“DESARROLLO DE UNA HERRAMIENTA PARA IDENTIFICAR LA PRESENCIA DE BACHES EN VÍAS ASFÁLTICAS, UTILIZANDO TÉCNICAS DE PERCEPCIÓN REMOTA E INTELIGENCIA ARTIFICIAL”

MARÍA DEL PILAR CAICEDO GUARAMA

UNIVERSIDAD DEL VALLE
FACULTAD DE INGENIERÍA
PROGRAMA DE INGENIERÍA CIVIL Y GEOMÁTICA
SANTIAGO DE CALI
2025

“DESARROLLO DE UNA HERRAMIENTA PARA IDENTIFICAR LA PRESENCIA DE BACHES EN VÍAS ASFÁLTICAS, UTILIZANDO TÉCNICAS DE PERCEPCIÓN REMOTA E INTELIGENCIA ARTIFICIAL”

AUTOR:
MARÍA DEL PILAR CAICEDO GUARAMA

DIRECTOR:
M.Sc. FRANCISCO LUIS HERNÁNDEZ TORRES

UNIVERSIDAD DEL VALLE
FACULTAD DE INGENIERÍA
PROGRAMA DE INGENIERÍA CIVIL Y GEOMÁTICA
SANTIAGO DE CALI
2025

NOTA DE ACEPTACIÓN

_____

_____

_____

_____

_____

_____

_____

Firma del director

_____

Firma del jurado 1

_____

Firma del jurado 2

AGRADECIMIENTOS

A mis padres, con amor y gratitud infinita.

Este trabajo de grado es el reflejo de años de esfuerzo, sacrificio y perseverancia, pero sobre todo, del amor y apoyo incondicional que ustedes me han brindado en cada paso de mi vida. Desde mis primeros aprendizajes hasta este importante logro, siempre han estado a mi lado, guiándome con su sabiduría y motivándome a seguir adelante, incluso en los momentos más difíciles.

A ustedes, que con su ejemplo me enseñaron el valor del trabajo, la responsabilidad y la honestidad, que me mostraron que los sueños se alcanzan con dedicación y constancia. Cada consejo, cada gesto de aliento y cada muestra de confianza en mí han sido la fuerza que me ha impulsado a superar cada obstáculo.

TABLA DE CONTENIDO

RESUMEN 9

INTRODUCCIÓN 10

PROBLEMÁTICA 11

OBJETIVOS 13

OBJETIVO GENERAL 13

OBJETIVOS ESPECÍFICOS 13

JUSTIFICACIÓN 13

Marco Teórico 14

Marco conceptual 14

Marco de referencia 21

Marco teórico 29

Herramientas y Materiales 33

METODOLOGÍA 35

Diseño de la Investigación 35

Población y Muestra 36

Variables e Indicadores 38

Obtención del Modelo 39

Adquisición de Datos 40

Preprocesamiento de Datos 41

Ciclo de Entrenamiento 42

Construcción de la Aplicación 43

Análisis de Requerimientos 44

Diseño de Aplicación 45

Desarrollo de Aplicación 47

Validación del modelo y de la aplicación 48

RESULTADOS Y ANÁLISIS 48

Transformación del Conjunto de Datos 49

Modelo YoloV7-Tiny 49

Aplicación Web: GeoBach 60

DISCUSIÓN 66

CONCLUSIONES 68

FUTUROS TRABAJOS 69

Referencias.....71
Anexos 76

Listado de Tablas

Tabla 1: Estado del arte sobre autores que trataron el mismo caso.....26
Tabla 2: Herramientas y materiales utilizados para el desarrollo del modelo y de la aplicación web.....34
Tabla 3: Definición de rutas.....46
Tabla 4: Parámetros de entrenamiento para los modelos de YOLOV7-Tiny.....49
Tabla 5: Valores de las métricas obtenidas de los entrenamientos. 60
Tabla 6: Medidas obtenidas haciendo uso de GeoBach y medidas obtenidas en campo. 64
Tabla 7: Errores globales 64

Listado de Figuras

Figura 1: Arquitectura de una red neuronal 15
Figura 2: Modelo común de una red neuronal convolucional (CNN).....16
Figura 3: Proceso de una capa convolucional.....17
Figura 4: Capa de agrupación máxima 17
Figura 5: Capa de agrupación mínima.....18
Figura 6: Capa de agrupación promedio.....18
Figura 7: Transformación a una matriz unidimensional.....19
Figura 8: Comparación del modelo YoloV7 con respecto a otros modelos de Yolo 20
Figura 9. Imágenes de baches para primer entrenamiento del modelo. 37
Figura 10. Imágenes de baches capturadas para el segundo entrenamiento y validación de la vía Sibundoy – San Francisco en el departamento del Putumayo. 38
Figura 11. Flujo metodológico para la obtención del modelo de detección de baches en pavimento asfáltico. 40
Figura 12. Flujo metodológico para el desarrollo de la aplicación web de detección de baches en pavimento asfáltico. 44
Figura 13. Diseño de base de datos relacional. 45
Figura 14: Resultados de YOLO V7 Tiny para 200 épocas. 50
Figura 15: Valores del Recall en el primer entrenamiento entre las épocas 175 y 185. 51
Figura 16: Valores de la precisión en el primer entrenamiento entre las épocas 140 y 150.....51
Figura 17: Valores de la métrica mAP@0.5 en el primer entrenamiento entre las épocas 145 y 155.....52
Figura 18: Resultados de YOLO V7 Tiny para 100 épocas. 53
Figura 19: Valores del Recall en el segundo entrenamiento entre las épocas 70 y 80.....53
Figura 20: Valores de la precisión en el segundo entrenamiento entre las épocas 90 y 100.....54
Figura 21: Valores de la métrica mAP@0.5 en el segundo entrenamiento entre las épocas 110 y 119. 54
Figura 22: Gráficas obtenidas de los modelos entrenados, curva precisión-recall. 55
Figura 23: Gráficas obtenidas de los modelos entrenados, curva precisión-confianza.....56
Figura 24: Gráficas obtenidas de los modelos entrenados, curva recall-confianza. 57
Figura 25: Gráficas obtenidas de los modelos entrenados, matriz de confusión. 58
Figura 26: Gráficas obtenidas de los modelos entrenados, F1 Score.....59
Figura 27: Paginas de inicio de sesión y de registro. 60
Figura 28: Pagina para la creación de proyectos.....60
Figura 29: Creación de un proyecto en la aplicación GeoBach. 61

Figura 30: Proyecto creado en GeoBach. 61
Figura 31: Datos del proyecto. 61
Figura 32: Proceso de carga de imágenes. 62
Figura 33: Caja delimitadora con dimensiones reales del bache. 62
Figura 34: Resultados del proyecto que incluyen una tabla, un mapa y un gráfico de barras. 63
Figura 35: Resultados del proyecto en formato CSV. 63
Figura 36: Bache real con dimensiones de 1x1m y detección hecha por GeoBach con dimensiones de 0.94x1.19m. 64
Figura 37: Grafico de dispersión medidas de GeoBach con medidas reales (Ancho). 65
Figura 38: Grafico de dispersión medidas de GeoBach con medidas reales (Largo). 66
Figura 39: Bache real con dimensiones de 0.35 x 0.6m y detección hecha por GeoBach con dimensiones de 0.47x0.99m. 76
Figura 40: Bache real con dimensiones de 0.55x0.5m y detección hecha por GeoBach con dimensiones de 0.49x0.57m. 76
Figura 41: Bache real con dimensiones de 0.4x0.5 y detección hecha por GeoBach con dimensiones de 0.39x0.66m. 77
Figura 42: Bache real con dimensiones de 0.4x0.37 y detección hecha por GeoBach con dimensiones de 0.38x0.36m. 77
Figura 43: Bache real con dimensiones de 0.6x0.64m y detección hecha por GeoBach con dimensiones de 0.54x0.65m. 77
Figura 44: Bache real con dimensiones de 0.64x0.56m y detección hecha por GeoBach con dimensiones de 0.63x0.58m. 78
Figura 45: Bache real con dimensiones de 0.52x0.98m y detección hecha por GeoBach con dimensiones de 0.59x1.08m. 78
Figura 46: Bache real con dimensiones de 0.4x0.6m y detección hecha por GeoBach con dimensiones de 0.39x0.49m. 78
Figura 47: Bache real con dimensiones de 0.33x0.22m y detección hecha por GeoBach con dimensiones de 0.33x0.33m. 79
Figura 48: Bache real con dimensiones de 0.4x1.33m y detección hecha por GeoBach con dimensiones de 0.36x1.19m 79
Figura 49: Bache real con dimensiones de 0.42x0.6m y detección hecha por GeoBach con dimensiones de 0.49x0.62m 80

Listado de Anexos

Anexo 1: Comando utilizado para el primer entrenamiento con el modelo YoloV7 Tiny. 76

Anexo 2: Comando utilizado para el segundo entrenamiento con el modelo YoloV7 Tiny. 76

Anexo 3: Bache del área de estudio con dimensiones reales de 0.35 x 0.6. 76

Anexo 4: Bache con dimensiones reales de 0.55 x 0.5 76

Anexo 5: Bache con dimensiones de 0.4 x 0.5. 77

Anexo 6: Bache con dimensiones de 0.4 x 0.37. 77

Anexo 7: Bache con dimensiones de 0.6 x 0.64. 77

Anexo 8: Bache con dimensiones de 0.64 x 0.56. 78

Anexo 9: Bache con dimensiones de 0.52 x 0.98. 78

Anexo 10: Bache con dimensiones de 0.4 x 0.6 78

Anexo 11: Bache con dimensiones de 0.33 x 0.22.....79
Anexo 12: Bache con dimensiones de 0.4 x 1.33.....79
Anexo 13: Bache con dimensiones de 0.42 x 0.6.....79

RESUMEN

Los baches constituyen una de las patologías más comunes y estudiadas del pavimento a nivel mundial, afectando la seguridad, los costos de transporte y la calidad de vida en países como Estados Unidos, Reino Unido, India y Japón (Koch & Brilakis, 2011; HemaMalini et al., 2021; Chitale et al., 2020). La aparición de estas fallas, asociadas a procesos de infiltración de agua, fatiga del pavimento y tráfico pesado, ha impulsado el desarrollo de sistemas automatizados de inspección basados en visión por computador y aprendizaje profundo. En este contexto global, Colombia no es la excepción, presentando un deterioro vial significativo y afectaciones recurrentes para los usuarios (Gil, 2023; INVIAS, 2022).

Por lo anterior, teniendo en cuenta el crecimiento de las tecnologías y el aumento de investigaciones relacionadas con la inteligencia artificial, machine Learning y Deep Learning enfocadas a la detección de objetos; se propone el desarrollo de una aplicación web llamada GeoBach para el reconocimiento de baches en vías asfálticas por medio del uso de modelos de redes neuronales convolucionales (CNN). El objetivo de esta aplicación web es facilitar el proceso de inspección visual para la detección de baches en pavimentos asfálticos a través de la toma de fotografías con VANT; así mismo permite llevar un registro de estos con coordenadas donde fueron capturadas las imágenes, con el conteo de baches, medidas en metros y con la capacidad de descargar la información por medio de un archivo de tipo CSV.

Para el desarrollo de este trabajo, se hizo uso de React que permitió desarrollar el frontend de la aplicación de la mano con el principio DRY (Don't Repeat Yourself); el backend se construyó con Python y FastAPI, facilitando la gestión de solicitudes por parte del usuario. La aplicación resultante llamada GeoBach se encuentra conectada con un modelo de detección de YOLOV7 Tiny resultante de un proceso de doble entrenamiento, el primer modelo se entrenó con un dataset más robusto en comparación con el segundo que fue utilizado con el propósito de ajustar el modelo a características más específicas; finalmente como resultado se obtuvo un mAP @0.5 del 70.7%, una recuperación (recall) de 69.7% y una precisión del 70.17%.

Palabras Clave: Baches, Inteligencia Artificial, Percepción Remota, Redes Neuronales Convolucionales, YOLOV7 Tiny.

INTRODUCCIÓN

A nivel internacional, la detección temprana de baches ha sido reconocida como un desafío crítico para la gestión y conservación de la infraestructura vial debido a sus implicaciones en la seguridad, los costos de operación y la movilidad. Diversos países, entre ellos Estados Unidos, Reino Unido, Japón, India y Canadá, han estudiado este problema e incluso han implementado sistemas automatizados de inspección utilizando cámaras vehiculares, sensores LiDAR y técnicas avanzadas de visión por computador (Koch y Brilakis, 2011; Kang y Choi, 2017; Alshehhi et al., 2017).

La inspección visual de pavimentos es uno de los procedimientos más importantes en la evaluación integral de una vía. Actualmente en Colombia, existe un manual por parte del Instituto Nacional de Vías (INVIAS) con la universidad Nacional de Colombia del año 2006 que se titula "Manual para la inspección visual de pavimentos rígidos" en el que se describen las diferentes clases de daños que se pueden presentar en un pavimento de tipo rígido como lo son las grietas, baches, deterioro superficial, entre otros tipos de deterioro; además del procedimiento para el registro de daños en el que afirma que esta inspección tiene como objetivo determinar el porcentaje de área de pavimento afectada del tramo de vía, definiendo el tipo de daños con el fin de conocer las posibles causas del daño (Instituto Nacional de Vias; Universidad Nacional de Colombia, 2006).

Las inspecciones de la infraestructura vial se pueden realizar por metodologías como trabajo de campo aplicando el reconocimiento visual, lo que requiere de tiempo y está basado en opiniones subjetivas de los inspectores, lo que genera imprecisiones en los datos; por otra parte está el uso de tecnologías de percepción remota, que representan una alternativa viable en la captura de información y reducción de tiempo; Una posible solución a esta situación en el uso de tecnologías de percepción remota e inteligencia artificial como las redes neuronales debido a que representan un ahorro de tiempo y costos, obteniendo información con precisión y rapidez; así mismo, reduce el riesgo de accidentes para los inspectores. (HemaMalini, et al. 2021)

Por medio del presente trabajo se pretende realizar un aporte a la determinación del deterioro a nivel superficial de baches en vías asfálticas; para cumplir la meta propuesta, se hará uso de técnicas de percepción remota e inteligencia artificial, que permitan reducir tiempo, costos y exposición a factores de riesgo para los trabajadores.

PROBLEMÁTICA

El mal estado de las vías representa un factor de riesgo para todos aquellos usuarios que transiten por esta, generando situaciones que pueden desencadenar en accidentes de tránsito, afectaciones a los vehículos que transiten, también el mal flujo vehicular que puede desencadenar en embotellamientos o trancones.

La Cámara Colombiana de Infraestructura (CCI), indica que por cada desvío que los transportadores tienen que hacer a causa del mal estado de las carreteras, un viaje les puede valer hasta US$250 más. Esto significa que el sector de transporte puede llegar a perder anualmente cerca de $5,6 billones de pesos por dichos motivos, según cálculos hechos por Portafolio, basados en las cifras de la CCI y de la Federación Colombiana de Transportadores de Carga y su Logística (Colfecar) (Zonalogistica, 2018).

La presencia de baches en el pavimento forman parte de un indicador sobre el estado de la vía, generalmente se pueden presentar debido a la retención de agua en áreas con fisuras que con el flujo vehicular produce reducción de esfuerzos generando deformaciones; por lo que resulta de gran importancia realizar un monitoreo periódico sobre el estado de estos; actualmente en Colombia se realiza un análisis visual y un proceso de medición sobre el estado de los pavimentos, este proceso se realiza de manera convencional, en lo que corresponde a baches en el aspecto visual se identifica el porcentaje del tramo total que se encuentra afectado; y en cuanto al proceso de medición, se registran datos como la longitud, ancho, área afectada en m2 y profundidad del bache en mm, así como su ubicación precisa en el pavimento y según su severidad se categoriza el estado de estos como baja para profundidades menores o iguales a 25mm, media para profundidades entre 25mm y 50mm donde se encuentra expuesta la base y alta para profundidades mayores a 50mm donde se afecta la base granular; los datos de medición son recolectados generalmente por un trabajador con ayuda de un decámetro tomando las medidas de cada uno de los baches que puede observar, este proceso ocupa considerables tiempos y genera un factor de riesgo para el trabajador que realiza las mediciones sobre la vía (Macea, Morales y Márquez, 2016).

En otros países como Estados Unidos, Japón, Reino Unido y Canadá enfrentan estas situaciones haciendo uso de otros métodos como el uso de sistemas de detección automatizada de baches para rastrear el estado de las carreteras, otros han implementado tecnologías avanzadas como vehículos

equipados con cámaras y sensores, así como vehículos equipados con tecnología de escaneo laser. (KANG y CHOI, 2017)

Conservar en buen estado la infraestructura vial de nuestro país resulta importante al momento de evitar sobrecostos en reparaciones que no fueron tratadas a un debido tiempo; también resulta importante mantener una buena infraestructura vial para el normal flujo vehicular, seguridad y comodidad de los usuarios (Macea, Morales y Márquez, 2016). En Colombia, entidades como el Instituto Nacional de Vías (INVIAS) y el Ministerio de Transporte, han realizado esfuerzos en diseños de plan de acción que incluyen el inventario de los daños de pavimento con el fin de generar matrices de intervención que permiten mejorar el servicio del corredor vial (Instituto Nacional de Vías, 2013).

