realidad,
porque se le ha escapado a su investigación, es culpable de imperdonable arrogancia.”

— David Hume

4 Resultados

4.1 Análisis e interpretación

4.1.1 Resultados para la detección de grietas en carreteras

Para la obtención del modelo de red neuroanal se utilizan las siguientes arquitecturas de datos:

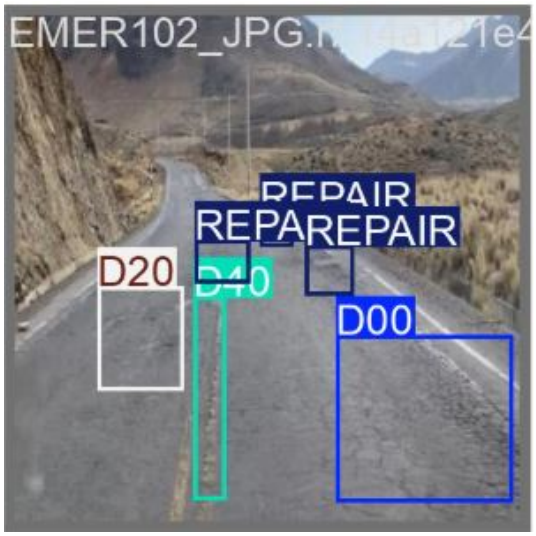
Tabla 9: Arquitectura de red YOLO V12

Variante	Capas Totales	Parámetros	FLOPs	Backbone	Capas Conv	Neuronas
YOLOv12n	168 capas	2.5M	6.7G	CSPDarknet-nano	125	850,000
YOLOv12s	218 capas	9.5M	21.5G	CSPDarknet-small	165	3.2 millones
YOLOv12m	308 capas	26.0M	65.8G	CSPDarknet-medium	235	8.5 millones
YOLOv12l	408 capas	52.0M	135.9G	CSPDarknet-large	315	17 millones
YOLOv12x	508 capas	87.0M	222.6G	CSPDarknet-xlarge	395	28 millones

***Fuente**:* Elaboración propia

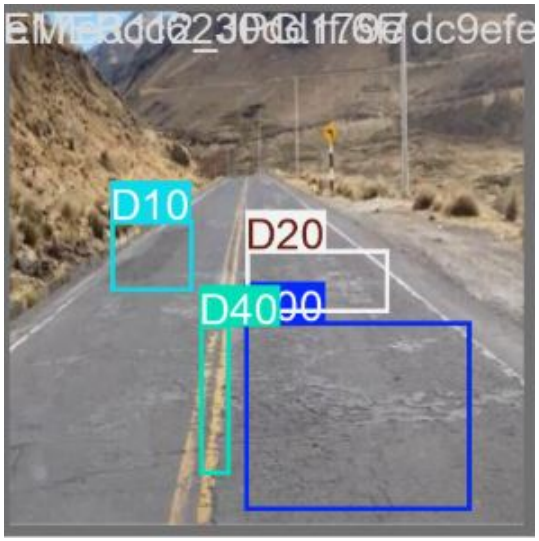
Nota: Para la presente investigación se utilizó la variante **YOLOv12s**.

Figura 77
Predicción del modelo YOLO V12, para el reconocimiento del tipo de falla



Fuente: Elaboración propia

Figura 78
Predicción del modelo YOLO V12, para el reconocimiento del tipo de falla

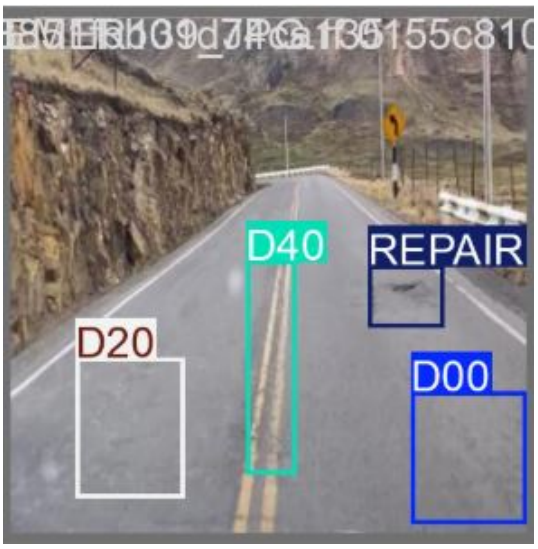


Fuente: Elaboración propia

Figura 79
Predicción del modelo YOLO V12, para el reconocimiento del tipo de falla

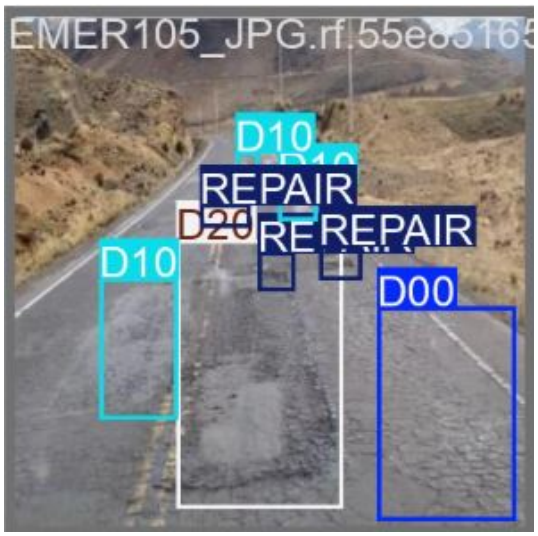
Fuente: Elaboración propia

Figura 80
Predicción del modelo YOLO V12, para el reconocimiento del tipo de falla



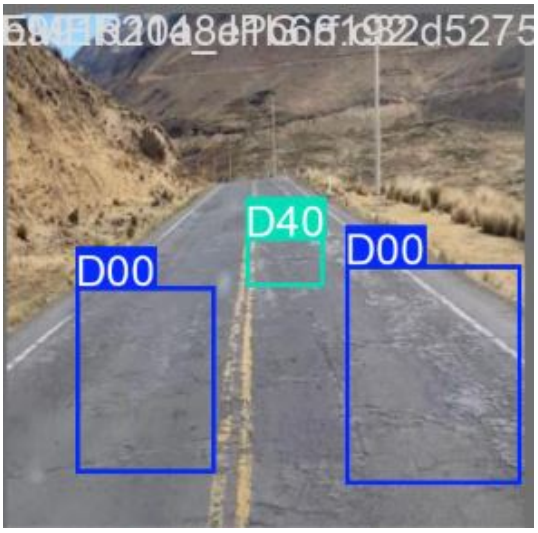
Fuente: Elaboración propia

Figura 81
Predicción del modelo YOLO V12, para el reconocimiento del tipo de falla



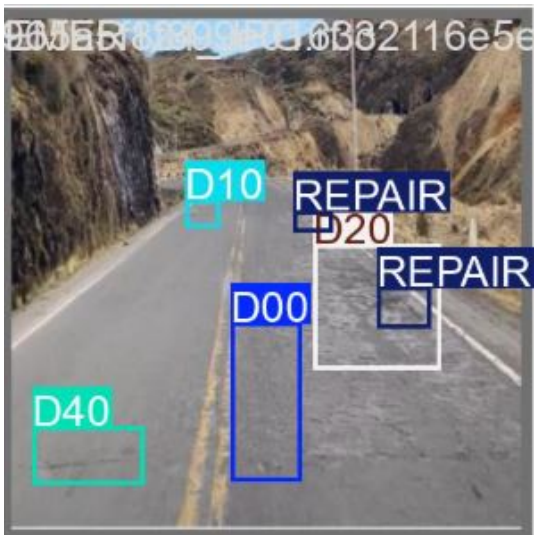
Fuente: Elaboración propia

Figura 82
Predicción del modelo YOLO V12, para el reconocimiento del tipo de falla