A nivel nacional, la situación de las vías en Colombia presenta serios desafíos que afectan tanto la seguridad vial como la economía del país. Según los últimos informes del Instituto Nacional de Vías (INVIAS), la red vial primaria pavimentada en Colombia tiene una extensión de 8.853,25 kilómetros. Sin embargo, se estima que el 46,49% de esta red se encuentra en condiciones deficientes, clasificándose como muy malo, malo o regular. Este deterioro de la infraestructura vial genera importantes problemas de movilidad y seguridad para los usuarios, lo que aumenta el riesgo de accidentes de tránsito y reduce la eficiencia del transporte. Además, más de la mitad de la red vial del país corresponde a vías terciarias, de las cuales solo un 10% está pavimentada. Esto dificulta la conexión entre las zonas rurales y las principales ciudades, afectando el desarrollo económico de diversas regiones. En consecuencia, es fundamental mejorar el estado de la infraestructura vial para garantizar la conectividad, reducir costos adicionales y promover el crecimiento regional (Instituto Nacional De Vias, 2022) (Fonter, 2023).

Pese a que en Colombia se han desarrollado estudios y existen documentos que proponen el uso de sistemas similares a los de otros países como el uso de vehículos equipados con cámaras y sensores o vehículos equipados con tecnología de escaneo laser, estos vehículos consumen combustible y requieren de una persona que realice el proceso de conducir el vehículo sobre la carretera; así mismo por parte de las entidades nacionales no existe una apropiación sobre el uso de nuevas tecnologías como los vehículos aéreos para desarrollar un análisis visual de baches, de cómo desarrollar un proceso de medición y monitoreo, como se puede evidenciar en los

documentos sobre inspección visual de pavimentos (Instituto Nacional de Vias; Universidad Nacional de Colombia, 2006).

Teniendo en cuenta la problemática existente en varias regiones del país sobre las áreas afectadas por baches en superficies asfálticas, surge la siguiente pregunta ¿Cómo identificar la presencia de baches haciendo uso de percepción remota e inteligencia artificial?

OBJETIVOS

OBJETIVO GENERAL

Determinar la presencia de baches en vías asfálticas, haciendo uso de técnicas de percepción remota e inteligencia artificial.

OBJETIVOS ESPECÍFICOS

1. Determinar la presencia de baches haciendo uso de fotografías aéreas y técnicas de inteligencia artificial.
2. Identificación de las características de baches (largo y ancho) en pavimento asfáltico haciendo uso de fotografías aéreas.
3. Comprobar resultados obtenidos con respecto a los resultados capturados in situ.

JUSTIFICACIÓN

Debido a la problemática presente en Colombia en lo que respecta a la presencia de baches, surge la necesidad de conocer las características de estos baches específicamente, su presencia en pavimentos flexibles teniendo en cuenta que este tipo de pavimento es el que requiere mayor mantenimiento y se deteriora más rápido en comparación a otros tipos como lo son las placa huellas y los pavimentos rígidos.

El uso de tecnologías de percepción remota para este tipo de casos, permite comprender de mejor manera el entorno en el que se va a trabajar, también se puede realizar cálculos a la distancia que existe entre estos sensores y la superficie del terreno con el que se va a trabajar (RMS, 2021), así como la generación de modelo tridimensional de la zona de estudio que aporta a una ubicación precisa de los daños en pavimento y a obtener valores objetivos del nivel de deterioro. Otros beneficios que se obtienen del uso de estas tecnologías a la hora de estar en la zona de trabajo es una menor exposición de los trabajadores a la zona de estudio, de la misma manera no se interrumpe el flujo vehicular que esté presente. Por medio de percepción remota se puede obtener una mayor base de conocimiento del lugar y sus alrededores, ya que se llega a espacios inaccesibles, aporta mucha más información gráfica que una inspección convencional donde el

trabajador con ayuda de un decámetro toma los datos sobre el bache presente en la vía; así mismo el uso de estas tecnologías representa un ahorro de tiempo y una precisión importante específicamente en la identificación de patologías de pavimento según lo expresan (Vasconcelos, et al., 2021). Por otra parte, la información es mucho más completa en cuanto a color de las fotografías, mayor resolución y realismo del terreno, también se presenta una reducción de costos hasta en un 70% (Cortés, 2021)

Debido a que los baches representan una de las patologías que más afecta y ocasiona repercusiones a los usuarios de las vías y así mismo teniendo en cuenta que en la ciudad de Cali no existe un registro actualizado del estado de las vías en cuanto a la presencia de baches resulta importante presentar una solución adecuada a esta problemática donde al mismo tiempo sea eficiente y eficaz, generando resultados confiables y también que representen un ahorro de tiempo y de recursos.

En resumen, el problema de la identificación de baches es importante resolverlo haciendo uso de técnicas de percepción remota e inteligencia artificial como lo son las redes neuronales debido a que se cubren grandes áreas de carretera de manera rápida y eficiente, así mismo, la aplicación de redes neuronales entrenadas es de utilidad para identificar y categorizar los baches con una velocidad y precisión difícil de lograr en comparación a un método manual de inspección. Entre los puntos clave en la importancia de resolver este problema están: mayor precisión, mayor rentabilidad en comparación de los métodos tradicionales y capacidad de monitoreo constante.

La información obtenida del uso de técnicas de percepción remota e inteligencia artificial aplicada a la identificación del nivel de deterioro superficial de baches en vías asfálticas puede ser de interés para instituciones que se dediquen al mantenimiento de las vías como a nivel municipal la Secretaría de Infraestructura y Valorización o a nivel nacional como el Instituto Nacional de Vías (INVIAS); contar con esta información puede aportar a la ejecución de planes de mantenimiento de la infraestructura vial dentro de municipios y a nivel general dentro de Colombia.

Marco Teórico

Marco conceptual

El pavimento asfáltico es uno de los tipos de superficie vial más comunes en las carreteras y calles de todo el mundo. Sin embargo, con el tiempo y el uso, este tipo de pavimento puede sufrir un deterioro que afecta su calidad y seguridad (Ministerio De Transporte Instituto Nacional De Vias, 2006). Los daños en la superficie vial pueden ser variados, desde fisuras y deformaciones hasta pérdida de capas estructurales y daños superficiales (Garzón y Hernández, 2018).

Entre los problemas más comunes se encuentran los baches, que son hoyos e imperfecciones en la superficie del pavimento que aparecen cuando se introduce agua en el pavimento a través de una grieta alojándose en las capas subyacentes. A medida que los vehículos transitan sobre estas superficies, el asfalto se derrumba formando un agujero que se amplía a medida que el tráfico golpea y rompe sus bordes (Volkswagen, 2021).

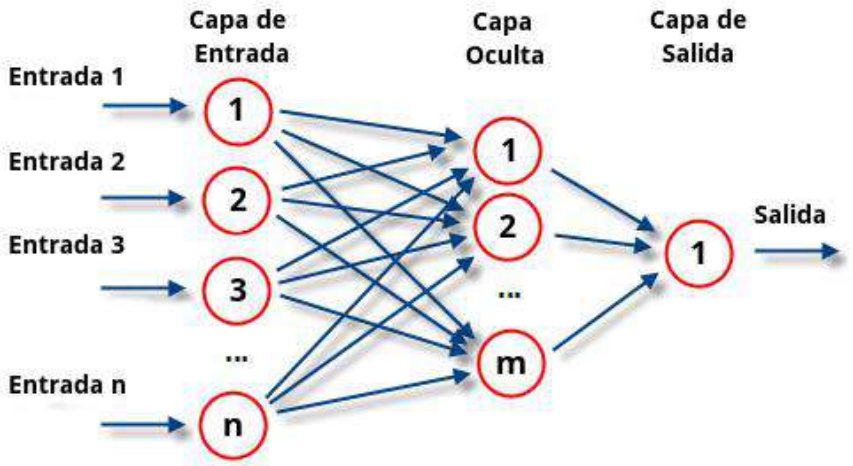


Figura 1: Arquitectura de una red neuronal

Fuente: Tomado de https://jacar.es/redes-neuronales-multicapa-avanzando-en-la-inteligencia-artificial/.

Para abordar el problema de los baches y otros daños en la superficie vial, existen técnicas de clasificación digital de imágenes como las redes neuronales que se inspiran en el modelo real de una neurona; en la figura 1 se puede observar la arquitectura de una red neuronal que se conforma de capas de entrada, capas de salida y en medio de estas dos capas también se encuentran capas ocultas que transforman la información de la capa de entrada por información que puede procesar la capa de salida (Steward, 2019). Cuentan con nodos que cumplen la función de procesamiento indispensable ya que se encarga de recibir entradas, aplicar pesos, realizar cálculos y producir una salida (Medium, 2023); también son las encargadas de aplicar la función de activación que analiza la respuesta de la señal para decidir si la neurona debe activarse teniendo en cuenta las entradas (Steward 2019). Los pesos determinan la fuerza de las conexiones entre neuronas y los sesgos que permiten que las neuronas se activen incluso cuando la suma ponderada de las entradas es cero (Medium, 2023).

Dentro de lo que son las redes neuronales convolucionales (CNN) es importante entender su arquitectura que contiene capas de entrada, capas convolucionales y de agrupación; estas redes son

un algoritmo de aprendizaje profundo y son análogas a las redes neuronales tradicionales ya que ambas se componen de neuronas que se auto optimizan por medio del aprendizaje (O´Shea y Nash, 2015). Las CNN son capaces de tomar una imagen como entrada, asignarle pesos y sesgos a varias características o aspectos de la imagen y diferenciarlos; en comparación de otros métodos más antiguos donde sus filtros se diseñan manualmente, las CNN tienen la capacidad de aprender de estos filtros (Saha, 2018).

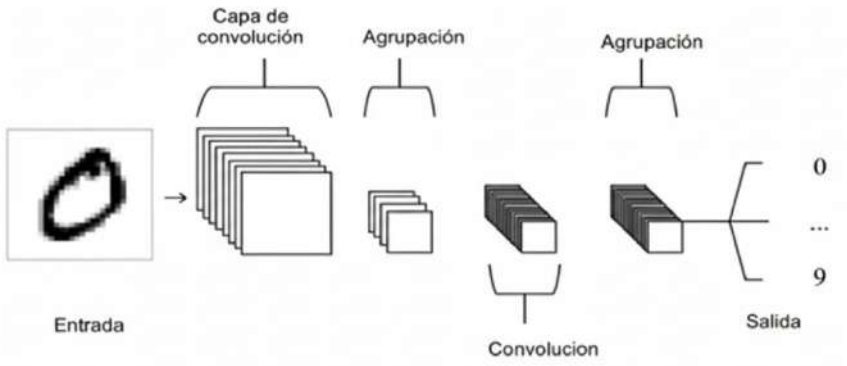


Figura 2: Modelo común de una red neuronal convolucional (CNN)

Fuente: Tomado de https://link.springer.com/chapter/10.1007/978-3-030-76493-7_8

En la figura 2 se observa el modelo común de una CNN donde en la capa de entrada se contienen los valores de pixeles de la imagen, la capa de convolución es la encargada de determinar la salida de las neuronas conectadas a regiones de la entrada mediante el cálculo del producto escalar entre sus pesos y la región conectada; la unidad lineal rectificada (ReLu) es la encargada de aplicar la función de activación a la salida de activación de la capa anterior (O´Shea y Nash, 2015); la capa de agrupación se encarga de hacer una reducción en la dimensionalidad espacial con el fin de disminuir el número de parámetros; finalmente se encuentran las capas completamente conectadas que realizan las mismas tareas que se encuentran dentro de una RNA que generan puntuaciones de clase a partir de las activaciones que se usan para la activación (O´Shea y Nash, 2015).

Las capas convolucionales cuentan con parámetros que se centran en el uso de Kernels capaces de aprender; estos Kernels tienen características de ser pequeños y de tener dimensión espacial que se extiende a lo largo de la profundidad de la entrada; cuando los datos llegan a la capa convolucional se convoluciona cada filtro a través de la dimensionalidad espacial de la entrada con el fin de generar un mapa de activación 2D (O´Shea y Nash, 2015).

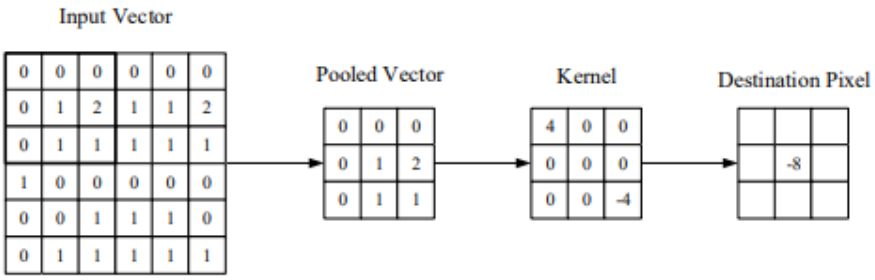


Figura 3: Proceso de una capa convolucional

Fuente: Tomado de https://trendspider.com/learning-center/convolutional-neural-networks-cnn/

En la figura 3 se observa el proceso que realiza una capa convolucional donde el kernel se coloca sobre el vector de entrada, que luego se calcula y se reemplaza con una suma ponderada de sí mismo y los pixeles cercanos (O´Shea y Nash, 2015).

Una vez obtenidos los mapas de activación, el siguiente paso en el proceso de extracción de características es la aplicación de las capas de agrupación, que se encargan de reducir el tamaño de los mapas de características, esto ayuda a que el cálculo se realice de manera más rápida ya que la cantidad de parámetros de entrenamiento se reducen.

La agrupación máxima realiza la reducción de características y toma el valor máximo de la región; este tipo de agrupación es utilizada cuando la imagen tiene un fondo oscuro porque la agrupación toma los píxeles más brillantes.

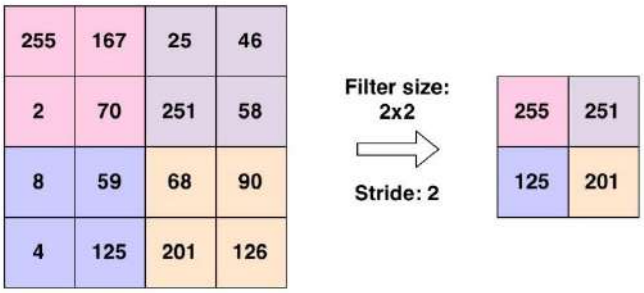


Figura 4: Capa de agrupación máxima

Fuente: Tomado de https://medium.com/@khrizoto/entendiendo-cnn-convolutional-neural-network-redes-neuronales-convolucionales-cb1ee89f9302

La agrupación mínima se encarga de reducir el tamaño de las características tomando el valor mínimo de una región; esta agrupación es usada cuando la imagen tiene un fondo Claro ya que se tomarán los valores de píxeles más oscuros.

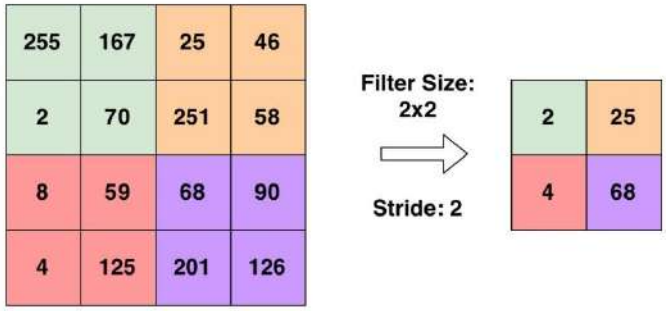


Figura 5: Capa de agrupación mínima

Fuente: Tomado de https://medium.com/@khrizoto/entendiendo-cnn-convolutional-neural-network-redes-neuronales-convolucionales-cb1ee89f9302

En otros casos, donde los bordes de la imagen no son tan relevantes, se utiliza la agrupación promedio que es el resumen de las características tomando el valor promedio de la región; es utilizada para suavizar los bordes de una imagen cuando estos no sean de importancia.

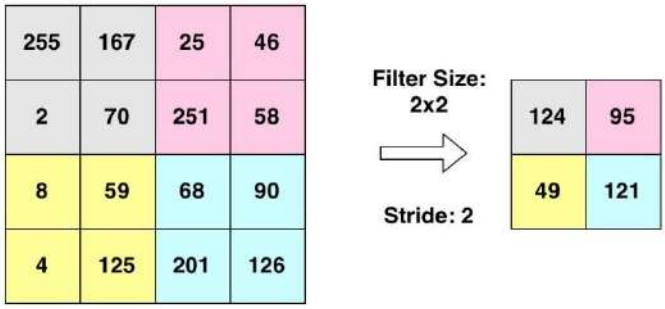


Figura 6: Capa de agrupación promedio.

Fuente: Tomado de https://medium.com/@khrizoto/entendiendo-cnn-convolutional-neural-network-redes-neuronales-convolucionales-cb1ee89f9302

Una vez procesadas estas características y obtenida la representación de la imagen, la capa de aplanamiento se encuentra entre la red neuronal convolucional y la red neuronal artificial. Su función es convertir la salida de la red convolucional en una entrada apta para el procesamiento en la red neuronal artificial, ya que las capas densas de la red neuronal requieren una matriz

unidimensional. Por lo tanto, se usa para aplanar los mapas de características obtenidos en los procesos anteriores, transformándolos en una matriz unidimensional.