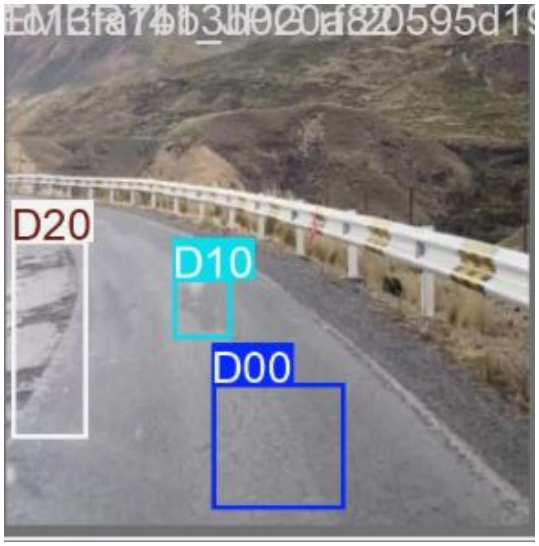
Fuente: Elaboración propia

Figura 83
Predicción del modelo YOLO V12, para el reconocimiento del tipo de falla



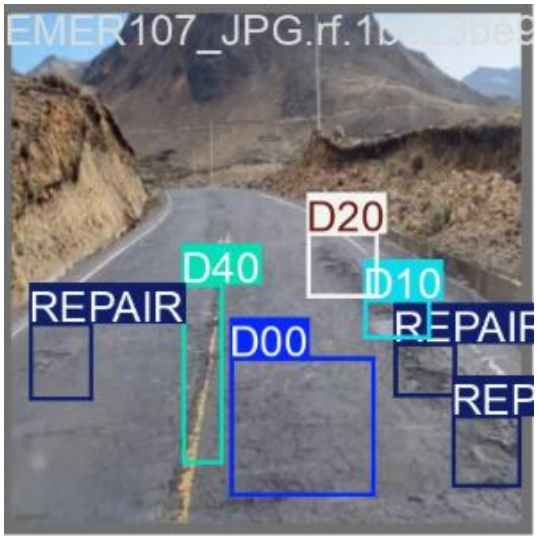
Fuente: Elaboración propia

Figura 84
Predicción del modelo YOLO V12, para el reconocimiento del tipo de falla



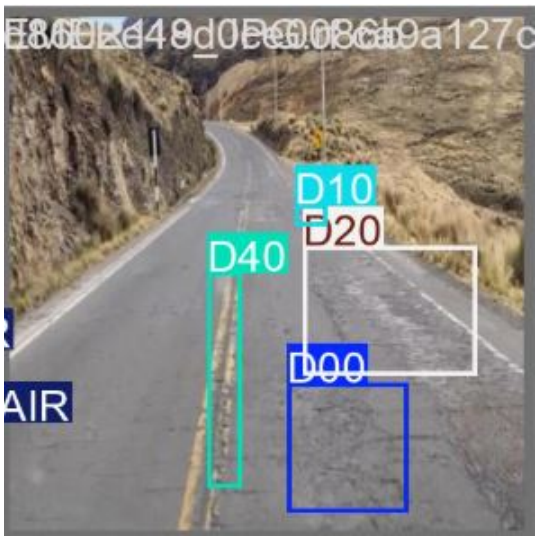
Fuente: Elaboración propia

Figura 85
Predicción del modelo YOLO V12, para el reconocimiento del tipo de falla



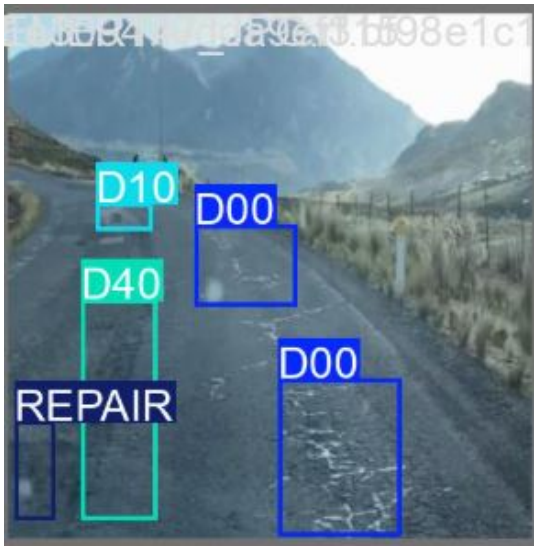
Fuente: Elaboración propia

Figura 86
Predicción del modelo YOLO V12, para el reconocimiento del tipo de falla



Fuente: Elaboración propia

Figura 87
Predicción del modelo YOLO V12, para el reconocimiento del tipo de falla

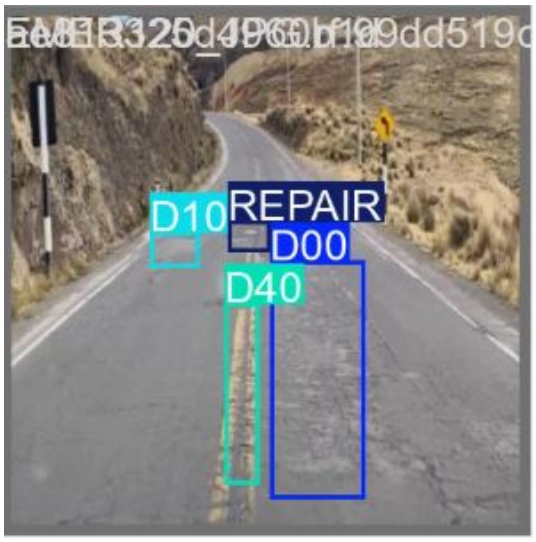


Fuente: Elaboración propia

Figura 88
Predicción del modelo YOLO V12, para el reconocimiento del tipo de falla

Fuente: Elaboración propia

Figura 89
Predicción del modelo YOLO V12, para el reconocimiento del tipo de falla



Fuente: Elaboración propia

Figura 90
Predicción del modelo YOLO V12, para el reconocimiento del tipo de falla

Fuente: Elaboración propia

Tabla 10: Resultados de los datos de entrenamiento

Clase	Imágenes	Instancias	Box P	Box R	mAP@50	mAP@[50:95]
Todos	**1871**	**515**	**0.668**	**0.668**	**0.704**	**0.419**
D00 (Piel de cocodrilo)	146	281	0.618	0.648	0.644	0.345
D10 (Parches)	72	107	0.632	0.495	0.596	0.246
D20 (Fisura en bloque)	53	74	0.675	0.645	0.736	0.436
D40 (Fisura de borde)	16	24	0.636	0.656	0.655	0.330
REPAIR (Reparación)	22	29	0.780	0.897	**0.887**	**0.736**

***Fuente**:* Elaboración propia

Nota: Box P = Precisión (*Precision*), Box R = Sensibilidad (*Recall*), mAP@50 = *Mean Average Precision* con IoU=0.50, mAP@[50:95] = *Mean Average Precision* promediada sobre IoU de 0.50 a 0.95 con paso 0.05.

4.1.2 Interpretación de los resultados

Análisis de las métricas reportadas para mAP@50 = 0.704 (70.4 %), tenemos la interpretación como un buen desempeño para la aplicación práctica, donde el significado del modelo detecta correctamente 70.4 % de las grietas cuando se usa un umbral de superposición del 50 %, donde el contexto de la detección de grietas se menciona en la siguiente tabla:

Tabla 11: Detección de grietas

Valor mAP@50	Interpretación	Aplicabilidad
$>$ 0.80	Excelente - Detección casi perfecta	Autónomo
0.70 - 0.79	**Bueno - Aceptable para producción**	**Semi-automatizado**
0.60 - 0.69	Moderado - Necesita mejoras	Con supervisión humana
$<$ 0.60	Deficiente - No recomendado	Solo investigación

***Fuente**:* Elaboración propia

Lo que implica que las siguientes resultados:

- De cada 10 grietas reales, el modelo detecta 7 correctamente
- Falsos negativos: Aproximadamente 3 de cada 10 grietas NO son detectadas
- Falsos positivos: Probablemente tiene algunas detecciones erróneas
- Aplicabilidad: Suficiente para sistemas semi-automatizados con supervisión humana

"Debo reconocer que un hombre que concluye que un argumento no tiene realidad, porque se le ha escapado a su investigación, es culpable de imperdonable arrogancia."

— David Hume

5 Conclusiones

5.1 Conclusiones

Se logró los objetivos de la siguiente manera:

1. **Desarrollo del modelo YOLOv12.** Se logró implementar exitosamente un modelo YOLOv12 con arquitectura `YOLOv12s`, cuyas características principales son: 218 capas totales, 9.5 millones de parámetros, 21.5 GFLOPs, backbone CSPDarknet-small y 165 capas convolucionales. El modelo alcanzó una precisión global mAP@50 del **70.4 %**, considerada como *buena y aceptable* para aplicaciones de producción semi-automatizadas en detección de grietas en pavimentos flexibles. Este resultado demuestra que la arquitectura seleccionada ofrece un balance óptimo entre precisión y requisitos computacionales.

2. **Precisión por tipo de falla.** El análisis de los parámetros del modelo YOLOv12 para el tramo Socos-Licapa reveló los siguientes resultados por clase de daño:
   - **REPAIR (reparación extrema):** 88.7 % (excelente)
   - **D20 (fisura en bloque):** 73.6 % (bueno)
   - **D40 (fisura de borde):** 65.5 % (moderado)
   - **D00 (piel de cocodrilo):** 64.4 % (moderado)
   - **D10 (parches):** 59.6 % (deficiente)

   La clase REPAIR presentó la mayor precisión (88.7 %), lo que indica que las reparaciones extremas son más fácilmente identificables por el modelo debido a su alta distintividad visual y contraste con el pavimento circundante. Por el contrario, la clase D10 (parches) mostró el rendimiento más bajo (59.6 %), sugiriendo la necesidad de aumentar el número de ejemplos de entrenamiento para esta categoría.

3. **Construcción y etiquetado del dataset.** Se recolectaron, procesaron y etiquetaron exitosamente **1871 imágenes** del tramo Socos-Licapa, distribuidas de la siguiente manera: 1582 imágenes (84.6 %) para entrenamiento, 197 imágenes (10.5 %) para validación y 92 imágenes (4.9 %) para prueba. El etiquetado se realizó mediante la plataforma Roboflow, identificando cinco clases de fallas representativas del tramo de estudio: D00 (piel de cocodrilo), D10 (parches), D20 (fisura en bloque), D40 (fisura de borde) y REPAIR (reparación extrema). Este dataset constituye un aporte original para futuras investigaciones en detección automatizada de daños en pavimentos de la región Ayacucho.

4. **Implementación computacional del modelo.** Se implementó el modelo YOLOv12 utilizando el lenguaje de programación **Python** en el entorno **Google Colab** (plataforma de código abierto). El código fuente desarrollado incluye las librerías necesarias (`ultralytics`, `opencv-python`, `numpy`, `matplotlib`), los comandos para el entrenamiento, validación y prueba, así como las funciones para el cálculo de métricas de rendimiento. Todo el código se encuentra disponible en el Anexo A.2, garantizando la reproducibilidad del estudio y permitiendo su adaptación a otros tramos viales.

5. **Limitaciones identificadas del estudio.** A partir del desarrollo de la investigación, se identificaron las siguientes limitaciones que deben ser consideradas para futuros trabajos:
   - *e)* El dataset se limita a un solo tramo vial (Socos-Licapa) y a condiciones específicas de captura (luz diurna, clima seco, junio de 2022).
   - *e)* La precisión para la clase D10 (parches) resultó deficiente (59.6 %), indicando la necesidad de más ejemplos de entrenamiento para esta categoría.
   - *e)* El modelo requiere recursos computacionales significativos (GPU con al menos 8 GB de VRAM) para un entrenamiento eficiente.
   - *e)* No se realizó una comparación sistemática con versiones anteriores de YOLO (v8, v9, v10, v11) para justificar cuantitativamente la selección de YOLOv12.

"Debo reconocer que un hombre que concluye que un argumento no tiene realidad, porque se le ha escapado a su investigación, es culpable de imperdonable arrogancia."

— David Hume

6 Recomendaciones

Con base en los hallazgos y limitaciones identificadas, se formulan las siguientes recomendaciones para futuras investigaciones y aplicaciones prácticas:

1. **Ampliación del dataset.** Se recomienda aumentar significativamente el número de imágenes del dataset, especialmente para las clases con menor precisión:
   - Clase D10 (parches): incorporar al menos 200-300 imágenes adicionales.
   - Clase D00 (piel de cocodrilo): incorporar al menos 150-200 imágenes adicionales.
   - Clase D40 (fisura de borde): incorporar al menos 100 imágenes adicionales.

   Un dataset más balanceado y numeroso mejorará la capacidad de generalización del modelo y aumentará la precisión en todas las clases.

2. **Aplicación de técnicas de aumento de datos.** Se recomienda implementar técnicas de **aumento de datos** (*data augmentation*) para enriquecer el dataset sin necesidad de nuevas capturas en campo. Las técnicas sugeridas incluyen:
   - Rotaciones aleatorias ($\pm 15°$, $\pm 30°$)
   - Cambios de brillo y contraste
   - Zoom aleatorio (0.8x a 1.2x)
   - Volteo horizontal (espejo)
   - Adición de ruido gaussiano

   Estas técnicas ayudarán a mejorar la robustez del modelo ante variaciones en las condiciones de captura.

3. **Validación en otros tramos viales.** Se sugiere validar el modelo entrenado en otros tramos de la Vía Los Libertadores-Wari (diferentes a Socos-Licapa) y, posteriormente, en carreteras de otras regiones del Perú. Esta validación cruzada permitirá evaluar la capacidad de **generalización geográfica** del modelo y detectar posibles sesgos introducidos por condiciones locales específicas.