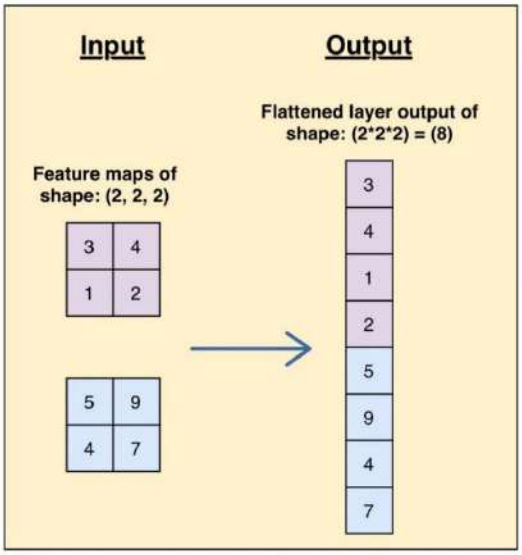


Figura 7: Transformación a una matriz unidimensional

Fuente: Tomado de https://zhuanlan.zhihu.com/p/706432870l

Dentro de lo que son las redes neuronales convolucionales, existe un tipo de red llamada YOLO que significa "You Only Look Once", este modelo fue creado por Joseph Redmon y Ali Farhadi y fue presentada en el año 2016 en el artículo "You Only Look Once: Unified, Real-Time Object Detection" donde es definida como un grupo de modelos de redes neuronales convolucionales que han sido diseñados para la detección de objetos en tiempo real, dentro de sus características principales está su capacidad para realizar detecciones a gran velocidad, también combina tareas de detección y clasificación en una sola red neuronal con el propósito de optimizar el proceso y reduciendo la complejidad computacional (Redmon, Divvala, et al. 2016). Desde que se presentó YOLO por primera vez hasta la actualidad, se han desarrollado diferentes versiones donde se ha mejorado la precisión, se han implementado técnicas avanzadas de entrenamiento con nuevas técnicas de arquitectura, entre otras.

Para este proyecto en particular se contempló la idea de utilizar YoloV7-tiny porque es una de las versiones más avanzadas y ligeras que existen de este modelo que supera en precisión y eficiencia a sus modelos anteriores; la versión completa es YoloV7 y establece un nuevo estado del arte; dentro de sus innovaciones en su arquitectura y técnicas de entrenamiento llamadas "Bag-of-

Freebies" que se encargan de mejorar el rendimiento reduciendo el sobreajuste sin afectar la velocidad del rendimiento y el costo computacional. La figura 8 obtenida (Wang, Bochkovskiy y Liao, 2023) es una comparación del rendimiento de YoloV7 con otros modelos como YOLOR, PPYOLOE, YOLOX, Scales-YOLOv4 y YOLOv5, donde se concluye que YoloV7 supera a los otros modelos en cuanto a precisión (AP) y rapidez (Wang, Bochkovskiy y Liao 2023).

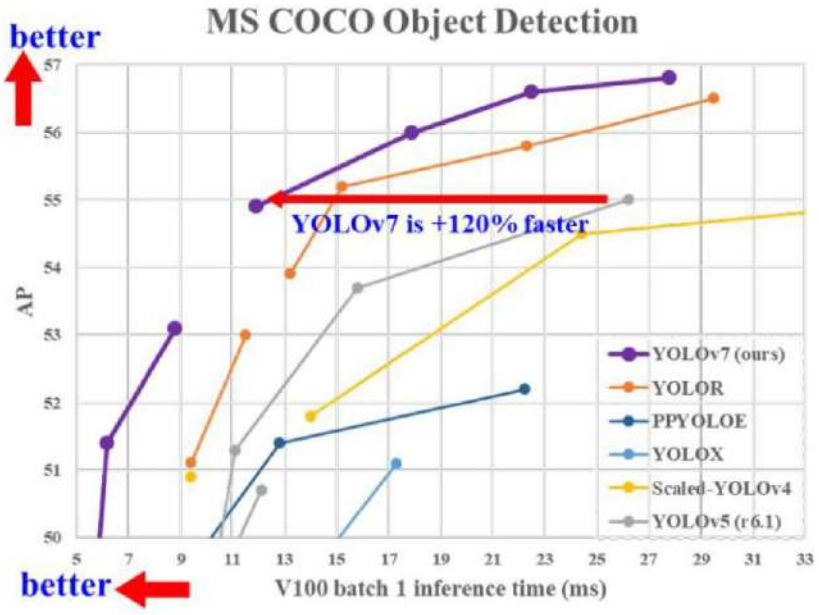


Figura 8: Comparación del modelo YoloV7 con respecto a otros modelos de Yolo

Fuente: Tomado de https://www.researchgate.net/figure/Comparison-between-YOLOv7-and-other-models-of-YOLO-Family-1_fig7_363848609

Dentro de las distintas técnicas que se pueden aplicar al modelo de Yolo v7 se destaca el entrenamiento multi escala que en los artículos (Redmon & Farhadi, 2017, 2018), se describe como aquella técnica que consiste en cambiar de manera aleatoria el tamaño de las imágenes almacenadas en el Dataset durante el entrenamiento; esto es beneficioso para el modelo en su etapa de entrenamiento ya que permite que este aprenda a detectar objetos en diferentes escalas y dimensiones de imagen, lo que mejora su capacidad de generalización y su robustez frente a las variaciones del tamaño de los baches (Redmon & Farhadi, YOLO9000: Better, 2017) (Redmon & Farhadi, 2018).

La técnica anteriormente mencionada presenta una solución que permite identificar daños en la superficie vial como son los baches (Ventura, y otros, 2016). Estas técnicas utilizan algoritmos de

inteligencia artificial para analizar las imágenes y extraer características relevantes como la textura, el color y la forma de las distintas zonas de la superficie vial (Garay, et al., 2025). Dichos algoritmos se pueden complementar con las tecnologías de percepción remota como los drones, que se definen como vehículos aéreos que reciben instrucciones de un piloto de manera remota y usualmente portan una cámara (Daley, 2023); con una utilidad ilimitada que puede aplicarse en distintos campos y que representan ventajas positivas en su implementación (Malini, et al., 2021).

Marco de referencia

En el contexto de la mejora y el mantenimiento de infraestructuras viales, el uso de tecnologías avanzadas ha demostrado ser fundamental para detectar y resolver problemas en carreteras y calles de manera eficiente. En este sentido, la detección de baches, intersecciones y otros problemas se ha beneficiado de enfoques automatizados que aprovechan el poder de las redes neuronales y la visión por computadora.

En cuanto a la extracción automatizada de carreteras e intersecciones se destacan dos enfoques: el primero presentado por Alshehhi, et al., (2017), se utilizan imágenes de alta resolución y una red neuronal convolucional. Este enfoque elimina la necesidad de preprocesamiento y utiliza segmentación basada en parches para la extracción de datos. Además, introduce un novedoso método de posprocesamiento basado en características de bajo nivel como asimetría y compacidad de regiones adyacentes con menor probabilidad de clasificarse como carreteras o edificios mediante la incorporación de estructura espacial como conectividad vial y cierre de edificios, mejorando la precisión de la clasificación, su principal diferencia está en el tamaño de los filtros del kernel, el número de unidades de filtro, el número de capas de convolución y la etapa de posprocesamiento; su método logra un mejor rendimiento usando capas completamente conectadas; el segundo se centra en la extracción automática de intersecciones viales. Dividen el proceso en dos módulos secuenciales: la extracción de la línea de la carretera y la extracción de las intersecciones. Esta metodología utiliza diferentes filtros de dirección morfológica para lograr la detección precisa de la orientación de la carretera y la interconectividad en las intersecciones (Hossain, Boshir y Islam, 2010).

Para la detección automatizada de baches y otros problemas de pavimento (Koch and Brilakis, 2011) presentan un enfoque innovador basado en imágenes de la superficie del pavimento asfáltico. Definen características visuales específicas de los baches, como la presencia de una o más sombras

que son más oscuras que el área circundante; la forma aproximadamente elíptica, debido a una vista en perspectiva y textura de la superficie dentro de un bache es mucho más gruesa y granulada que la textura de la superficie de un pavimento intacto. Su método se compone de segmentación de imágenes, extracción de formas y extracción y comparación de texturas, lo que permite una detección precisa (Koch y Brilakis, 2011). Por otro lado, Azhar et al., (2016) realizaron una propuesta para detectar y localizar baches en carreteras mediante la visión por computador; presentaron una metodología que se divide en tres partes. El primer paso es el cálculo de características de Histogramas de degradados orientados (HOG), clasificador Naïve Bayes y localización espacial de los baches; debido a que los baches no manifiestan formas fijas se determina el descriptor de características HOG, este evalúa la distribución de las direcciones de los bordes y luego calculan el histograma de las orientaciones de gradiente; el clasificador Naïve Bayes consiste en asignar una etiqueta a la imagen basado en la probabilidad máxima a posteriori; finalmente localizan los baches realizando una segmentación de la imagen basándose en cortes normalizados.

Alshehhi, et al., (2017) propusieron como objetivos desarrollar un sistema automatizado para la detección de baches y estimación de área usando Mask R-CNN que se define como un algoritmo para la detección de objetos con la capacidad de segmentar e identificar los límites de pixeles del objeto (Potrimba, 2023); su otro objetivo se basa en la creación de una aplicación Android que permite la captura de imágenes de baches que el sistema analiza automáticamente. Para lograr esto se usó un dataset con 1283 imágenes extraídas de Google Imágenes donde 1128 se usaron para entrenamiento y 140 imágenes para pruebas; en este documento no se habla de algún preprocesamiento aplicado a las imágenes, pero si se resalta el uso de Make Sense, una herramienta de código abierto para etiquetar imágenes en este caso según los requerimientos de Mask R-CNN. La metodología aplicada por Aware, et al., (2023) se divide en dos etapas, la primera se define como Modulo para la detección de baches, para esta etapa inicialmente se hace el uso de una cámara para capturar las imágenes que luego son procesadas por el modelo de aprendizaje para obtener cuadros delimitadores con el uso de Mask R-CNN (Aware, et al. 2023), se hace uso de una red de propuesta de región (RPN) que genera un conjunto de propuestas que luego son considerados como cuadros delimitadores de los objetos candidatos, también se aplica una capa Rol Pooling que se encarga de dividir cada propuesta en una cuadrícula con un tamaño fijo agrupando las características en cada celda de la cuadrícula creando un mapa de características con

tamaño fijo, el mapa es enviado al clasificador que es una red neuronal completamente conectada que predice la clase de objeto y el desplazamiento del cuadro delimitador, las puntuaciones se encargan de determinar el objeto mientras que los desplazamientos del cuadro delimitador refinan su ubicación; la segunda etapa se define como módulo de estimación de dimensiones dónde se realiza la estimación de dimensiones para los cuadros delimitadores (Aware, et al., 2023), en este módulo se aprovecha la elevación de la cámara desde el suelo con el fin de estimar las dimensiones de dicho cuadro para ello se hace uso de la fórmula 1 , esta ecuación describe la relación entre la distancia focal (F), la longitud de los píxeles, la distancia de la cámara (D) y el ancho de la imagen (W), para esta relación se debe tener en cuenta que la longitud del Pixel depende de la densidad de píxeles que se mide en PPI (pixeles por pulgada); la distancia focal se calcula utilizando la ecuación antes mencionada, durante esta etapa se obtienen las coordenadas del cuadro delimitador a partir de los cuales hoy se determina la longitud y el ancho del Pixel; en este caso se usó una distancia focal de 90 cm. Teniendo en cuenta que el modelo Mask R-CNN genera cuadros delimitadores de los que se puede extraer coordenadas de la esquina superior izquierda e inferior derecha de cada cuadro. Las coordenadas se pueden representar como (x1, y1) que es la esquina superior izquierda y (x2, y2) que es la esquina inferior derecha. Para calcular la longitud y el ancho de cada cuadro delimitador hicieron uso de las siguientes ecuaciones donde Longitud=y_2-y_1 y Ancho=x_2-x_1; finalmente se hace uso de la fórmula para el cálculo del área de un rectángulo con el fin de calcular el área de cada bache donde Area=Largo *Ancho. Para el análisis de los resultados obtenidos, hicieron uso de mAP (mean Average Precision) que se define como una métrica utilizada para medir el rendimiento de un modelo que se centra en tareas de detección de objetos y recuperación de información en imágenes (Henderson and Ferrari, 2017), donde se obtuvo una precisión promedio (mAP @0.5) de 0.48 lo que indica que el modelo tiene una capacidad moderada para detectar y clasificar de manera correcta los baches en las imágenes.

Chitale, el at., (2020) por su parte propusieron un sistema de estimación de dimensiones que hace uso de técnicas de aprendizaje profundo y procesamiento de imágenes, se analiza y propone el uso de Yolo que es un algoritmo que tiene la capacidad de detectar todos los objetos en una sola toma (Sharma, 2020), se hace uso de sus versiones v3 y v4 con el objetivo de comparar sus rendimientos ya que la primera (Yolo V3) compromete la velocidad con el fin de aumentar la precisión de la red general y la segunda (Yolo V4) tiene como objetivo construir un detector de objetos más rápido para sistemas de producción y optimizar la computación paralela; en este caso Yolo es aplicado

como sistema de detección de dos tipos de baches que los clasifica como anegados y secos; para el entrenamiento de la red neuronal se utilizó un dataset de 1300 imágenes provenientes de Google Imágenes que fueron procesadas por una herramienta de código abierto llamado "LabelImg" donde se realizan las anotaciones para adaptar estas imágenes a los estándares de Yolo. Tanto Chitale, et al., (2020) como Aware, et al., (2023) dividen la metodología en dos etapas, la primera relacionada con la detección de baches hace uso de Yolo y se considera la detección de objetos como como una tarea de regresión, los cuadros delimitadores se predicen con las imágenes de entrada directamente por lo que no es necesario un paso de propuesta de región por separado (Chitale, et al. 2020). En este caso no suena atractiva la idea de hacer uso de detectores de objetos basados en una arquitectura de dos etapas debido a que no son adecuadas para aplicaciones en tiempo real por su misma complejidad. Al momento de pasar la imagen de entrada por la red neuronal convolucional se obtiene una imagen codificada; Bounding Box Predictor realiza una predicción de múltiples cuadros delimitadores candidatos junto con varios anclajes; luego, se aplica una supresión que produce mejores cuadros delimitadores con el IoU más alto, que es una métrica de rendimiento que se encarga de evaluar la precisión de algoritmos de anotación, segmentación y detección de objetos (Chitale, et al. 2020). Ambos modelos se entrenaron con el dataset establecido y con múltiples iteraciones que van entre 3000, 4000, 5000 y 6000; con un lote de 64 y 8 subdivisiones durante el proceso de formación (Chitale, et al. 2020). En el segundo módulo de estimación de dimensiones se hace uso de la propiedad de similitud de triángulos que se define por la ecuación:

$$F = (P * D)/W \tag{1}$$

Donde F es la longitud focal de la cámara, P equivale a el ancho del objeto en píxeles de la imagen, D representa la distancia conocida entre la cámara y el objeto en una imagen de referencia y W es el ancho real del objeto en unidades físicas. La ecuación (1) fue aplicada de igual manera por Chitale, et al., (2020) como Aware, et al., (2023); en esta etapa se generan las coordenadas del cuadro delimitador con las que se pueden obtener el ancho y la longitud de los pixeles. Utilizando la información obtenida de la distancia fija de la cámara y la distancia focal se encuentra la longitud real aplicando la ecuación (1). Finalmente, para evaluar los resultados obtenidos se hizo uso de la métrica mAP @0.5 y de IoU donde finalmente concluyeron que tanto para YoloV3 y YoloV4 la validación de la métrica mAP y IoU comienza a degradarse después de las 4000 iteraciones con lo

que se considera que las 4000 iteraciones son el punto de parada; así mismo, se calculó la tasa de error del área calculada donde se obtuvo un error del 5.868%.

Los avances en tecnologías como redes neuronales y visión por computadora han revolucionado la forma en que se abordan los desafíos relacionados con la identificación de problemas en infraestructura vial. Los enfoques automatizados presentados demuestran la viabilidad y eficacia de estos métodos para la extracción de datos relevantes, la detección de baches y la identificación de intersecciones en imágenes de alta resolución; a continuación, se presenta una tabla que contiene el resumen de los antecedentes mencionados:

Tabla 1: Estado del arte sobre autores que trataron el mismo caso.