4. **Comparación con otras arquitecturas YOLO.** Para futuras investigaciones, se recomienda realizar un estudio comparativo sistemático que evalúe el rendimiento de diferentes versiones de YOLO (v8, v9, v10, v11, v12 y versiones futuras) sobre el mismo dataset. Este análisis permitirá:

- Justificar cuantitativamente la selección de la arquitectura más adecuada.
- Identificar las ventajas y desventajas de cada versión en el contexto específico de detección de grietas.
- Establecer una línea base para futuras mejoras.

5. **Requisitos computacionales.** Para el entrenamiento eficiente del modelo YOLOv12, se recomienda utilizar hardware con las siguientes especificaciones mínimas:
   - GPU: NVIDIA RTX 3060 (12 GB VRAM) o superior (recomendado RTX 4070 o superior)
   - RAM: 32 GB como mínimo
   - Almacenamiento: SSD de 500 GB para el dataset y los pesos del modelo

   El uso de CPU para el entrenamiento no es recomendable, ya que los tiempos de procesamiento pueden extenderse de horas a días. Para investigadores con recursos limitados, se sugiere utilizar entornos cloud como Google Colab Pro o AWS SageMaker.

6. **Implementación práctica semi-automatizada.** Para aplicaciones prácticas en el sector público o privado, se sugiere implementar el modelo en un **flujo semi-automatizado** donde un inspector humano valide las detecciones con niveles de confianza inferiores al 70 %. Este enfoque combina:
   - **Ventaja:** Reducción significativa del tiempo de inspección (estimada en un 80 %).
   - **Seguridad:** Supervisión humana para detecciones dudosas o críticas.
   - **Costo-efectividad:** Optimización del recurso humano especializado.

7. **Exploración de nuevas arquitecturas y técnicas.** Se recomienda mantenerse actualizado con los avances en visión por computadora y explorar:
   - Nuevas versiones de YOLO a medida que estén disponibles (YOLOv13, v14, etc.).
   - Arquitecturas basadas en *Transformers* para visión (ViT, DETR).
   - Técnicas de *aprendizaje por transferencia* (*transfer learning*) utilizando modelos pre-entrenados en grandes datasets de pavimentos.
   - *Aprendizaje activo* (*active learning*) para seleccionar las imágenes más informativas para el etiquetado manual.

8. **Incorporación de información complementaria.** Para mejorar la utilidad práctica del sistema, se recomienda incorporar información complementaria en futuras versiones:
   - **Severidad del daño:** Clasificar las grietas por niveles de severidad (bajo, medio, alto).
   - **Geolocalización automática:** Integrar coordenadas GPS en cada detección para mapear las fallas.
   - **Cálculo del PCI:** Automatizar el cálculo del Índice de Condición del Pavimento (PCI) a partir de las detecciones.

- **Generación de informes:** Exportar automáticamente los resultados a formatos estándar (PDF, Excel, Shapefile para GIS).

9. **Documentación y código abierto.** Se recomienda publicar el dataset anotado y el código fuente en repositorios de acceso abierto (GitHub, Zenodo, Kaggle) bajo licencias permisivas, con el objetivo de:
   - Facilitar la reproducibilidad de los resultados.
   - Permitir que otros investigadores mejoren y adapten el modelo.
   - Contribuir al desarrollo de la comunidad científica en ingeniería vial e inteligencia artificial.

10. **Evaluación del tiempo de procesamiento.** En futuras investigaciones, se recomienda medir y reportar sistemáticamente los tiempos de procesamiento del modelo:
    - Tiempo de inferencia por imagen (milisegundos).
    - Tiempo de procesamiento por kilómetro de carretera (segundos/km).
    - Throughput del sistema (imágenes por segundo, FPS).

    Estas métricas son fundamentales para evaluar la viabilidad del modelo en aplicaciones en tiempo real, como sistemas montados en vehículos de inspección.

Referencias Bibliográficas

Ali, S. y Shahbaz, M. (2020). Streamfow forecasting by modeling the rainfall–streamfow relationship using artifcial neural networks. *©Springer Nature Switzerland AG 2020* (vid. pág. 55).

Basogain Olabe, X. (1998). *Redes neuronales artificiales y sus aplicaciones. Publicaciones de la Escuela de Ingenieros, Escuela Superior de Ingeniería de Bilbao, UPV, 79 pp.* (Vid. pág. 55).

Borja, M. (2012). *Metodología de la Investigación Científica pra Ingenieros*. (Vid. pág. 55).

Cartaya, S. y Mantuano, R. (2016). Identificación de zonas en riesgo de inundación mediante la simulación hidráulica en un segmento del Río Pescadillo, Manabí, Ecuador. *Revista de investigación*, *40*(89), 158-170. http://ve.scielo.org/scielo.php?script=sci_arttext&pid=S1010-29142016000300009 (vid. pág. 55)

Chen SM, T. I., Wang YM. (2013). Using artifcial neural network approach for modellingrainfall–runof due to typhoon. J Earth (vid. pág. 55).

Cretu, G. F., Stavrou, A., Locasto, M. E., Stolfo, S. J. y Keromytis, A. D. (2008). Casting out Demons: Sanitizing Training Data for Anomaly Sensors. *Security and Privacy, IEEE Symposium on*, *0*, 81-95. https://doi.org/http://doi.ieeecomputersociety.org/10.1109/SP.2008.11 (vid. pág. 55)

Franco y Michael. (2010). Redes Neuronales artificiales. *Springer Series in Optical Sciences* (vid. pág. 55).

Gestal, P. (2013). *Introducción a las redes de neuronas artificiales, Universidad de La Coruña, España.* http://sabia.tic.udc.es/mgestal/cv/RNATutorial/TutorialRNA.pdf. (Vid. pág. 55)

Gutiérrez Peña, P. A. (2009). *Nuevos modelos de redes neuronales evolutivas y regresión logística generalizada utilizando funciones de base. Aplicaciones* (Tesis doctoral). Universidad de Granada. decsai.ugr.es/Documentos/tesis_dpto/124.pdf. (Vid. pág. 55)

Herrera, J., Yari, Y., Luque, E. y Valdivia, Y. J. T. (2013). Red Neuronal aplicada a la generación de caudales mensuales estocásticos. *Proceedings del XII Congreso de la Sociedad Peruana de Computacion*. https://doi.org/10.13140/2.1.4047.7762 (vid. pág. 55)

Hiemstra, L. A. V. y Creese, R. C. (1970). SYNTHETIC GENERATION OF SEASONAL PRECIPITATION. *Journal of Hydrology 11 (1970) 30-46*. https://sci-hub.st/https://www.sciencedirect.com/science/article/pii/0022169470901137 (vid. pág. 55)

Jia, Y. y Culver, T. B. (2006). Bootstrapped artificial neural networks for synthetic flow generation with a small data sample. *Journal of Hydrology*. https://doi.org/10.1016/j.jhydrol.2006.06.005 (vid. pág. 55)

Kim, T., Shin, J. Y., Kim, H. y Heo, J. H. (2020). Ensemble-based Neural Network Modeling for Hydrologic Forecasts: Addressing Uncertainty in the Model Structure and Input Variable Selection. https://sci-hub.st/https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2019WR026262 (vid. pág. 55)

Montaño, J. (2002). *Redes neuronales artificiales aplicadas al análisis de datos. Tesis Doc., Palma de Mallorca, España, Universidad de las Islas Baleares.* (Vid. pág. 55).