Articulo	Autores	Año	Objetivos	Dataset	Herramientas	Preprocesamiento y etiquetado	Tecnica de deteccion utilizada	Métricas	Resultados
Pothole Detection in Asphalt Pavement Images	C. Koch y I. Brilakis	2011	Presentar un enfoque novedoso para la deteccion automatizada de baches basado en imagenes de la superficie del pavimento asfaltico	120 imágenes : 50 para entrenamiento y 70 de prueba	Robot que hace la simulacion de vehiculo montado con una camara de alta velocidad	Conversion a escalas de grises	Algoritmos de deteccion de baches	Precisión	81.6
								Accuracy	85.9
								Recall	86.1
Pothole Detection and Area Estimation System Using Mask R-CNN	S. Aware, T. Deore, S. Khatale, K. Malode y Chaudhari	2023	Desarrollar un sistema automatizado para la detección de baches y estimación de área usando Mark R-CNN	1283 imágenes de Google Imágenes: 1128 para entrenamiento y 140 para prueba	No hay Información	Make Sense	Mask R-CNN	mAP @ 50	0.48
Pothole Detection and Dimension Estimation Using Deep Learning (Yolo) and Image Processing	P. Chitale, H. Shenai, Gala, Jay, K. Kekre y R. Karani	2020	Proponer un sistema de estimacion de dimensiones de baches que usa un sistema con aprendizaje profundo y procesamiento de imagenes	1300 imagenes de Google Imagenes	No hay Información	LabelImg	YOLOV3	mAP @50	0.889
								IoU	0.635
							YOLOV4	mAP @50	0.933
								IoU	0.741
Pothole Classification Model Using Edge Detection in Road Image	Baek, J. W., & Chung, K.	2020	Proponer un modelo de clasificacion de baches utilizando la deteccion de bordes en imagenes de carreteras.	13376 imagenes donde: 2675 para validacion 1338 para prueba 9363 para entrenamiento	Computador con Windows 10 Intel Core i5-8400 16GB de RAM y 2070 super GPU	Conversion a escalas de grises	YOLO	Precision	0.8345
								Accuracy	0.7786

Articulo	Autores	Año	Objetivos	Dataset	Herramientas	Preprocesamiento y etiquetado	Tecnica de deteccion utilizada	Métricas	Resultados
Deep Learning Based Detection of Potholes in Indian Roads Using YOLO	Dharneeshkar, J., Aniruthan, S. A., Karthika, R., & Parameswaran, L.	2020	Proporcionar un sistema que pueda detectar baches a partir de una imagen 2D con mayor precision y mayor velocidad.	1500 imágenes tomadas desde el tablero de un automóvil con una cámara de Iphone 7 1200 imagenes para entrenamiento y 300 imagenes para testear	No hay Información	LabelImg	YOLOV3 (416*416)	mAP @0.25	58.79
								mAP @0.50	38.41
								IoU	46.29
								Presicion	0.69
								Recall	0.35
							YOLOV2 (416*416)	mAP @0.25	64.05
								mAP @0.50	39.46
								IoU	31.6
								Presicion	0.46
								Recall	0.45
							YOLOV2 (608*608)	mAP @0.25	68.57
								mAP @0.50	45.33
								IoU	34.77
								Presicion	0.52
								Recall	0.5
							Tiny YOLOV3 (608*608)	mAP @0.25	72.12
								mAP @0.50	49.71
								IoU	50.96
								Presicion	0.76
								Recall	0.4
Detection of Potholes Using a Deep Convolutional Neural Network	Suong, L. K., & Jangwoo, K.	2018	Abordar el problema de la detección de baches utilizando técnicas basadas en aprendizaje profundo. Detectar obstáculos como grietas y baches en las carreteras.	1199 imágenes de Flickr, Google Images y Pixabay 996 imágenes para entrenamiento y 203 imágenes para prueba	NVIDIA GeForce GTX 1080 10 GB de RAM	LabelImg	F1 (YOLOV2)	Presicion	60.14
								Recall	65.61
							F2-Anchor	Presicion	67.74
								Recall	74.93
							Den-F2-Anchor	Presicion	82.43
								Recall	83.72

Articulo	Autores	Año	Objetivos	Dataset	Herramientas	Preprocesamiento y etiquetado	Tecnica de deteccion utilizada	Métricas	Resultados
An Architectural Multi-Agent System for a Pavement Monitoring System With Pothole Recognition In UAV Images	L. Silva, H. Sanchez, D. Peral, A. Sales & G. Villarubia	2020	Diseñar una plataforma distribuida que permita la detección de daños en las rutas de transporte utilizando drones y proporcionar los resultados de los clasificadores más importantes	600 imágenes con una resolucion de 3840x2160 píxeles capturadas con un drone DJI Mavic Air 2; 70% para entrenamiento, 20% para validación y 10% para pruebas	DJI Mavir Air 2 Nvidia Jetson Tesla K80 12GB	Redimensionamiento LabelImg Cambio de orientación Magnificación	YOLOV4-Tiny	AP %	98.45
								Presicion %	96.54
Pothole Detection System Using Yolo V8	Dr. S. N Gujar, Prathmesh Shinde, Shlok Parihar, Aditya Pawar, Saikrishna Yemul	2024	Crear y evaluar un modelo de aprendizaje automático especialmente entrenado para la detección de baches en carreteras en tiempo real	No hay Información	No hay Información	No hay Información	YOLO V8	mAP@0.5 %	95.82
								Presicion %	95
								Recall %	97
Pothole Detection System Using YOLO v4 Algorithm	Kshitija Chavan, Chinmay Chawathe, Vatsal Dhabalia, Amruta Sankhe	2022	Aautomatizar la tarea de detección de baches mediante un enfoque de aprendizaje profundo para alertar a los conductores	No hay Información	PC con GPU Nvidia 1660 y 8 GB de RAM	No hay Información	YOLO V4	Precisión Promedio %	85-90
YOLO V8 and Point Cloud Fusion for Enhaced Road Pothole Detection and Quantification	Junkui Zhong, Deyi Kong, Yuliang Wei y Bin Pan	2.025,00	Introducir un método novedoso que integra imágenes de cámaras de profundidad con datos de nubes de puntos para detectar baches irregulares en la superficie de la carretera	Dataset de 5,000 imágenes de alta resolución (640x640) de baches en carreteras para el entrenamiento del modelo	Cámara de profundidad RealSense D435i Cámara de profundidad RealSense D435i (con sensor RGB de 20 megapíxeles y sensor de profundidad 3D)	No hay Información	YOLO V8	F1 %	94.53
								Presicion %	95.8
								Recall %	93.3

Marco teórico

Dentro de la detección de objetos se encuentran las redes neuronales artificiales que son un modelo que imitan un comportamiento como el de las redes neuronales biológicas, cuenta con entradas x_i que están ponderadas por pesos w_i y son combinados con la función de red, lo que define el nivel de potencial de la neurona. Posteriormente, la salida de la función de la red se evalúa en la función de activación que da como resultado la salida de la unidad de proceso (Delgado, 2010).

Ecuación de modelo de una red neuronal simple:

$$y = W_0 + W_1X_1 \ldots + W_nX_n \quad (2)$$

Donde:

$\boldsymbol{x_i}$= Entradas

$\boldsymbol{W_i}$= Pesos

Recordando las partes en las que se conforman las redes neuronales especialmente lo que son los nodos, se debe hablar sobre las funciones de activación que se aplican a la salida o final de estos nodos, entre estas funciones de activación están:

Función de activación Sigmoidea/Logística:

Selecciona cualquier valor real como una entrada y después genera valores desde 0 hasta 1; entre mayor el valor de entrada estará más cerca del valor de salida 1,0 y mientras más pequeño el valor de entrada, el valor de salida se aproxima más a la salida 0,0.

$$f(x) = \frac{1}{1+e^{-x}} \quad (3)$$

Esta función se utiliza para modelos donde se debe predecir la probabilidad como resultado dentro del rango de 0 y 1. Así mismo, esta función es diferenciable y tiene un gradiente suave que evita que existan grandes saltos en los valores de salida. También presenta una limitación en cuanto a su derivada porque genera valores representativos de -3 a 3 lo que significa que para valores que sobrepasen este rango, se tendrán gradientes muy pequeños próximos a 0 lo que provoca que la red deje de aprender así mismo el entrenamiento de la red neuronal se vuelve más difícil e inestable (Pragati, 2021).

Función ReLU:

Su nombre hace referencia a unidad lineal rectificada, se conforma de una función derivada y permite la retropropagación, también la hace eficiente de manera computacional; esta función presenta un inconveniente debido a que no activa el total de neuronas simultáneamente y también que estas neuronas se desactivan únicamente si la salida de la transformación lineal es menor que cero (Pragati, 2021).

$$f(x) = \max(0, x) \tag{4}$$

Utilizar esta función trae algunas ventajas como una mayor eficiencia computacional en comparación a funciones como sigmoidea dado que esta función solo activa una cierta cantidad de neuronas; así mismo acelera la convergencia del gradiente hacia el mínimo global de la función de pérdida; también presenta una limitación debido a que el lado negativo del gráfico hace que el gradiente también sea cero que provoca como consecuencia que procesos de retropropagación, pesos y sesgos de algunas neuronas no se actualicen por lo que crean neuronas muertas (Pragati, 2021).

Función Swish:

Esa función de activación automática fue desarrollada por Google y se utiliza en redes profundas aplicadas a clasificación de imágenes, traducción automática, entre otros temas; esta función se encuentra acotada por debajo pero no acotada por arriba, entonces cuando *x* se acerca al infinito negativo *y* se acerca a un valor constante, pero cuando *y* se acerca al infinito *x* también se acerca al infinito (Pragati, 2021). La función Swish se puede ver representada por la ecuación (5).

$$f(x) = x * sigmoid(x) \tag{5}$$

Algunas de las ventajas de la función Swish en comparación a la función ReLU es su suavidad debido a que no presenta cambios abruptos de dirección cuando se acerca en x igual a cero; al no ser monótona se mejora la expresión de los datos de entrada y el peso que se va a aprender (Pragati, 2021).

Precisión:

Esta se define por la ecuación (6).

$$Precision = \frac{TP}{TP+FP} \tag{6}$$

Donde TP son las fracciones de verdaderos positivos, FP se refiere a fracciones de falsos positivos y FN se refiere a fracciones de falsos negativos (Minaee, et al. 2021), otras ecuaciones que hacen uso de estas fracciones son:

Recall:

$$Recall = \frac{TP}{TP+FN} \tag{7}$$

Puntaje de falsos negativos o FNR:

$$FNR = \frac{FN}{TP+FN} \tag{8}$$

puntaje de falsos positivos o FPR:

$$FPR = \frac{FP}{TN+FP} \tag{9}$$

Una métrica popular que contiene un equilibrio entre las ecuaciones de precisión y recall es conocida como F1-Score Y se define por la ecuación (10).

$$F1 - Score = \frac{2\ Prec\ Rec}{Prec+Rec} \tag{10}$$

Precisión promedio media (mAP):

Es una métrica bastante conocida que es aplicada para medir el rendimiento de modelos de visión por computadora, mAP es igual al promedio de la métrica de Precisión promedio en todas las clases de un modelo (Solawetz, 2020).

$$mAP = \frac{\sum_{q=1}^{Q} AveP(q)}{Q} \tag{11}$$

Donde Q es el número de consultas en el conjunto y *AveP(q)* es la precisión promedio *(AP)* para una consulta determinada q.

F1 Score:

Es una métrica que combina precisión y recall para representarlas simétricamente a través de una media armónica.

$$F1 = 2\frac{P \times R}{P + R} \tag{12}$$

Intersección sobre Unión (IoU):

Se utiliza en modelos de detección de objetos para medir la precisión de la localización y calcular errores de localización (Shah, 2023), su fórmula es la siguiente:

$$IoU = \frac{TP}{TP + FN + FP} \tag{13}$$

Transformación de pixeles a unidades reales:

Para obtener las dimensiones reales de los baches por medio de las imágenes capturadas por VANT´S es necesario hacer uso de las coordenadas de las cajas delimitadoras (x_1,y_1) y (x_2,y_2); y conocer ciertas características del VANT como puede ser el FoV (Wikimedia Foundation Inc, 2025) (Callari, 2015) (Barris, n.d.).

Para lentes rectilíneos, la definición del ángulo de visión α en cualquiera de sus dimensiones (diagonal, horizontal o vertical) es:

$$\alpha = 2 \times arctan\frac{d}{2f} \tag{14}$$

Siendo θ el FoV diagonal:

$$tan\frac{\theta}{2} = \frac{(\sqrt{w^2 + h^2})/2}{f} = \frac{d_{diag}/2}{f} \tag{15}$$

Donde d es la dimensión del sensor en esa dirección y f la distancia focal. Esta ecuación relaciona directamente el ángulo diagonal con el espacio tangente del sensor (The VRguy's Blog, 2013).

Definiendo $AR = \frac{w}{h}$, se tiene

$$\sqrt{w^2 + h^2} = h\sqrt{AR^2 + 1} \tag{16}$$

$$w = ARh \tag{17}$$

De modo que

$$\frac{w/2}{f} = \frac{(ARh)/2}{f} = tan\frac{\theta}{2} \times \frac{AR}{\sqrt{AR^2+1}} \quad (18)$$

Y similarmente

$$\frac{h/2}{f} = \tan\left(\frac{\theta}{2}\right) \times \frac{1}{\sqrt{AR^2+1}} \quad (19)$$

Sustituyendo estas expresiones en la formula genérica $\alpha = 2 \times arctan\frac{d}{2f}$ para $d = w$ y $d = h$, se obtienen las siguientes ecuaciones:

$$FoV_h = 2 \times \tan^{-1}\left(\tan\left(\frac{\theta}{2}\right) \times \frac{relación\ aspecto}{\sqrt{relación\ aspecto^2+1}}\right) \quad (20)$$

$$FoV_v = 2 \times \tan^{-1}\left(\tan\left(\frac{\theta}{2}\right) \times \frac{1}{\sqrt{relación\ aspecto^2+1}}\right) \quad (21)$$

Finalmente se hace uso de las siguientes ecuaciones para encontrar las dimensiones reales del objeto

$$W = 2 \times H \times \tan\left(\frac{FoV_h}{2}\right) \quad (22)$$

$$L = 2 \times H \times \tan\left(\frac{FoV_v}{2}\right) \quad (23)$$

$$W_{real} = \frac{\Delta x}{I_w} \times W \quad (24)$$

$$H_{real} = \frac{\Delta y}{I_h} \times L \quad (25)$$

Herramientas y Materiales

Para la obtención del modelo de detección y la aplicación web, fue necesario hacer uso de algunas herramientas y materiales. Esta selección incluye tanto el hardware necesario para el entrenamiento y captura de datos, como el software y plataformas utilizadas para el desarrollo, programación, almacenamiento y control de versiones. Cada herramienta cumple un rol específico que facilita la adquisición de imágenes y la implementación de la aplicación web.

Tabla 2: Herramientas y materiales utilizados para el desarrollo del modelo y de la aplicación web.

Herramienta	Clase	Función	Características
ACER NITRO AN515-52	Hadware	Almacenamiento de conjunto de datos (imagenes y anotaciones) y almacenamiento de modelos entrenados con metadatos Calculo intensivos debido a las operaciones que conlleva el entrenamiento de Yolo que hace uso de la GPU del equipo Proporcionar una interfaz desde la que se puede interactuar con el sistema	Procesador: Intel(R) Core(TM) i5-8300H CPU @ 2.30GHz RAM: 16,0 GB Tarjeta Grafica: NVIDIA GEFORCE GTX 1050 Disco: Solido marca Samsung de 1T
DJI Phantom 4	Hadware	Capturar imagenes de baches de manera horizontal a una determinada altura	Video: 4K Fotografía: 20 megapixeles Tiempo de vuelo maximo: 30 minutos aproximadamente Tamaños de imagen: 5472x3648 4864x3648 5472x3078
GitHub	Herramienta de desarrollo de software y servicio de alojamiento de repositorios de codigo	Proporcionar los repositorios necesarios para clonar en Visual Studio Code, como los de LabelImg y YOLO	GitHub usa Git para realizar un seguimiento de los cambios en el codigo furnte Facilita la colaboracion simultanea entre personas para un mismo proyecto Aloja diversos repositorios de codigo publico y privado
LabelImg	Software	Dibujar cajas delimitadoras al rededor de los baches para poder etiquetarlos Guardar las anotaciones de los baches en archivos de texto que contienen la clase y las coordenadas de la caja delimitadora	Interfaz grafica intuitiva para dibujar cajas delimitadoras al rededor de objetos en las imágenes y etiquetarlos.
YOLOV7	Modelo de deteccion de objetos	Procesar imagenes de baches a gran esala para poder identificarlos	Alcanza un mAP del 56.8% en el conjunto de datos COCO test-dev Puede procesar imágenes a una velocidad de hasta 140 frames por segundo(FPS) en una GPU V100
Visual Studio Code	Software	Proporcionar un entorno completo para ecribir, compilar y depurar el codigo de YOLOV7	Cuenta con una interfaz intuitiva que facilita la escritura, edición y depuración de código. Esta integrada con GIT que facilita la obtencion y clonacion de código. Cuenta con terminal integrada para ejecutar comandos y scripts.
Python	Lenguaje de programación	Facilitar el desarrollo, entrenamiento e implementación del modelo de aprendizaje profundo (Yolov7)	Versión 3.12.64
CUDA	Plataforma de computación paralela y un modelo de programación desarrollados por NVIDIA	Proporciona la capacidad de procesamiento necesario para llevar a cabo tareas intensivas en cálculos de manera eficiente y en tiempo real; permite aprovechar la arquitectura paralela de las Unidades de Procesamiento Gráfico (GPU) de NVIDIA	
Postman	Herramienta para desarrollar, probar y gestionar API	Probar los end points creados en el backend	
Postgres	Sistema de gestión de base de datos relacional de código abierto	Como base de datos	
Pgadmin	Heramienta para la administración y desarrollo de código abierto para postgres SQL	Facilitar la interacción con la base de datos	

METODOLOGÍA

Diseño de la Investigación

Este proyecto tiene como principio un enfoque cuantitativo y experimental; cuyo objetivo principal es desarrollar y evaluar un modelo de inteligencia artificial para la detección de baches en pavimentos asfálticos, además de integrarlo con una aplicación web que permita obtener las dimensiones de los baches detectados. La investigación se organiza en dos fases clave: la obtención del modelo de detección y el desarrollo de la aplicación web.

La elección del modelo YOLOV7-Tiny para la detección de baches surgió debido al costo computacional que es menor al de YOLOV7; el modelo YOLOV7-Tiny está diseñado para ser más compacto, eficiente, con menos parámetros y operaciones (Wang, Bochkovskiy y Liao, 2023). Lo que se adecua perfectamente teniendo en cuenta que en el entrenamiento del modelo se utilizó el parámetro *multi-scale* que cambia dinámicamente el tamaño de la imagen, por lo que el modelo procesa entradas con diferentes dimensiones, que afecta el tamaño de los tensores internos y la cantidad de operaciones necesarias. Imágenes más grandes implican más píxeles y por tanto más cálculos (Redmon and Farhadi, Yolov3: An incremental improvement, 2018).