Morales, I. (2010). *Comparación teórico práctica entre modelos estadísticos y el perceptrón multicapa. Tesis Lic., Pontificia Universidad Católica Valparaíso, 110 p.* (Vid. pág. 55).

Moreno, A. R. y Fontalvo, J. A. P. (2017). Análisis hidrológico e hidráulico de la cuenca del Río Fríon municipios de Ciénaga y zona Bananera, departamento del Magdalena. *Revista Logos, Ciencia and Tecnologia*, *9*(1), 156-178. https://www.redalyc.org/pdf/5177/517754057016.pdf (vid. pág. 55)

Olivos, V. y Contreras, J. (2019). *Diseño de una defensa ribereña mediante roca al volteo en la margen derecha del rio Pativilca tramo: Km 16 Al Km 20* (Tesis Pregrado). Universidad Nacional Pedro Ruiz Gallo. https://drive.google.com/file/d/1prScbZY_-ZOMz6r2B9D_8hqZLqW2xelw/view?usp=sharing. (Vid. pág. 55)

Rao y Rao. (1996). C++ Neural Network and fuzzy logic, BPG ,New Delhi India 380-381. *SCI HUB* (vid. pág. 55).

Sampieri, R. H., Collado, C. F. y Lucio, P. B. (2014). *Metodología de la Investigación* (M. G.-H. /. Interamericana, Ed.; 6.ª ed.). (Vid. pág. 55).

Socha, D. y Ortiz, G. (2005). *Aplicación de redes neuronales MLP a la predicción de un paso en series de tiempo. Fundación Universitaria Konrad Lorenz, Bogotá, pp. 183.* (Vid. pág. 55).

Socha, G. D. F. y Ortiz, H. G. A. (2005). *Aplicación de redes neuronales MLP a la predicción de un paso en series de tiempo. Fundación Universitaria Konrad Lorenz, Bogotá, pp. 183*. (Vid. pág. 55).

Ventura, W. H. (2024). Redes Neuronales en la Hidrologia. *Sci Direct*. https://link.springer.com/article/10.1007/s40808-020-00803-z (vid. pág. 55)

Zhang, G. y Patuwo, E. (1998). *Forecasting with artificial neural networks: The state of the art. International Journal of Forecasting 14, 35-62.* (Vid. pág. 55).

“Debo reconocer que un hombre que concluye que un argumento no tiene realidad, porque se le ha escapado a su investigación, es culpable de imperdonable arrogancia.”

— David Hume

A Anexos

A.1 Evidencia fotográfica en campo

Figura 91 Obtención de la dataset en campo de las imágenes de detección de daños de pavimento

Fuente: Elaboración propia

Figura 92

Obtención de la dataset en campo de las imágenes de detección de daños de pavimento

Fuente: Elaboración propia

Figura 93

Obtención de la dataset en campo de las imágenes de detección de daños de pavimento

Fuente: Elaboración propia

Figura 94

Obtención de la dataset en campo de las imágenes de detección de daños de pavimento

Fuente: Elaboración propia

Figura 95

Obtención de la dataset en campo de las imágenes de detección de daños de pavimento

Fuente: Elaboración propia

Figura 96

Obtención de la dataset en campo de las imágenes de detección de daños de pavimento

Fuente: Elaboración propia

Figura 97

Obtención de la dataset en campo de las imágenes de detección de daños de pavimento

Fuente: Elaboración propia

A.2 Código fuente en Python con GOOGLE COLAB

Figura 98 Código fuente en Python con GOOGLE COLAB

EMERGIO ALANYA CUBA - CODIGO PARA DETECCIÓN DE GRIETAS EN CARRETERAS

```
!nvidia-smi
```

```
Tue Dec  9 01:26:42 2025
+-----------------------------------------------------------------------------------------+
| NVIDIA-SMI 550.54.15              Driver Version: 550.54.15      CUDA Version: 12.4     |
|-----------------------------------------+------------------------+----------------------+
| GPU  Name                 Persistence-M | Bus-Id          Disp.A | Volatile Uncorr. ECC |
| Fan  Temp   Perf          Pwr:Usage/Cap |           Memory-Usage | GPU-Util  Compute M. |
|                                         |                        |               MIG M. |
|=========================================+========================+======================|
|   0  Tesla T4                       Off |   00000000:00:04.0 Off |                    0 |
| N/A   59C    P0             29W /   70W |     896MiB /  15360MiB |      0%      Default |
|                                         |                        |                  N/A |
+-----------------------------------------+------------------------+----------------------+

+-----------------------------------------------------------------------------------------+
| Processes:                                                                              |
|  GPU   GI   CI        PID   Type   Process name                              GPU Memory |
|        ID   ID                                                               Usage      |
|=========================================================================================|
+-----------------------------------------------------------------------------------------+
```

Fuente: Elaboración propia

Figura 99 Código fuente en Python con GOOGLE COLAB

Step 01 # Install the Ultralytics Package

```
!pip install ultralytics
```

Mostrar el resultado oculto

Step 02 # Import All the Requried Libraries

```
import os
import ultralytics
ultralytics.checks()
```

```
Ultralytics 8.3.235 🚀 Python-3.12.12 torch-2.9.0+cu126 CUDA:0 (Tesla T4, 15095MiB)
Setup complete ✅ (2 CPUs, 12.7 GB RAM, 38.4/112.6 GB disk)
```

```
from ultralytics import YOLO
from IPython.display import Image
```

```
HOME = os.getcwd()
print(HOME)
```

```
/content
```

Fuente: Elaboración propia

Figura 100 Código fuente en Python con GOOGLE COLAB

Step # 03 Download Dataset from Roboflow

https://universe.roboflow.com/project-uyrxf/ppe_detection-v1x3l/dataset/2

```
!pip install roboflow

from roboflow import Roboflow
rf = Roboflow(api_key="cy3vgmYxOusQICN3ExGg")
project = rf.workspace("zhichenjun").project("crack1-vodtg")
version = project.version(1)
dataset = version.download("yolov12")
```

```
Requirement already satisfied: roboflow in /usr/local/lib/python3.12/dist-packages (1.2.11)
Requirement already satisfied: certifi in /usr/local/lib/python3.12/dist-packages (from roboflow) (2025.11.12)
Requirement already satisfied: idna==3.7 in /usr/local/lib/python3.12/dist-packages (from roboflow) (3.7)
Requirement already satisfied: cycler in /usr/local/lib/python3.12/dist-packages (from roboflow) (0.12.1)
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.12/dist-packages (from roboflow) (1.4.9)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.12/dist-packages (from roboflow) (3.10.0)
Requirement already satisfied: numpy>=1.18.5 in /usr/local/lib/python3.12/dist-packages (from roboflow) (2.0.2)
Requirement already satisfied: opencv-python-headless==4.10.0.84 in /usr/local/lib/python3.12/dist-packages (from roboflow) (4.10.0.84)
Requirement already satisfied: Pillow>=7.1.2 in /usr/local/lib/python3.12/dist-packages (from roboflow) (11.3.0)
Requirement already satisfied: pi-heif<2 in /usr/local/lib/python3.12/dist-packages (from roboflow) (1.1.1)
Requirement already satisfied: pillow-avif-plugin<2 in /usr/local/lib/python3.12/dist-packages (from roboflow) (1.5.2)
Requirement already satisfied: python-dateutil in /usr/local/lib/python3.12/dist-packages (from roboflow) (2.9.0.post0)
Requirement already satisfied: python-dotenv in /usr/local/lib/python3.12/dist-packages (from roboflow) (1.2.1)
Requirement already satisfied: requests in /usr/local/lib/python3.12/dist-packages (from roboflow) (2.32.4)
Requirement already satisfied: six in /usr/local/lib/python3.12/dist-packages (from roboflow) (1.17.0)
Requirement already satisfied: urllib3>=1.26.6 in /usr/local/lib/python3.12/dist-packages (from roboflow) (2.5.0)
Requirement already satisfied: tqdm>=4.41.0 in /usr/local/lib/python3.12/dist-packages (from roboflow) (4.67.1)
Requirement already satisfied: PyYAML>=5.3.1 in /usr/local/lib/python3.12/dist-packages (from roboflow) (6.0.3)
```