La primera fase se dedica al entrenamiento y validación del modelo YOLOv7-Tiny; el modelo es entrenado con dos dataset, el primero con mayor cantidad de imágenes a comparación del segundo dataset, lo que contribuye a mejorar su capacidad de generalización. El rendimiento del modelo se evalúa mediante métricas como la precisión promedio (mAP), el Intersection over Union (IoU)y el F1 Score; para asegurar que el modelo localice y clasifique correctamente los baches.

La segunda fase consiste en el desarrollo de una aplicación web que integra el modelo entrenado. Esta aplicación permite a los usuarios crear proyectos donde pueden cargar imágenes de pavimentos asfalticos, obtener su ubicación, procesarlas para detectar los baches y obtener información sobre sus dimensiones. El backend de la aplicación está construido con Python y FastAPI, lo que facilita la inferencia del modelo y la gestión eficiente de las solicitudes de los usuarios. El frontend está desarrollado utilizando JavaScript y React, proporcionando una interfaz interactiva y fácil de usar para que los usuarios puedan visualizar y analizar los resultados.

Este diseño de investigación se enmarca en un enfoque cuantitativo porque se hace uso de datos numéricos para evaluar tanto el rendimiento del modelo como el funcionamiento de la aplicación

web. Las métricas clave de desempeño incluyen la precisión del modelo (medida mediante mAP e IoU), el tiempo de inferencia y la comparación entre las dimensiones estimadas de los baches y las medidas reales obtenidas manualmente.

El diseño adoptado es experimental, dado que se manipulan diversas variables, como los parámetros del modelo YOLOv7-Tiny y las condiciones de prueba, para observar su impacto en los resultados obtenidos. Durante esta fase, se realizan pruebas con diferentes imágenes, lo que permite evaluar la efectividad del sistema en situaciones reales, como cambios en la iluminación y la calidad de las imágenes.

La investigación también sigue un enfoque aplicado, ya que el propósito no es solo medir el rendimiento del modelo, sino también implementarlo en una herramienta funcional que pueda ser utilizada en el mundo real por técnicos o ingenieros encargados de la inspección de pavimentos. Mediante la integración del modelo en la aplicación web, se proporciona una solución que permite crear proyectos sobre la vía que se va a analizar y obtener información detallada sobre los baches detectados, lo que facilita la toma de decisiones en el proceso de mantenimiento de las infraestructuras viales.

Población y Muestra

La población en este estudio está compuesta por imágenes de pavimentos asfálticos con baches, utilizadas para entrenar y validar el modelo de detección de baches. Para este propósito, se utilizan dos conjuntos de datos distintos, los cuales se usan en diferentes fases del entrenamiento del modelo, con el objetivo de mejorar la precisión en la detección de baches en diversas condiciones y situaciones.

El primer dataset utilizado para el entrenamiento inicial del modelo consiste en 936 imágenes de baches en pavimentos asfálticos, obtenidas del estudio (Silva, et al., 2020). Como se observa en la figura 9, este conjunto de datos incluye una variedad de imágenes que muestran diferentes tipos de baches, con diferentes tamaños, lo que permite al modelo aprender a identificar los baches en situaciones generales. Las imágenes fueron anotadas manualmente con el uso de LabelImg para identificar las ubicaciones y características de los baches, lo que facilita el entrenamiento del modelo. Aunque este conjunto de datos proporciona una base inicial sólida, las imágenes no están adaptadas a un contexto o ubicación específicos, lo que implica que el modelo entrenado con este

conjunto pueda tener limitaciones para generalizar a situaciones reales y específicas de pavimentos de una vía concreta.



Figura 9. Imágenes de baches para primer entrenamiento del modelo.

Fuente: Tomado de https://www.mdpi.com/1424-8220/20/21/6205

Para mejorar la precisión y adaptabilidad del modelo, se emplea un segundo dataset, que consta de 61 imágenes capturadas específicamente por el investigador con un dron en una vía de pavimento asfáltico localizada entre el municipio de Sibundoy y San Francisco en el departamento del Putumayo. Al momento de hacer el vuelo para la conformación de imágenes para el dataset, se presentaron limitaciones en cuanto a poder realizar el vuelo con una altura estable y fija, debido a que el dron con el que se realizó el vuelo no contaba con un sistema de RTK y sólo se contaba con la altura de vuelo con respecto al punto de despegue; aun así, se trató de tener una altura de vuelo de 7 metros; como se observa en la figura 10 estas imágenes son representativas de un entorno particular y proporcionan una perspectiva más cercana a las condiciones reales en las que el modelo será implementado. Al ser imágenes tomadas en una vía concreta, las condiciones de iluminación, los ángulos y la calidad de las imágenes pueden variar, lo que refleja de manera más precisa el entorno en el que el modelo de detección de baches será utilizado. Este conjunto de datos tiene una gran relevancia ya que, al tratarse de imágenes obtenidas directamente en el contexto de uso real, permite al modelo ajustarse mejor a las condiciones específicas de la vía estudiada.



Figura 10. Imágenes de baches capturadas para el segundo entrenamiento y validación de la vía Sibundoy – San Francisco en el departamento del Putumayo.

El modelo entrenado inicialmente con el primer dataset se afina y ajusta utilizando las 61 imágenes capturadas con el dron. Este enfoque, que combina un entrenamiento inicial más general con un ajuste específico utilizando datos más contextualizados, tiene como objetivo mejorar la capacidad del modelo para detectar baches en situaciones reales y específicas, como las que pueden encontrarse en una vía pavimentada en particular. Al incorporar estas imágenes más específicas, el modelo tiene más oportunidades de aprender las características y variaciones que podrían no estar presentes en un conjunto de datos general, aumentando su capacidad de generalización y precisión.

Variables e Indicadores

En este estudio, las variables e indicadores clave se centran en la evaluación del modelo de detección de baches y la eficiencia de la aplicación web en cuanto a la ejecución de la función de identificación de baches. A continuación, se presentan las variables principales utilizadas para medir el rendimiento del sistema.

La precisión del modelo de detección de baches es una de las variables más relevantes en este estudio. Esta variable mide la capacidad del modelo para identificar correctamente los baches en las imágenes de pavimentos asfálticos. Para evaluar esta precisión, se emplean dos indicadores clave.

Precisión Promedio (mAP): Como se definió en el marco teórico, el mAP es una métrica utilizada para evaluar la precisión del modelo en detección de objetos. En este caso, se aplica para medir la precisión global del modelo al identificar baches en las imágenes.

Intersection Over Union (IoU): Como se mencionó previamente, el IoU es una métrica que mide la superposición entre las cajas delimitadoras predichas por el modelo y las reales. Un alto IoU indica que el modelo está localizando los baches con alta precisión.

La precisión en las dimensiones de los baches es también una variable importante que mide la exactitud de las dimensiones del bache detectado, comparando las dimensiones predichas por el modelo con las dimensiones reales anotadas manualmente. Para esta variable, se emplean dos indicadores: el error absoluto en las dimensiones, que calcula la diferencia entre las dimensiones estimadas por el modelo y las reales, y el porcentaje de error, que se obtiene dividiendo el error absoluto por la medida real del bache, proporcionando una medida relativa de la precisión de las dimensiones detectadas.

En cuanto a la aplicación web, el principal indicador evaluado es el tiempo de ejecución de la función de identificación. Este indicador mide el tiempo que toma la aplicación para cargar una imagen, procesarla a través del modelo de detección y devolver los resultados de identificación del bache. Este tiempo es clave para evaluar la eficiencia de la aplicación en un entorno de ejecución en computador, ya que la función de identificación debe ejecutarse rápidamente para proporcionar una experiencia de usuario fluida.

Obtención del Modelo

Para obtener el modelo entrenado fue necesario seguir una serie de pasos que abarcan desde la adquisición de los datos hasta el entrenamiento y validación del modelo de detección de baches. La figura 11 corresponde al flujo de trabajo, donde se especifican los procedimientos llevados a cabo para cumplir parte de los objetivos propuestos.

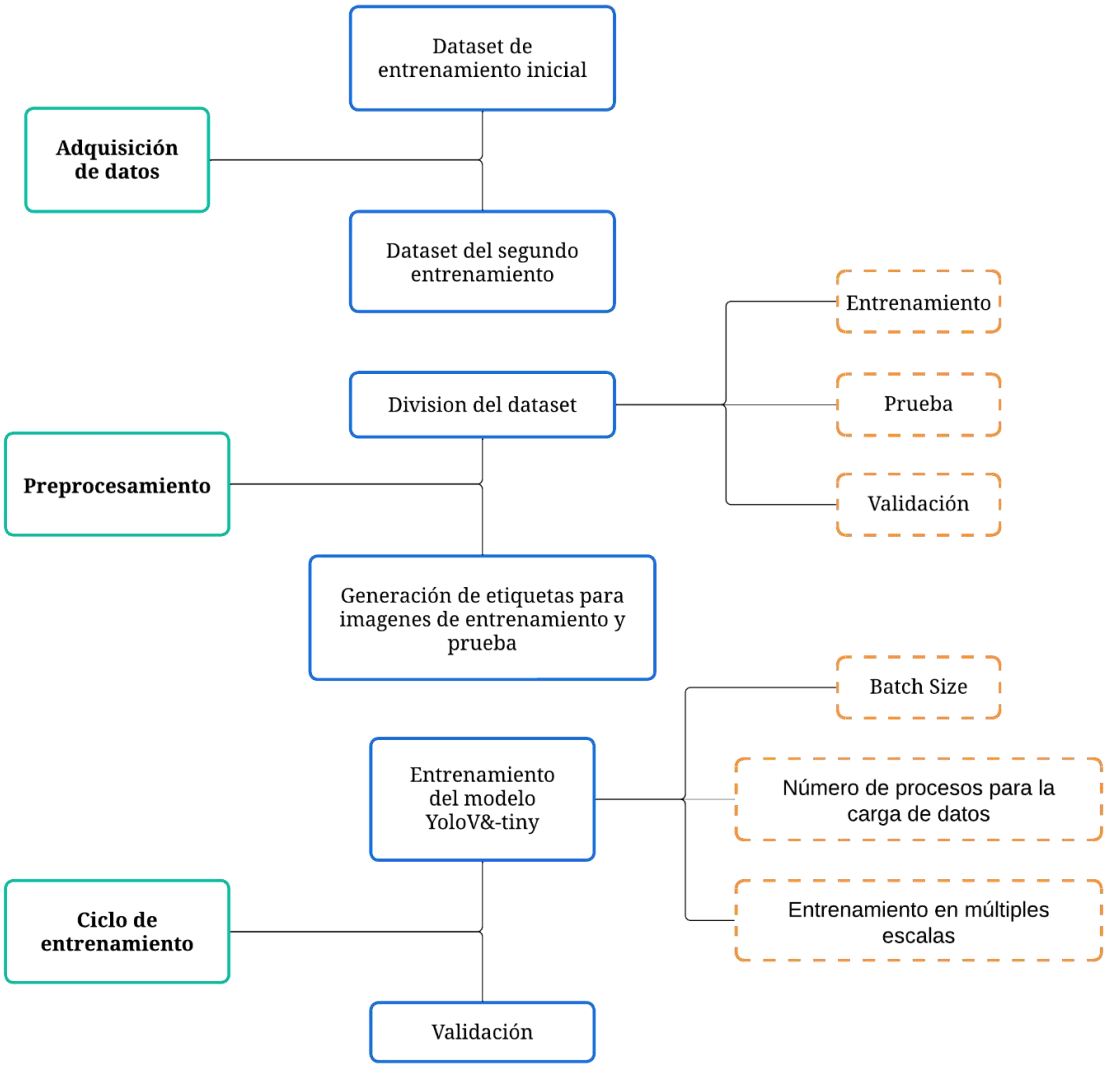


Figura 11. Flujo metodológico para la obtención del modelo de detección de baches en pavimento asfáltico.

Adquisición de Datos

Para el entrenamiento del modelo de detección de baches, se utilizaron dos conjuntos de datos diferentes. El primer dataset proviene de un artículo de Silva et al., (2021) que originalmente contiene 4845 imágenes de las que se extrajeron un total de 936 imágenes con dimensiones de 640x640 píxeles debido a que las otras imágenes no correspondían a baches sino a grietas por lo que no era útil para este caso de detección de baches.

El segundo dataset, utilizado para ajustar y mejorar el modelo, está compuesto por 61 imágenes con dimensiones de 4000x2250 píxeles capturadas con un dron DJI Mini SE en una vía de pavimento asfáltico localizada entre Sibundoy y San Francisco en el departamento del Putumayo.

Preprocesamiento de Datos

El preprocesamiento de los datos comenzó con la división de los dataset en tres partes: entrenamiento, validación y prueba. El conjunto de datos se dividió de manera que hubiera un equilibrio entre las subclases implícitas (imágenes sin baches, imágenes con un solo bache e imágenes con varios baches) para asegurar una distribución equilibrada de imágenes en cada subconjunto, permitiendo una evaluación justa del rendimiento del modelo. Aproximadamente el 70% de las imágenes de cada dataset se utilizaron para el entrenamiento, el 15% para validación y el 15% restante para la prueba. Esta división garantiza que el modelo tenga suficientes datos para aprender durante el entrenamiento y que se pueda evaluar su rendimiento en datos no vistos durante el proceso de entrenamiento.

Una vez realizada la división, las imágenes destinadas a entrenamiento y validación fueron redimensionadas con una técnica de "cropping" que aun que involucra la perdida de información, esta contenía en su mayoría información del contexto alrededor de la vía, que no fue removida del todo ya que después de aplicar la técnica de "cropping" las imágenes contenían información de su contexto. Finalmente se pasó de tener imágenes con dimensiones de 4000x2250 a 640x640 píxeles, estas son las dimensiones de entrada requeridas por el modelo YOLOv7-Tiny. Este paso es crucial para asegurar que todas las imágenes tengan las mismas dimensiones y puedan ser procesadas de manera uniforme por la red neuronal. Hay que resaltar que para el segundo entrenamiento se realizó un giro de 180° al dataset de entrenamiento con el objetivo de aumentar la cantidad de imágenes de 61 a 122 lo que se puede considerar como un paso de "data augmentation".

Además del giro de 180°, se evaluaron otras técnicas de "data augmentation" como rotaciones menores, variaciones de brillo y contraste, traslaciones y escalado, sin embargo, estas transformaciones no mejoraron el rendimiento del modelo. En las imágenes aéreas capturadas por dron, las rotaciones pequeñas y traslaciones alteraron la relación geométrica entre la cámara y la superficie del pavimento, generando cajas delimitadoras inconsistentes y aumentando los falsos positivos. Así mismo, las variaciones fotométricas afectaron la textura del asfalto que es clave para

diferenciar baches de sombras, manchas o irregularidades superficiales, lo que degradó la precisión y el recall durante las pruebas preliminares.

Después de redimensionar las imágenes, se llevó a cabo la generación de etiquetas para cada una de las imágenes, utilizando la herramienta LabelImg. Esta herramienta permitió crear cajas delimitadoras alrededor de los baches y asignarles la clase correspondiente. Las etiquetas generadas fueron almacenadas en un formato compatible con el modelo YOLOv7-Tiny, asegurando que el modelo pudiera aprender a identificar y localizar correctamente los baches en las imágenes durante el entrenamiento.

Ciclo de Entrenamiento

El ciclo de entrenamiento del modelo se dividió en dos partes para optimizar el rendimiento y adaptar el modelo a los diferentes conjuntos de datos disponibles.

Primer Entrenamiento

Se entrenó el modelo utilizando el primer dataset obtenido del artículo de Silva, et al., (2020) con imágenes de baches en pavimentos asfálticos, con una configuración específica para el modelo YOLOv7-Tiny. Se utilizaron 4 procesos de trabajo para cargar los datos de manera paralela. Esto se eligió debido al uso de multi-scale en el entrenamiento, que aumentó la carga computacional y, debido a limitaciones de la GPU, no se pudieron usar más de 4 hilos de procesamiento; el tamaño del lote se configuró en 4, ya que con el entrenamiento multi escala, la memoria de la GPU no soportaba un tamaño de lote mayor sin causar problemas de memoria; el entrenamiento se configuró inicialmente para 300 épocas, pero se detuvo antes de alcanzar la época 200, ya que el modelo alcanzó su límite de aprendizaje y las métricas no mostraron mejoras significativas. El modelo se detuvo cuando alcanzó su punto de máxima eficiencia, y el mejor modelo obtenido fue guardado como best.pt, que luego se utilizó para el segundo ciclo de entrenamiento.

Segundo Entrenamiento

El modelo fue ajustado utilizando el archivo best.pt del primer entrenamiento y un segundo conjunto de datos que contiene 122 imágenes capturadas con un dron DJI Mini SE en una vía pavimentada específica. El objetivo de esta fase fue realizar un ajuste fino (finetuning) del modelo para mejorar su rendimiento en un entorno específico y realista. Se aumentó el tamaño del batch a 16 para que aumente el número de imágenes procesadas al tiempo, el tamaño de las imágenes se

sigue manteniendo en 640x640 píxeles; el número de épocas se redujo a 100, ya que el objetivo de este ciclo era realizar un ajuste fino, y se esperaba que el modelo alcanzara su rendimiento óptimo en un menor número de épocas; también, se especificó el archivo **best.pt** del primer entrenamiento como punto de partida para este ajuste fino, asegurando que el modelo ya tuviera un aprendizaje previo con los datos iniciales; las imágenes se mantuvieron en la dimensión estándar de 640x640 píxeles, consistente con el tamaño de entrada del modelo YOLOv7-Tiny; se configuró un congelamiento de 10 con el fin de congelar las primeras 10 capas que son las encargadas de extraer las características básicas de la imagen (backbone) esto permite mantener fijos los pesos entrenados de esas capas para entrenar solamente las capas finales (neck y head), lo que también ahorra tiempo de entrenamiento y consumo de GPU.