Fuente: Elaboración propia

Figura 101 Código fuente en Python con GOOGLE COLAB

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
!unzip roboflow.zip -d /content/roboflow_dataset
```

```
unzip:  cannot find or open roboflow.zip, roboflow.zip.zip or roboflow.zip.ZIP.
```

```
!ls {dataset.location}
```

```
data.yaml  README.dataset.txt  README.roboflow.txt  test  train  valid
```

```
!unrar x "data_yolov12.rar" "/content/dataset/"
```

```
UNRAR 6.11 beta 1 freeware      Copyright (c) 1993-2022 Alexander Roshal

Cannot open data_yolov12.rar
No such file or directory
No files to extract
```

```
!apt-get install unrar -y
```

```
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
unrar is already the newest version (1:6.1.5-1ubuntu0.1).
0 upgraded, 0 newly installed, 0 to remove and 41 not upgraded.
```

Fuente: Elaboración propia

Figura 102

Código fuente en Python con GOOGLE COLAB

Step # 04 Fine-tune YOLOv12 model on a Custom Dataset

NOTE: We need to make a few changes to our downloaded dataset so it will work with YOLOv12. Run the following bash commands to prepare your dataset for training by updating the relative paths in the `data.yaml` file, ensuring it correctly points to the subdirectories for your dataset's `train`, `test`, and `valid` subsets.

```
!sed -i '$d' {dataset.location}/data.yaml
!sed -i '$d' {dataset.location}/data.yaml
!sed -i '$d' {dataset.location}/data.yaml
!sed -i '$d' {dataset.location}/data.yaml
!echo -e "test: ../test/images\ntrain: ../train/images\nval: ../valid/images" >> {dataset.location}/data.yaml
```

```
!cat {dataset.location}/data.yaml
```

```
train: ../train/images
val: ../valid/images
test: ../test/images

nc: 5
names: ['D00', 'D10', 'D20', 'D40', 'Repair']

roboflow:
  workspace: zhichenjun
test: ../test/images
train: ../train/images
val: ../valid/images
```

Fuente: Elaboración propia

Figura 103

Código fuente en Python con GOOGLE COLAB

```
model = YOLO("yolo12s.yaml")
```

```
results = model.train(data=f'{dataset.location}/data.yaml', epochs=50)
```

```
Ultralytics 8.3.235 🚀 Python-3.12.12 torch-2.9.0+cu126 CUDA:0 (Tesla T4, 15095MiB)
engine/trainer: agnostic_nms=False, amp=True, augment=False, auto_augment=randaugment, batch=16, bgr=0.0, box=7.5,
Overriding model.yaml nc=80 with nc=5

                   from  n    params  module                                       arguments
  0                  -1  1       928  ultralytics.nn.modules.conv.Conv             [3, 32, 3, 2]
  1                  -1  1     18560  ultralytics.nn.modules.conv.Conv             [32, 64, 3, 2]
  2                  -1  1     26080  ultralytics.nn.modules.block.C3k2            [64, 128, 1, False, 0.25]
  3                  -1  1    147712  ultralytics.nn.modules.conv.Conv             [128, 128, 3, 2]
  4                  -1  1    103360  ultralytics.nn.modules.block.C3k2            [128, 256, 1, False, 0.25]
  5                  -1  1    590336  ultralytics.nn.modules.conv.Conv             [256, 256, 3, 2]
  6                  -1  2    689408  ultralytics.nn.modules.block.A2C2f           [256, 256, 2, True, 4]
  7                  -1  1   1180672  ultralytics.nn.modules.conv.Conv             [256, 512, 3, 2]
  8                  -1  2   2689536  ultralytics.nn.modules.block.A2C2f           [512, 512, 2, True, 1]
  9                  -1  1         0  torch.nn.modules.upsampling.Upsample         [None, 2, 'nearest']
 10             [-1, 6]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 11                 -1  1    345856  ultralytics.nn.modules.block.A2C2f           [768, 256, 1, False, -1]
 12                 -1  1         0  torch.nn.modules.upsampling.Upsample         [None, 2, 'nearest']
 13             [-1, 4]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 14                 -1  1     95104  ultralytics.nn.modules.block.A2C2f           [512, 128, 1, False, -1]
 15                 -1  1    147712  ultralytics.nn.modules.conv.Conv             [128, 128, 3, 2]
 16            [-1, 11]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 17                 -1  1    296704  ultralytics.nn.modules.block.A2C2f           [384, 256, 1, False, -1]
 18                 -1  1    590336  ultralytics.nn.modules.conv.Conv             [256, 256, 3, 2]
 19             [-1, 8]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 20                 -1  1   1511424  ultralytics.nn.modules.block.C3k2            [768, 512, 1, True]
 21       [14, 17, 20]  1    821343  ultralytics.nn.modules.head.Detect           [5, [128, 256, 512]]
YOLO12s summary: 272 layers, 9,255,071 parameters, 9,255,055 gradients, 21.5 GFLOPs
```

Fuente: Elaboración propia

Figura 104 Código fuente en Python con GOOGLE COLAB

```
Freezing layer 'model.21.dfl.conv.weight'
AMP: running Automatic Mixed Precision (AMP) checks...
AMP: checks passed ✅
train: Fast image access ✅ (ping: 0.0±0.0 ms, read: 1558.3±535.2 MB/s, size: 94.9 KB)
train: Scanning /content/crack1-1/train/labels... 1582 images, 0 backgrounds, 0 corrupt: 100% ━━━━━━━━━━━━ 1582/1582 1.2Kit/s 1.3s
train: New cache created: /content/crack1-1/train/labels.cache
albumentations: Blur(p=0.01, blur_limit=(3, 7)), MedianBlur(p=0.01, blur_limit=(3, 7)), ToGray(p=0.01, method='weighted_average', num_output_channels=3),
val: Fast image access ✅ (ping: 0.0±0.0 ms, read: 544.5±348.0 MB/s, size: 92.9 KB)
val: Scanning /content/crack1-1/valid/labels... 197 images, 0 backgrounds, 0 corrupt: 100% ━━━━━━━━━━━━ 197/197 617.4it/s 0.3s
val: New cache created: /content/crack1-1/valid/labels.cache
Plotting labels to /content/runs/detect/train7/labels.jpg...
optimizer: 'optimizer=auto' found, ignoring 'lr0=0.01' and 'momentum=0.937' and determining best 'optimizer', 'lr0' and 'momentum' automatically...
optimizer: AdamW(lr=0.001111, momentum=0.9) with parameter groups 113 weight(decay=0.0), 120 weight(decay=0.0005), 119 bias(decay=0.0)
Image sizes 640 train, 640 val
Using 2 dataloader workers
Logging results to /content/runs/detect/train7
Starting training for 50 epochs...

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       1/50       6.2G       4.04       4.69      4.149         56        640: 100% ━━━━━━━━━━━━ 99/99 1.8it/s 54.9s
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 7/7 1.0it/s 6.7s
                   all        197        515   0.000618      0.148   0.000805   0.000211

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       2/50      6.97G      3.563       4.29      3.733         58        640: 100% ━━━━━━━━━━━━ 99/99 2.2it/s 44.9s
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 7/7 2.4it/s 2.9s
                   all        197        515      0.202      0.108    0.00319   0.000811

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       3/50      7.01G      3.161      3.826      3.286         81        640: 100% ━━━━━━━━━━━━ 99/99 2.3it/s 43.6s
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 7/7 3.0it/s 2.3s
                   all        197        515       0.24     0.0735     0.0172    0.00385

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       4/50      7.05G      2.996      3.559      3.071         57        640: 100% ━━━━━━━━━━━━ 99/99 2.3it/s 43.4s
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 7/7 3.0it/s 2.3s
                   all        197        515      0.241      0.125     0.0326    0.00866

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       5/50      7.09G      2.903      3.366      2.933         49        640: 100% ━━━━━━━━━━━━ 99/99 2.3it/s 43.6s
```

Fuente: Elaboración propia

Figura 105 Código fuente en Python con GOOGLE COLAB

Step # 05 Evaluate fine-tuned YOLOv12 model

```
import locale
locale.getpreferredencoding = lambda: "UTF-8"

!ls {HOME}/runs/detect/train3/
```

```
args.yaml                                        R_curve.png          val_batch0_labels.jpg
confusion_matrix_normalized.png                  results.csv          val_batch0_pred.jpg
confusion_matrix.png                             results.png          val_batch1_labels.jpg
events.out.tfevents.1740634900.2d1fba484f31.2551.2 train_batch0.jpg   val_batch1_pred.jpg
F1_curve.png                                     train_batch1.jpg     val_batch2_labels.jpg
labels_correlogram.jpg                           train_batch2.jpg     val_batch2_pred.jpg
labels.jpg                                       train_batch5680.jpg  weights
P_curve.png                                      train_batch5681.jpg
PR_curve.png                                     train_batch5682.jpg
```

Fuente: Elaboración propia

Figura 106 Código fuente en Python con GOOGLE COLAB

Step # 06 Download the Model Weights from the Google Drive

```
!gdown "https://drive.google.com/uc?id=1ka7Gj8RE6iP8-ExtigZJhmiyr5k5lqRQ&confirm=t"
```

```
Downloading...
From: https://drive.google.com/uc?id=1ka7Gj8RE6iP8-ExtigZJhmiyr5k5lqRQ&confirm=t
To: /content/best.pt
100% 18.9M/18.9M [00:00<00:00, 27.7MB/s]
```

Step # 07 Validate Fine-Tuned Model

```
model = YOLO("best.pt")  # load a custom model

# Validate the model
metrics = model.val()  # no arguments needed, dataset and settings remembered
metrics.box.map  # map50-95
metrics.box.map50  # map50
metrics.box.map75  # map75
metrics.box.maps  # a list contains map50-95 of each category
```

Fuente: Elaboración propia

Figura 107 Código fuente en Python con GOOGLE COLAB

Step # 08 Inference with Custom Model on Images

```
dataset.location
```

```
'/content/PPE_Detection-2'
```

```
results = model.predict(source = f"{dataset.location}/test/images", save = True)
```