Construcción de la Aplicación

El desarrollo de la aplicación web se llevó a cabo utilizando React para el frontend y Python FastAPI para el backend, junto con Alembic para gestionar las migraciones de la base de datos en PostgreSQL. En la figura 12 se detalla el proceso de desarrollo dividido en varias fases clave.

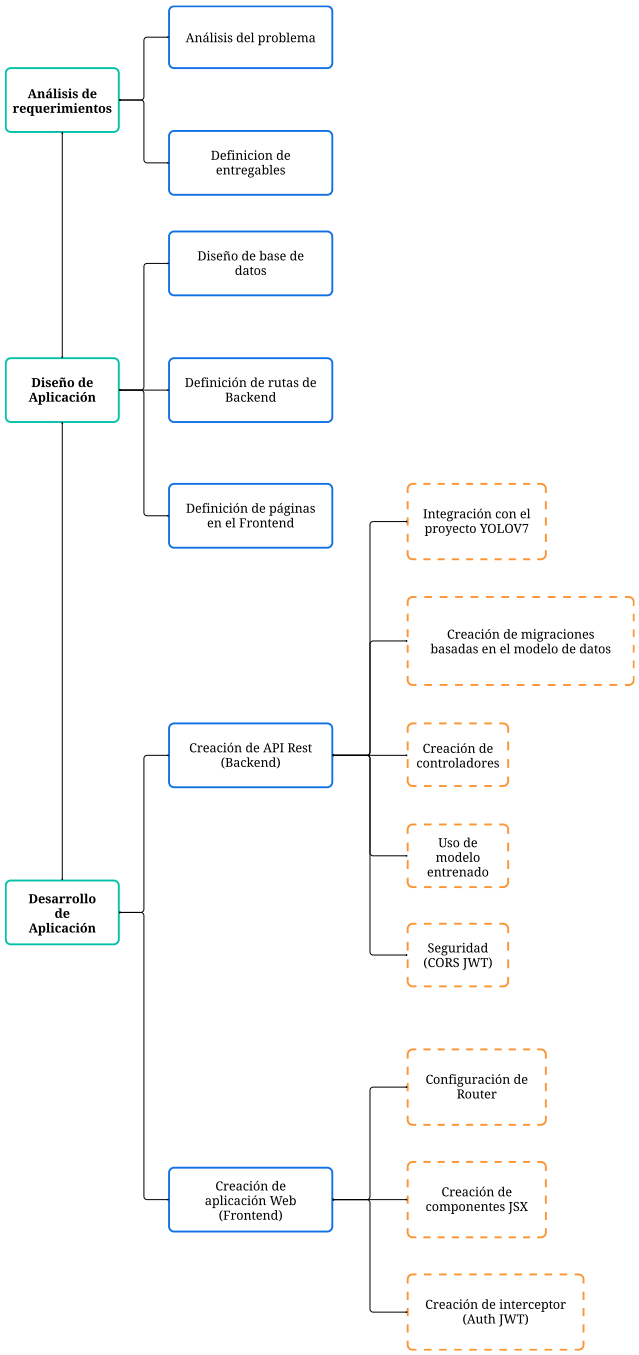


Figura 12. Flujo metodológico para el desarrollo de la aplicación web de detección de baches en pavimento asfáltico.

Análisis de Requerimientos

En esta fase inicial, se realizó el análisis del problema para comprender los requerimientos del sistema, que incluyen la carga de imágenes, la detección de baches en pavimentos asfálticos, y la visualización de los resultados. Además, se definieron los entregables claves, como la

funcionalidad de la aplicación web y las características necesarias. Se estableció que los entregables incluyen la detección de baches y la visualización de las dimensiones de estos.

Diseño de Aplicación

Diseño de Base de Datos

Se diseñó una base de datos relacional en PostgreSQL que almacena toda la información necesaria para el desarrollo de la aplicación como la información de usuario, detecciones, nombre de la imagen, guardar el proyecto en el que se está trabajando y las relaciones entre estas tablas como se muestra en la figura 13.

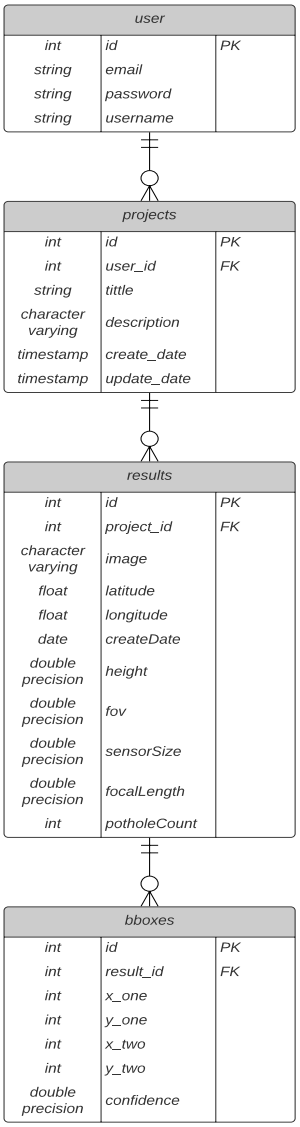


Figura 13. Diseño de base de datos relacional.

Definición de Rutas de Backend

Se definieron las rutas necesarias para la API REST en FastAPI, permitiendo que el frontend interactuara con el backend para procesar las imágenes y devolver los resultados.

Tabla 3: Definición de rutas.

Ruta	Descripción	Método	Request	Response
/detect-multiple/	Analiza las imágenes adjuntas y retorna las detecciones	POST	Files	
/register/	Crea un nuevo usuario y cifra la contraseña en base de datos	POST	Email, password, userName	message
/login/	Recibe las credenciales del usuario y genera un token de autenticación	POST	Email, password	Acces token, user info
/projects/	Crea un nuevo proyecto asociado al usuario logueado	POST	Tittle, description	Project list
/results/	Guarda el resultado de un análisis y lo asocia al proyecto activo	POST	Image, project_id, latitude, longitude, height, fov, bboxes	Id, project_id
/projects/{project_id}/results	Trae los resultados de un proyecto por medio de su ID	GET	project_id	

Definición de Paginas en el Frontend

Para este parte se tuvo que considerar que acciones podría realizar el usuario en la aplicación además de la detección de los baches, por lo que se definieron las siguientes páginas:

- **Registro o inicio de sesión:** Donde se puede crear una cuenta en caso de no tener una o iniciar sesión para acceder a la información.

- **Creación de proyectos:** En esta página se puede crear diferentes proyectos asignándoles un nombre y una descripción.
- **Proyecto:** El usuario podrá visualizar toda la información relacionada con su proyecto específicamente datos como una tabla con el resumen de sus detecciones, un mapa con las ubicaciones de las imágenes que subió, grafico sobre que imagen contiene más detecciones y botones con los que podrá realizar acciones para editar, localizar y descargar la información.
- **Analizar Imágenes:** Para que el usuario suba las imágenes sobre las que desea realizar el proceso de detección y/o medición.

Desarrollo de Aplicación

Creación de API Rest (Backend)

Primero se clonó el proyecto de yolov7 de https://github.com/WongKinYiu/yolov7 sobre este proyecto se instaló Fast API que es un framework de código abierto, a partir de este punto se crearon las rutas, la conexión con la base de datos postgres usando SQLAlchemy que es una biblioteca de Python que facilitó la interacción con la base de datos SQL, las migraciones usando ALEMBIC ya que está diseñada para usarse con SQLAlchemy; y se usaron las funciones del proyecto yolov7 para la detección utilizando el modelo previamente entrenado.

Dentro de la aplicación se crearon unas clases (modelos) de las entidades del modelo de datos diseñado además de sus relaciones, esto usando SQLAlchemy y posteriormente Alembic para migrar estos modelos y sus relaciones a la base de datos.

Creación de Aplicación Web (Frontend)

Para la creación del frontend se decidió usar React ya que es una de las librerías más populares que cuenta con mucha documentación y con la que se puede seguir el principio DRY (Dont Repeat Yourself) con el fin de no crear código que ya exista sino de aprovechar y reutilizar el código ya existente.

Se uso React Router para la configuración de las diferentes rutas y Axios para controlar las peticiones al backend y para la configuración de un interceptor que valida que el usuario este previamente logueado y tenga una sesión activa para interactuar con la aplicación.

Para dar estilos a los diferentes componentes se optó por el uso de Tailwind que es un framework de css que da una serie de clases que se pueden aplicar en los archivos .jsx y evita el uso de código css escrito de manera explícita; para este proyecto se creó una división de componentes, los primeros son paginas asociadas a una ruta que se podrían definir como componentes padres y la segunda parte comprende los componentes hijos que se reutilizan dentro de las páginas. Dado que la aplicación también cuenta con un componente espacial, hay que resaltar el uso de Leaflet como herramienta para la creación de un mapa.

Validación del modelo y de la aplicación

Para realizar el proceso de validación del modelo obtenido en conjunto con la aplicación web desarrollada, es importante comparar las detecciones del modelo con relación a la presencia real de baches en el lugar de prueba, así mismo comprobar si las medidas del bache que estima la aplicación son cercanas a las medidas reales del bache.

Por lo que se hace uso de métricas como la precisión para medir la proporción de baches que se detectaron correctamente entre todas las predicciones del modelo; por su parte el Recall dará la proporción de baches reales que el modelo detectó de manera correcta y el F1 Score que puede indicar si el modelo tiene una buena precisión y sensibilidad. Para la comparación de las medidas arrojadas por la aplicación en comparación con las medidas reales existen métricas como la de error absoluto medio (MAE) del que se obtiene la diferencia promedio entre las predicciones y las medidas reales; también es apropiado el uso del coeficiente de relación de Pearson ya que se obtiene una relación lineal entre las predicciones y las medidas reales donde un valor cercano a 1 indica que las predicciones son similares a las medidas reales.

RESULTADOS Y ANÁLISIS

En esta sección se presentan los resultados obtenidos de la metodología aplicada para la detección de baches en vías asfálticas haciendo uso de técnicas de percepción remota e inteligencia artificial como las redes neuronales. El proceso de obtención del modelo entrenado se dividió en dos entrenamientos, el primero con el fin de que el modelo aprenda las características generales de los baches en vías asfálticas por medio de un dataset amplio y a su vez para proporcionar una base al entrenamiento del segundo modelo; el segundo entrenamiento se usa para ajustar el modelo a unas condiciones más específicas como el ángulo del dron, características más específicas que los

baches pueden presentar, también para reducir los errores de localización y a su vez mejorar la detección de baches en contextos reales.

Transformación del Conjunto de Datos

El conjunto de datos obtenido de https://www.mdpi.com/1424-8220/20/21/6205 comprende un total de 4845 imágenes con unas dimensiones de 640x640 píxeles, este conjunto de datos contiene imágenes de baches, grietas y otras afectaciones del pavimento asfaltico por lo que fue necesario seleccionar sólo aquellas imágenes que contenían baches; para el segundo conjunto de datos que fue capturado con un dron DJI mini SE con el que se obtuvieron un total de 61 imágenes con dimensiones de 4000x2250 pixeles, estas dimensiones significaban una gran cantidad de parámetros y mayor tiempo de entrenamiento por lo tanto se ajustaron a 640x640 píxeles para que se tenga un tamaño consistente entre ambos entrenamientos y no afecte la estabilidad del modelo ni las características aprendidas en su primer entrenamiento; también se realizó un giro de 180° al conjunto de datos de entrenamiento con el objetivo de aumentar la cantidad de imágenes de 61 a 122.

Modelo YoloV7-Tiny

En la búsqueda de un modelo adecuado para la detección de baches en pavimento asfaltico, se realizaron diferentes entrenamientos entre los que se destacan los hechos con el modelo YoloV7-Tiny inicialmente con el conjunto de datos de https://www.mdpi.com/1424-8220/20/21/6205 y el segundo conjunto de datos con imágenes propias, se ajustaron con los parámetros:

Tabla 4: Parámetros de entrenamiento para los modelos de YOLOV7-Tiny.

Entrenamiento	Parámetros	Valor
YoloV7-Tiny-Custom Primer Modelo	Device	0
	Weights	None
	Epochs	200
	Batch Size	4
	Multi Scale	
YoloV7-Tiny-FineTuned Modelo Final	Weights	YoloV7-Tiny-Custom
	Epochs	100
	Batch Size	16
	Freeze	10

Como se observa en la tabla 4 el primer modelo se entrenó con 200 épocas, un tamaño de lote de 4, sin pesos y con la configuración de multi escala; la figura 14 muestra los resultados obtenidos por época para cada una de las métricas (Box, val box, objectness, val objectness, precisión, mAP@0.5, recall, mAP@0.5:0.95); en la figura 15 se observa a mayor detalle entre las épocas 175 y 185 el valor para la métrica recall donde el valor más alto durante todo el entrenamiento es de 0.7448 en la época 181, en la figura 16 se observa a mayor detalle entre la época 140 y 150 la precisión donde el valor más alto de todo el entrenamiento es de 0.813 en la época 146 y en la figura 17 se observa a mayor detalle entre la época 145 y 155 que el valor más alto de la métrica mAP durante todo el entrenamiento es 0.751 en la época 154; se deduce que el modelo tiene una buena capacidad para la identificación de baches, por otro lado los gráficos de Objectness indica como disminuye la pérdida a medida que el modelo aprende o mejora su habilidad para predecir si una caja contiene un bache o no.

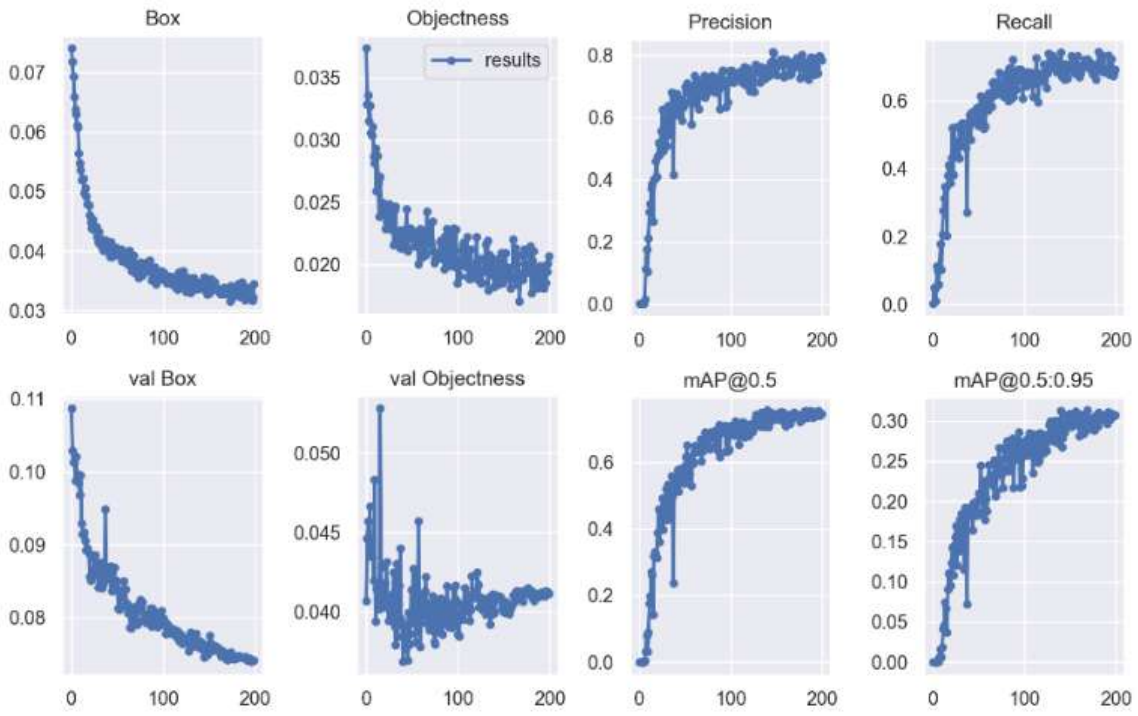


Figura 14: Resultados de YOLO V7 Tiny para 200 épocas.

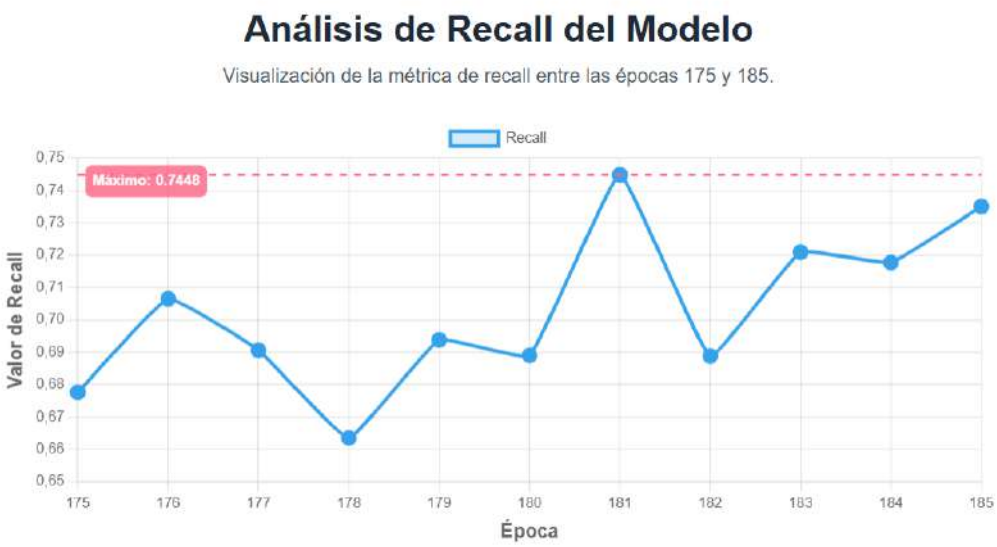


Figura 15: Valores del Recall en el primer entrenamiento entre las épocas 175 y 185.