```
image 132/322 /content/PPE_Detection-2/test/images/image_161_jpg.rf.e9f1edc598de732fbe06e17ab7f1093f.jpg: 640x640
image 133/322 /content/PPE_Detection-2/test/images/image_162_jpg.rf.8d23febe04d304ca04ca41e4e05e0d58.jpg: 640x640
image 134/322 /content/PPE_Detection-2/test/images/image_163_jpg.rf.f7e82ddfd2bb385b4e75b518b38da2bb.jpg: 640x640
image 135/322 /content/PPE_Detection-2/test/images/image_166_jpg.rf.7ccb62279e978b77695b5add605d2efb.jpg: 640x640
image 136/322 /content/PPE_Detection-2/test/images/image_16_jpg.rf.a06ad87e8d4dacae98077b9c59da7e3b.jpg: 640x640 1
image 137/322 /content/PPE_Detection-2/test/images/image_16_jpg.rf.d4d1dc0d5a04338e738fded4cf7ceda0.jpg: 640x640 1
image 138/322 /content/PPE_Detection-2/test/images/image_170_jpg.rf.204f03f511e2142a1d86f1699590bf41.jpg: 640x640
image 139/322 /content/PPE_Detection-2/test/images/image_170_jpg.rf.4cef213cb5c85645b40d421b483aa40d.jpg: 640x640
image 140/322 /content/PPE_Detection-2/test/images/image_171_jpg.rf.9e2639d816f7d563ac9dd735579997f5.jpg: 640x640
image 141/322 /content/PPE_Detection-2/test/images/image_172_jpg.rf.31406c1d19149a1d436e825b7f61a5e8.jpg: 640x640
image 142/322 /content/PPE_Detection-2/test/images/image_172_jpg.rf.424269707e1d13cddee2309bf7f0c338.jpg: 640x640
image 143/322 /content/PPE_Detection-2/test/images/image_172_jpg.rf.7921942e5666dd87229965a9ce527b2a.jpg: 640x640
image 144/322 /content/PPE_Detection-2/test/images/image_173_jpg.rf.01e37d693c49a2adf05f97e16a26cf6d.jpg: 640x640
image 145/322 /content/PPE_Detection-2/test/images/image_174_jpg.rf.ace92f513833c9347aab6ade2721a275.jpg: 640x640
image 146/322 /content/PPE_Detection-2/test/images/image_175_jpg.rf.1e607ab4b1eaded279344a7a8596ae0a.jpg: 640x640
image 147/322 /content/PPE_Detection-2/test/images/image_176_jpg.rf.bab7418f86a2ce94bbfb816aff110642.jpg: 640x640
image 148/322 /content/PPE_Detection-2/test/images/image_179_jpg.rf.0d3acd089502d4670d412596d830293d.jpg: 640x640
image 149/322 /content/PPE_Detection-2/test/images/image_179_jpg.rf.86f217f7efafc7284873efb1ac383009.jpg: 640x640
image 150/322 /content/PPE_Detection-2/test/images/image_17_jpg.rf.3e96ecd509ab52dcfe14a0b0283c1a32.jpg: 640x640 1
image 151/322 /content/PPE_Detection-2/test/images/image_17_jpg.rf.8c8ced2e1c8b73d04df0662bb5844d9f.jpg: 640x640 1
image 152/322 /content/PPE_Detection-2/test/images/image_181_jpg.rf.4df247d45e7d766c34abf549fbcc179a.jpg: 640x640
image 153/322 /content/PPE_Detection-2/test/images/image_181_jpg.rf.d2f67747766d024f1f4794ec58b67450.jpg: 640x640
image 154/322 /content/PPE_Detection-2/test/images/image_184_jpg.rf.6e3e662326936a9058c1dfe28eac386d.jpg: 640x640
image 155/322 /content/PPE_Detection-2/test/images/image_185_jpg.rf.1d76c6b574ec1a513a896e49e0819f66.jpg: 640x640
```

Fuente: Elaboración propia

A.3 Matriz de consistencia

Tabla 12: Operacionalización de variables

VARIABLE		Definición conceptual	Dimensiones	Indicadores	Unidad de medida
Tipo	**Nombre**				
Independiente (Causa)	Modelo YOLOv12 con Deep Learning	Algoritmo de deep learning de última generación para detección automatizada de objetos en una sola pasada (You Only Look Once, Versión 12)	Arquitectura de red	Número de capas	Und.
				Parámetros (M)	Millones
			Hiperparámetros	Tasa de aprendizaje (learning rate)	Valor numérico
				Número de épocas	Und.
Dependiente (Efecto)	Detección de daños en pavimentos flexibles	Identificación, localización y clasificación de fallas en pavimentos flexibles mediante procesamiento de imágenes		mAP@50	%
			Precisión de detección	mAP@[50:95]	%
				Precisión (Precision)	%
			Clasificación por tipo de daño	Recall (Sensibilidad)	%
				F1-Score	%

UNSCH FACULTAD DE INGENIERÍA DE MINAS, GEOLOGÍA Y CIVIL

ACTA DE SUSTENTACIÓN DE TESIS Nº 20-2026-FIMGC

PARA OPTAR EL TÍTULO PROFESIONAL DE INGENIERO DE CIVIL

En la Universidad Nacional de San Cristóbal de Huamanga, en la ciudad de Ayacucho, en cumplimiento a la **Resolución Decanal No 099-2026-FIMGC-D**, a los **ocho días del mes de mayo de 2026**, siendo las **4:22 p.m.**, reunidos en el **Auditorio de la Escuela Profesional de Ingeniería Civil**, bajo la presidencia del **MSc. José Ernesto ESTRADA CARDENAS**, y los miembros: ., **Ing. Saul Walter RETAMOZO FERNANDEZ** y **Ing. Edmundo CANCHARI GUTIERREZ**, actuando como secretario docente el **Ing. Alex Sander IRCAÑAUPA HUAMANÍ**, para proceder a la sustentación de tesis para optar el **Título Profesional de Ingeniero de Civil**, del Bachiller en Ciencias de la Ingeniería de Civil:

Emergio ALANYA CUBA

Quien presentó la tesis denominada:

"**Desarrollo de un modelo de aprendizaje automático para evaluar daños en pavimentos flexibles mediante el procesamiento de imágenes en el tramo Socos-Licapa, 2025**"

Los señores miembros del jurado luego de expuesta la tesis y absueltas las preguntas, deliberaron y declararon:

Aprobado con 16 (Diesiseis)

Siendo las **5:17 p.m.** del día **08 de mayo del 2026**, culmina el acto de sustentación de tesis, y en conformidad de lo actuado los miembros del jurado firmamos al pie del presente.

MSc. José Ernesto ESTRADA CARDENAS
Presidente

Ing. Edmundo CANCHARI GUTIERREZ
Miembro

Ing. Saul Walter RETAMOZO FERNANDEZ
Miembro - Asesor

Ing. Alex Sander IRCAÑAUPA HUAMANÍ
Secretario

FACULTAD DE INGENIERÍA
DE MINAS Y CIVIL
Av. Independencia S/N
Ciudad Universitaria
Central Tel. 066 312510
Anexo 151

CONSTANCIA DE ORIGINALIDAD DE TRABAJO DE INVESTIGACIÓN

CONSTANCIA N° 13-2026-FIMGC/ASIH

El que suscribe; responsable verificador de originalidad de trabajos de tesis de pregrado con el software Turnitin, de la Escuelas Profesional de **Ingeniería Civil** de la **Facultad de Ingeniería de Minas, Geología y Civil**; en cumplimiento a la **Resolución de Consejo Universitario N° 039-2021-UNSCH-CU**, Reglamento de Originalidad de Trabajos de Investigación de la Universidad Nacional San Cristóbal de Huamanga y **Resolución Decanal N° 697-2024-FIMGC-UNSCH-D**, dejo constancia de originalidad de trabajo de investigación, que el/la Sr./Srta.

Apellidos y Nombres	:	Emergio ALANYA CUBA
Escuela Profesional	:	INGENIERÍA CIVIL
Título de la Tesis	:	" Desarrollo de un modelo de aprendizaje automático para evaluar daños en pavimentos flexibles mediante el procesamiento de imágenes en el tramo Socos-Licapa, 2025"
Evaluación de la Originalidad	:	15 % Índice de Similitud
Identificador de la entrega	:	2967270509

Por tanto, según los Artículos 12, 13 y 17 del Reglamento de Originalidad de Trabajos de Investigación, es **PROCEDENTE** otorgar la **Constancia de Originalidad** para los fines que crea conveniente.

En señal de conformidad y verificación se firma la presente constancia

Ayacucho, 22 de mayo del 2026

UNIVERSIDAD NACIONAL DE SAN CRISTÓBAL DE HUAMANGA
Facultad de Ingeniería de Minas, Geología y Civil

Mg. Ing. Alex Sander IRCAÑAUPA HUAMANI
Verificador de Originalidad de Trabajos de Tesis de Pregrado
Escuela de Formación Profesional de Ingeniería Civil

Con depósito para Tramites
Cc. Archivo.

"Desarrollo de un modelo de aprendizaje automático para evaluar daños en pavimentos flexibles mediante el procesamiento de imágenes en el tramo Socos-Licapa, 2025"

por Emergio ALANYA CUBA

Fecha de entrega: 22-may-2026 01:10p. m. (UTC-0500)
Identificador de la entrega: 2967270509
Nombre del archivo: Te_Emergio_Alanya_Cuba.pdf (8.15M)
Total de palabras: 15142
Total de caracteres: 87800

"Desarrollo de un modelo de aprendizaje automático para evaluar daños en pavimentos flexibles mediante el procesamiento de imágenes en el tramo Socos-Licapa, 2025"

INFORME DE ORIGINALIDAD

15%	11%	5%	9%
INDICE DE SIMILITUD	FUENTES DE INTERNET	PUBLICACIONES	TRABAJOS DEL ESTUDIANTE

FUENTES PRIMARIAS

1	Submitted to Universidad Nacional de San Cristóbal de Huamanga Trabajo del estudiante	8%
2	hdl.handle.net Fuente de Internet	3%
3	repositorio.unfv.edu.pe Fuente de Internet	1%
4	repositorio.unsch.edu.pe Fuente de Internet	1%
5	repositorio.ucv.edu.pe Fuente de Internet	1%
6	es.slideshare.net Fuente de Internet	1%
7	repository.eia.edu.co Fuente de Internet	<1%
8	lume.ufrgs.br Fuente de Internet	<1%
9	repositorio.unh.edu.pe Fuente de Internet	<1%
10	Copari Romero, Fredy Gonzalo. "Procesamiento digital de imágenes para la calificación de exámenes con opción múltiple basado en equipos convencionales de bajo costo para la Universidad Nacional de Juliaca,	<1%

2018", Universidad Nacional del Altiplano de Puno (Peru)
Publicación

11	www.universitatcarlemany.com Fuente de Internet	<1 %

Excluir citas	Activo	Excluir coincidencias	< 30 words
Excluir bibliografía	Activo		