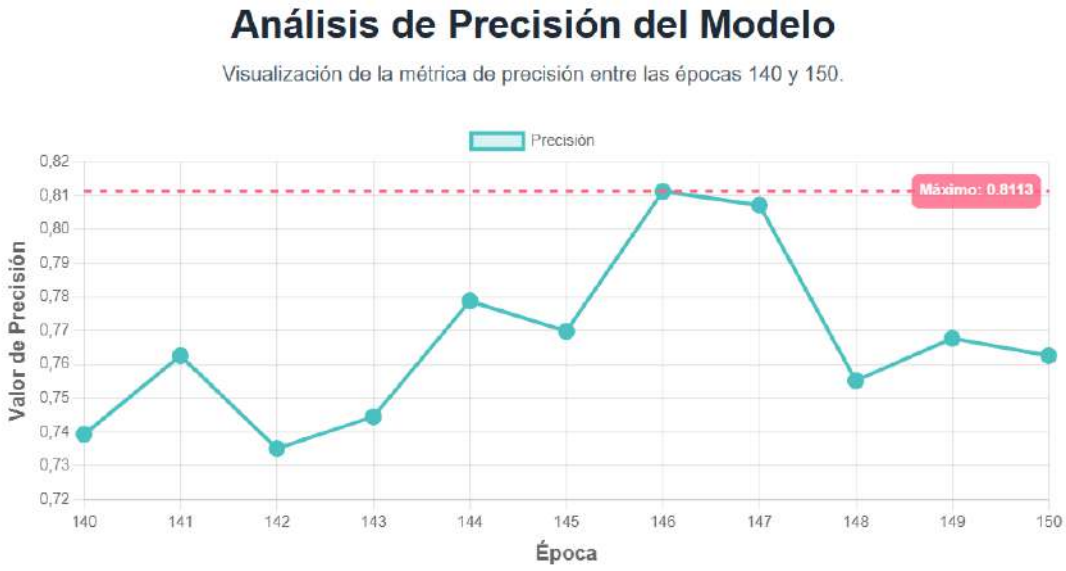


Figura 16: Valores de la precisión en el primer entrenamiento entre las épocas 140 y 150.



Figura 17: Valores de la métrica mAP@0.5 en el primer entrenamiento entre las épocas 145 y 155.

Con el fin de adaptar el modelo a condiciones reales y más cercanas a las que se pueden presentar en Colombia, se hace un segundo entrenamiento con un lote de 8, como pesos se tomó el modelo anterior y se congelaron las capas correspondientes al backbone para asegurar de que estas capas no sean alteradas con el nuevo entrenamiento; la figura 18 muestra los resultados obtenidos de varias métricas (Box, val box, objectness, val objectness, precisión, mAP@0.5, recall, mAP@0.5:0.95) a medida que las épocas de entrenamiento van avanzando; en la figura 19 se observa a mayor detalle entre las épocas 70 y 80 el valor para la métrica recall donde el valor más alto durante todo el entrenamiento es de 0.728 en la época 74, en la figura 20 se observa a mayor detalle entre la época 90 y 100 la precisión donde el valor más alto de todo el entrenamiento es de 0.8087 en la época 99 y en la figura 21 se observa a mayor detalle entre la época 110 y 119 que el valor más alto de la métrica mAP@0.5 durante todo el entrenamiento es 0.7116 en la época 114, significa que el modelo detecta el bache en un 71% con respecto al cuadro delimitador de referencia, esta grafica tiene una tendencia creciente, lo que significa que el modelo está siendo más eficaz para detectar objetos en general.

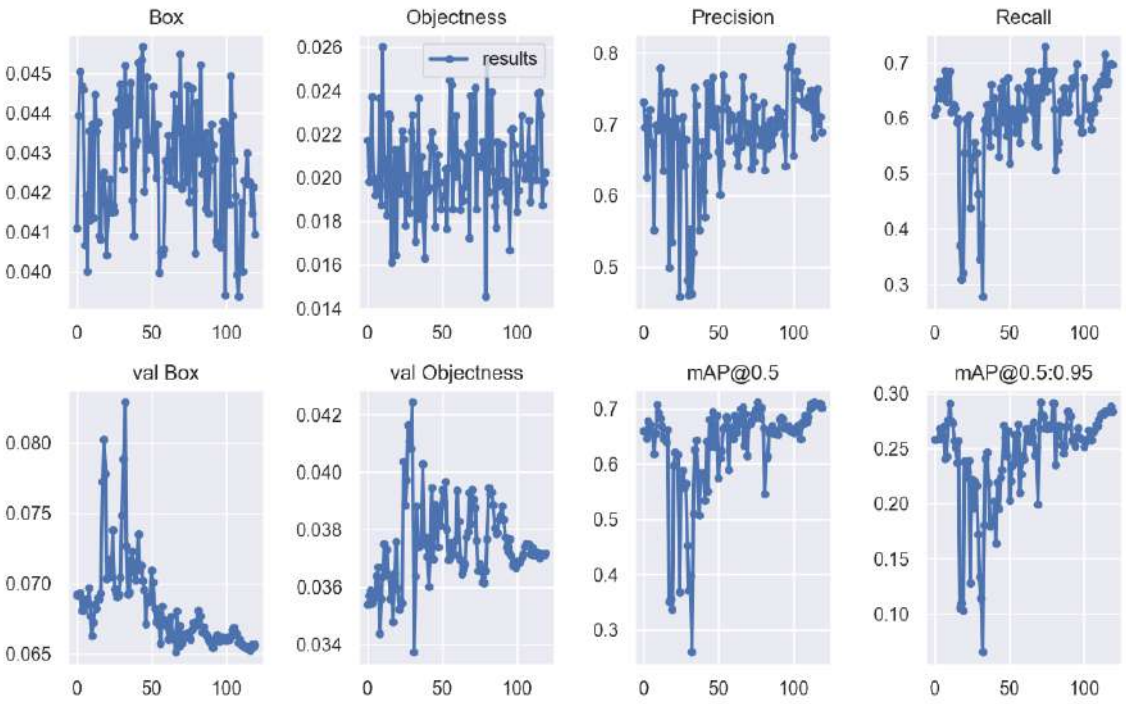


Figura 18: Resultados de YOLO V7 Tiny para 100 épocas.

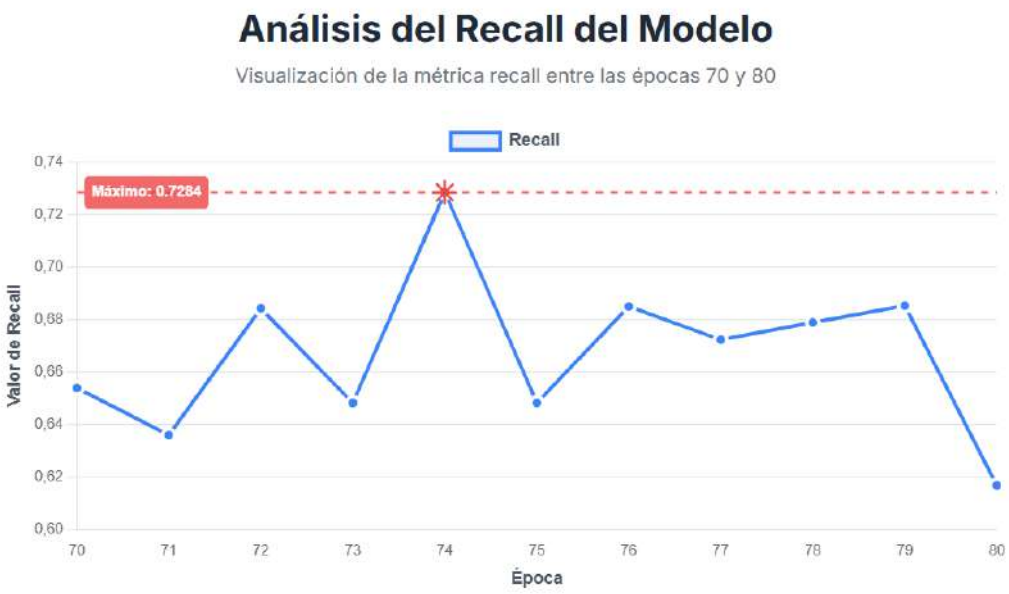


Figura 19: Valores del Recall en el segundo entrenamiento entre las épocas 70 y 80.

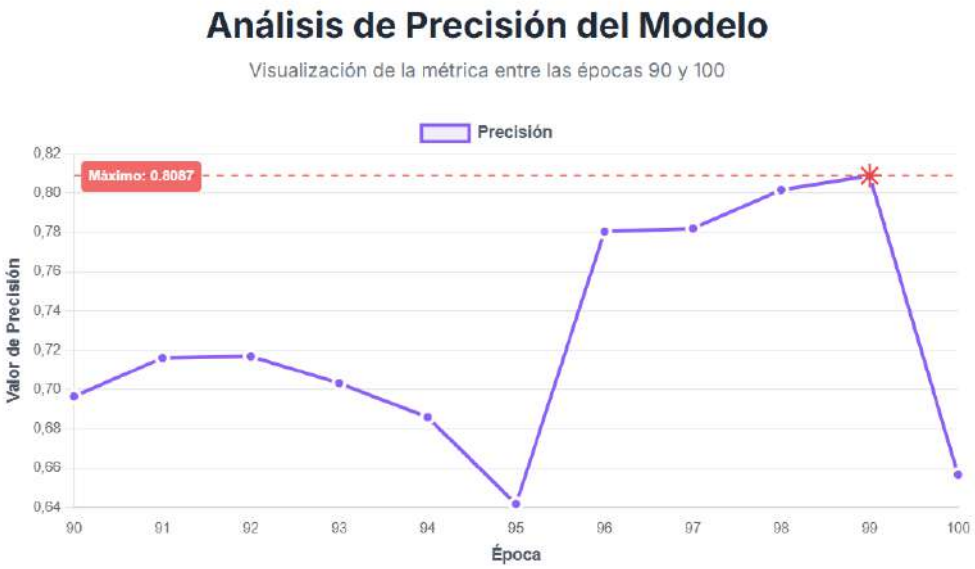


Figura 20: Valores de la precisión en el segundo entrenamiento entre las épocas 90 y 100.

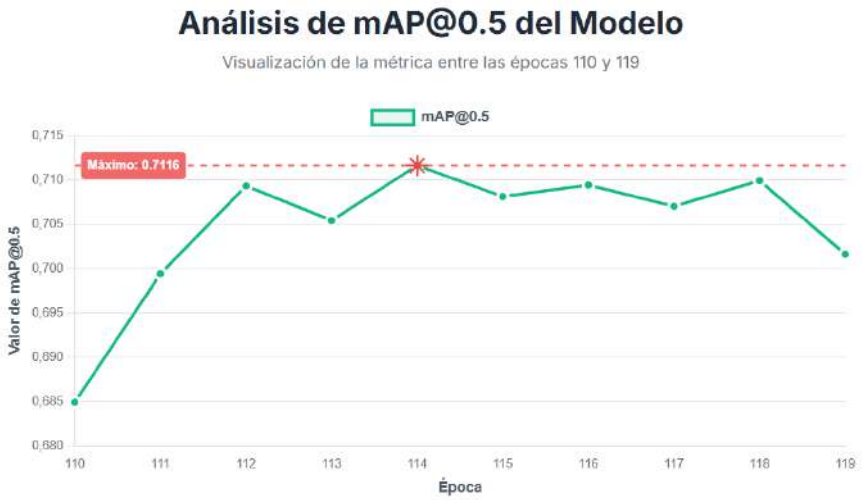


Figura 21: Valores de la métrica mAP@0.5 en el segundo entrenamiento entre las épocas 110 y 119.

En la figura 22 se observa el cambio del recall en función de la confianza, el primer gráfico corresponde al entrenamiento del primer modelo donde al inicio la precisión es de 1.0 cuando el recall es 0.0 lo que indica que el modelo hace las detecciones de los baches cuando se encuentra muy seguro de que lo que está detectando es un bache por lo que puede también estar omitiendo la detección de otros baches reales, cuando el recall se aproxima a 1.0 la precisión tiene una disminución llegando a 0.0 que indica que en esta zona el modelo está detectando objetos pero al no tener una buena precisión el modelo está dejando pasar posibles falsos positivos es decir, está

detectando baches donde realmente no los hay; el segundo grafico corresponde al segundo entrenamiento, se observa que al inicio el modelo es muy estricto con las predicciones por lo que la precisión se mantiene en el valor de 1.0, a medida que el recall avanza y es superior a 0.2 el modelo empieza a ser menos estricto y al final de la gráfica el modelo se comporta de manera permisiva detectando casi todos los baches dando espacio a un mayor número de falsos positivos.

En cuanto a los valores de la métrica Map @0.5 el primer entrenamiento obtuvo un valor de 0.746 lo que indica que durante el proceso de entrenamiento el modelo logró un buen equilibrio entre la detección correcta de baches (recall) y la precisión en sus predicciones (precisión); en el caso del segundo entrenamiento el modelo obtuvo un valor de 0,702 que sigue siendo un buen balance en cuanto a la combinación de la precisión y el recall.

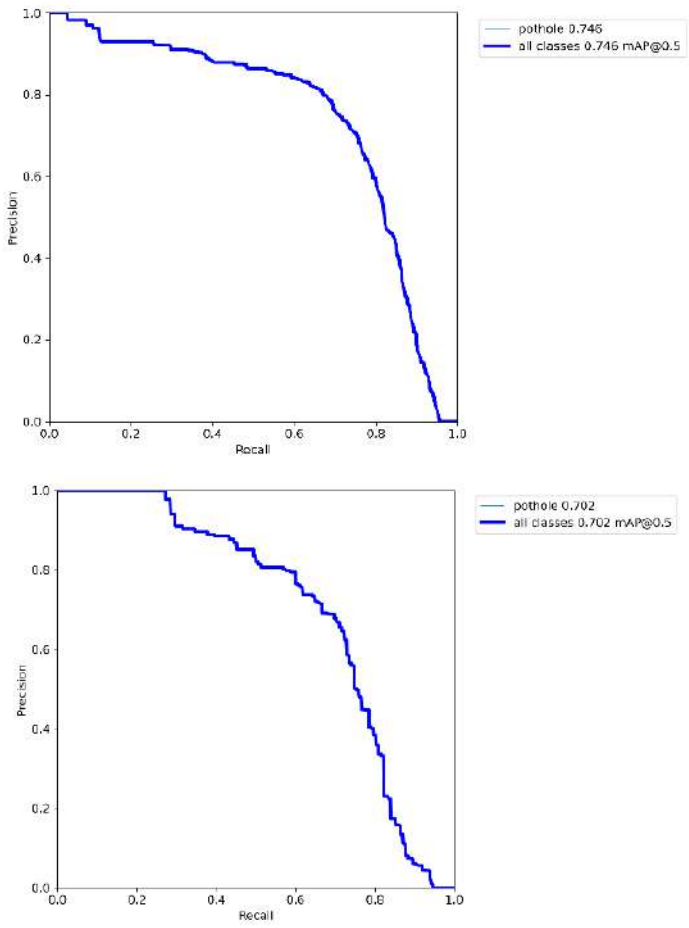


Figura 22: Gráficas obtenidas de los modelos entrenados, curva precisión-recall.

En la figura 23 se observa el comportamiento de la precisión en función de la confianza, se deduce que cuando la confianza es baja (0.0), el modelo hace muchas más predicciones lo que puede

aumentar el número de falsos positivos; en el caso contrario cuando la confianza se aproxima a 1.0 el modelo comienza a ser más selectivo con sus predicciones lo que puede aumentar la precisión, pero también se pierden verdaderos positivos.

Para el primer gráfico correspondiente al primer entrenamiento, se observa que alcanza una precisión de 1.0 cuando la confianza llega al 0.86; para el segundo gráfico que corresponde al segundo entrenamiento del modelo, la precisión alcanza un valor de 1.0 en una confianza menor a la del primer gráfico es decir cuando esta es de 0.693.

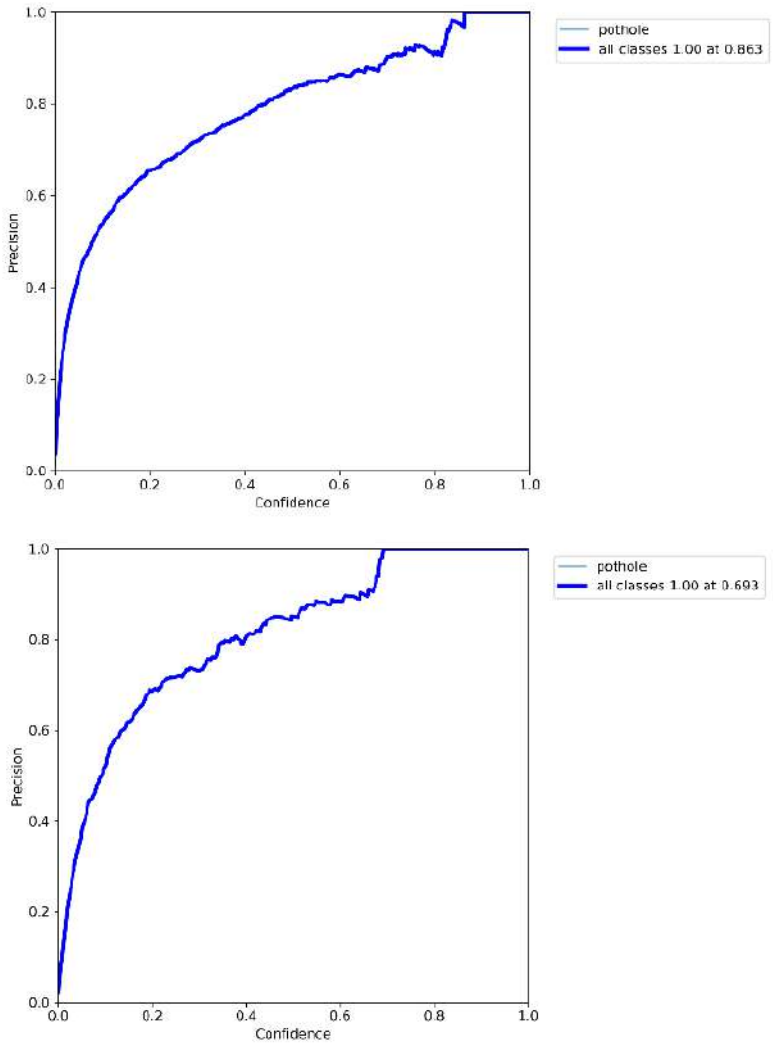


Figura 23: Gráficas obtenidas de los modelos entrenados, curva precisión-confianza.

En la figura 24 se observa el comportamiento del recall en función de la confianza donde se puede deducir que cuando hay bajos niveles de confianza, el modelo acepta más predicciones con baja probabilidad, es decir, acepta más verdaderos positivos, pero así mismo está aceptando falsos

positivos; cuando la confianza está cerca de 1.0 el modelo filtra predicciones menos probables, aumentando los falsos negativos y disminuyendo verdaderos positivos, lo que reduce el recall. En el grafico de la izquierda que corresponde al primer entrenamiento, se puede observar que el recall llega a 0 cuando la confianza es del 95%; en el caso del grafico de la derecha, el recall llega a 0 cuando la confianza es del 94%. Para ambos casos con valores cercanos, esto implica que a niveles tan altos de confianza ninguna predicción alcanzo los mismos valores, es decir no hay verdaderos positivos detectados que cumplan para el primer caso una confianza del 95% o para el segundo caso una confianza del 94%.

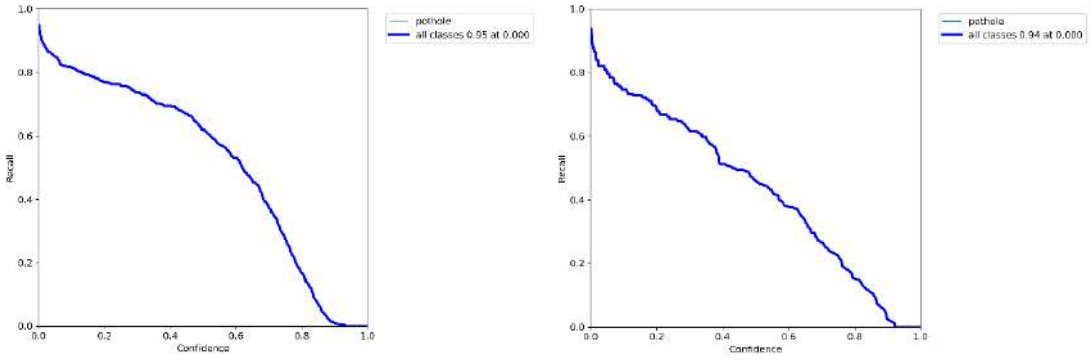


Figura 24: Gráficas obtenidas de los modelos entrenados, curva recall-confianza.

En la figura 25 se encuentran las matrices de confusión donde el grafico de la izquierda corresponde al primer entrenamiento del que se obtuvo un valor del 79% para la tasa de verdaderos positivos, 21% para la tasa de falsos negativos; para el segundo entrenamiento estos valores fueron del 66% y del 34% respectivamente; para los dos gráficos se obtuvo un valor de 1 en el background de los falsos positivos, esto se interpreta que en ambos modelos el 100% de sus falsos positivos fueron clasificados como baches ya que es la única clase existente, este valor no cuantifica la tasa de falsos positivos ya que esta debe evaluarse a través de la métrica de precisión. La comparación directa entre ambos modelos revela del primer entrenamiento una mayor capacidad para identificar correctamente los baches existentes, reduciendo los errores de omisión en 13 puntos porcentuales respecto al segundo modelo.

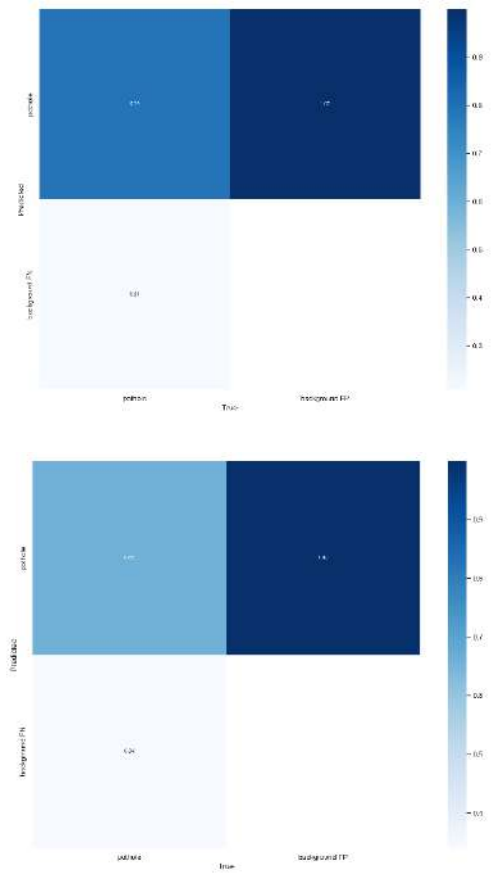


Figura 25: Gráficas obtenidas de los modelos entrenados, matriz de confusión.

En la figura 26 se observa el comportamiento de la variable F1 en función de la confianza, se quiere encontrar el valor de la confianza donde el recall y la precisión están en equilibro y en su valor más alto; para el primer entrenamiento el valor de la confianza es del 41.3% y F1 alcanza un valor del 0.74; para el segundo entrenamiento cuando la confianza es del 19.4% la variable F1 alcanza un valor del 0.69; el rendimiento general del modelo del segundo entrenamiento se redujo porque la precisión y el recall se redujeron, a pesar de eso la puntuación del F1 no tuvo una diferencia abrupta.

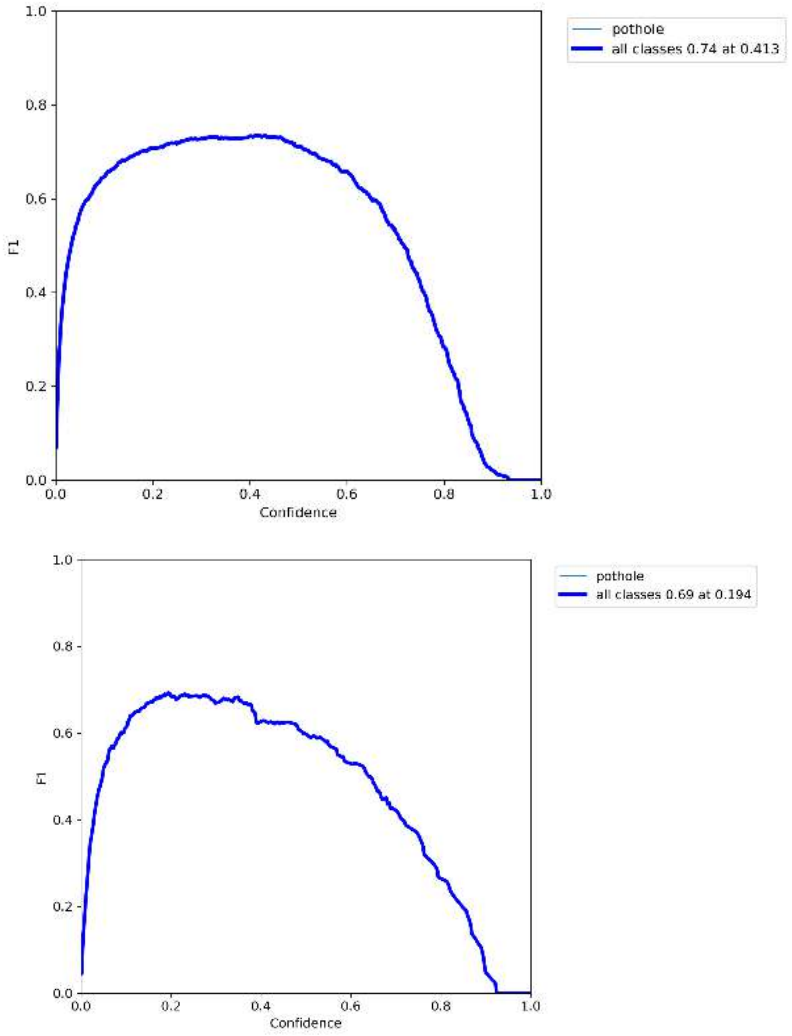


Figura 26: Gráficas obtenidas de los modelos entrenados, F1 Score.

En cuanto a los valores de las métricas para los mejores modelos del primer y segundo entrenamiento, se presenta la tabla 5 donde se observa que el primer entrenamiento obtuvo un mAP @0.5 superior al segundo entrenamiento en un 3.72%; para la precisión el primer entrenamiento fue superior en un 4.13% y para el recall, el primer entrenamiento fue superior en un 2.71%. A pesar de que del segundo entrenamiento se obtuvieron métricas con valores menores a las métricas del primero, estos valores siguen siendo competentes dentro de una buena detección de baches.

Tabla 5: Valores de las métricas obtenidas de los entrenamientos.

Entrenamiento	Métricas		
	mAP @0.5	Precision	Recall
1	74.42%	74.3%	72.41%
2	70.7%	70.17%	69.7%
Diferencia	3.72%	4.13%	2.71%

Aplicación Web: GeoBach

Las primeras páginas permiten al usuario iniciar sesión con su correo electrónico y su contraseña, en caso de que este no tenga una cuenta creada también podrá crear su cuenta para lo que, es necesario que proporcione un nombre de usuario, un correo electrónico y una contraseña.



Figura 27: Paginas de inicio de sesión y de registro.

Una vez el usuario inicie sesión con su cuenta, será redirigido a la página principal donde podrá ver sus proyectos o en caso de no tener ningún proyecto también tiene la opción de crear uno.

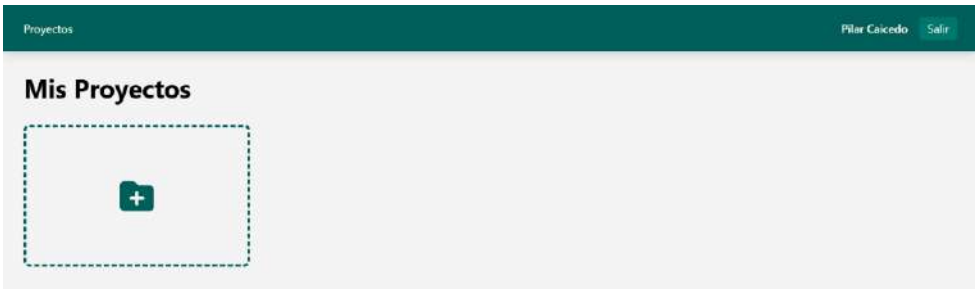


Figura 28: Pagina para la creación de proyectos.

Cuando el usuario decide crear un nuevo proyecto, se le solicitará un título y una descripción que haga referencia al proyecto que manejará como se puede observar en la figura 18.

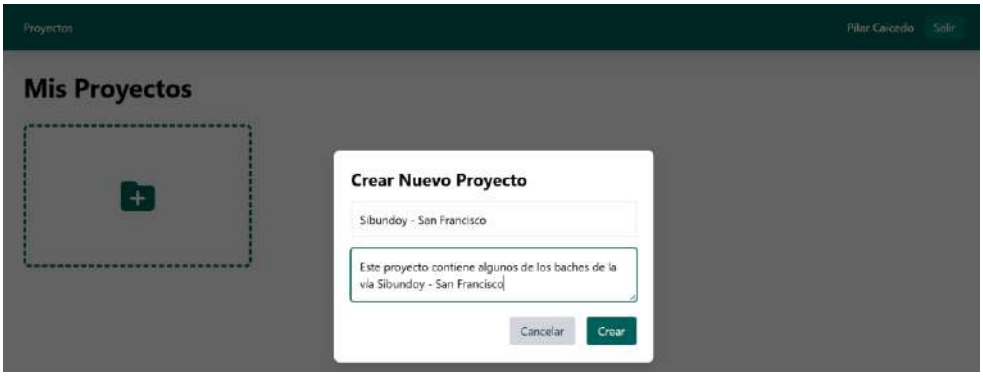


Figura 29: Creación de un proyecto en la aplicación GeoBach.

Una vez el usuario haya creado un proyecto, este aparecerá en su página principal con la fecha en que fue creado y debajo de estará la fecha de su última modificación; el proyecto estará listo para trabajar en él.



Figura 30: Proyecto creado en GeoBach.

Al entrar al proyecto se puede visualizar la información del proyecto e inicialmente la sección de resultados se encontrará vacía.

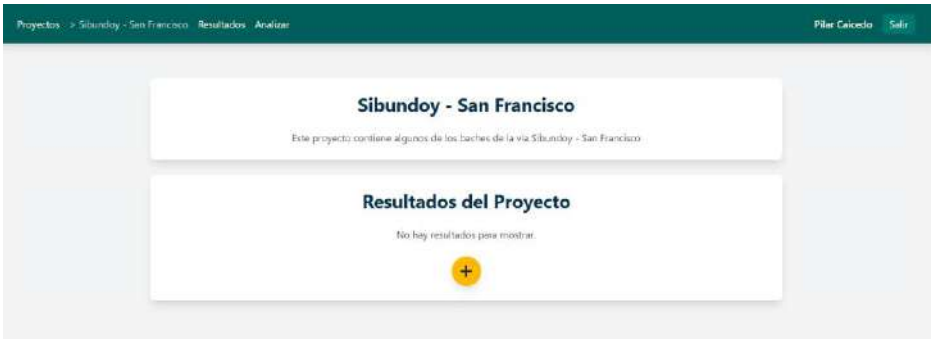


Figura 31: Datos del proyecto.

Al hacer clic en el icono + será redirigido a la página *Analizar Múltiples*, donde se puede hacer el ejercicio de la detección de baches y el proceso de medición con el uso de parámetros como el ancho del campo de visión (FOV).

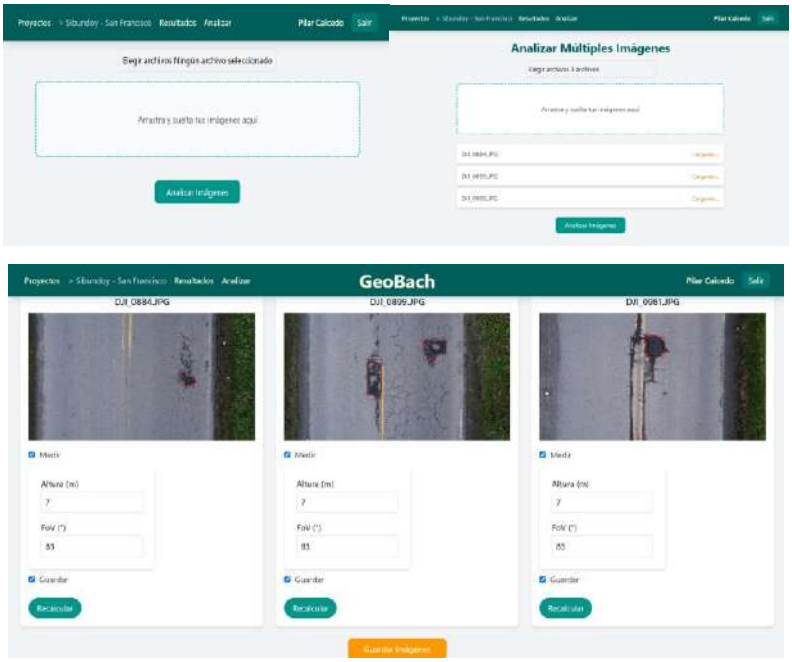


Figura 32: Proceso de carga de imágenes.

Una vez detectados los baches y con los parámetros completados, se puede observar al pasar el cursor sobre las cajas delimitadoras, las dimensiones reales de los baches como se observa en la figura 27.



Figura 33: Caja delimitadora con dimensiones reales del bache.

Dentro del proyecto se pueden visualizar los resultados del análisis que contienen una tabla con características de la imagen como sus coordenadas, parámetros registrados por el usuario, cantidad de baches detectados en la imagen y dos botones de acciones el primero que permite editar los parámetros y el segundo botón encargado de redirigir al mapa hecho con Leaflet donde se resaltará de un color distinto la imagen seleccionada; y finalmente se encuentra un diagrama de barras donde se puede observar que imagen contiene mayor número de baches.

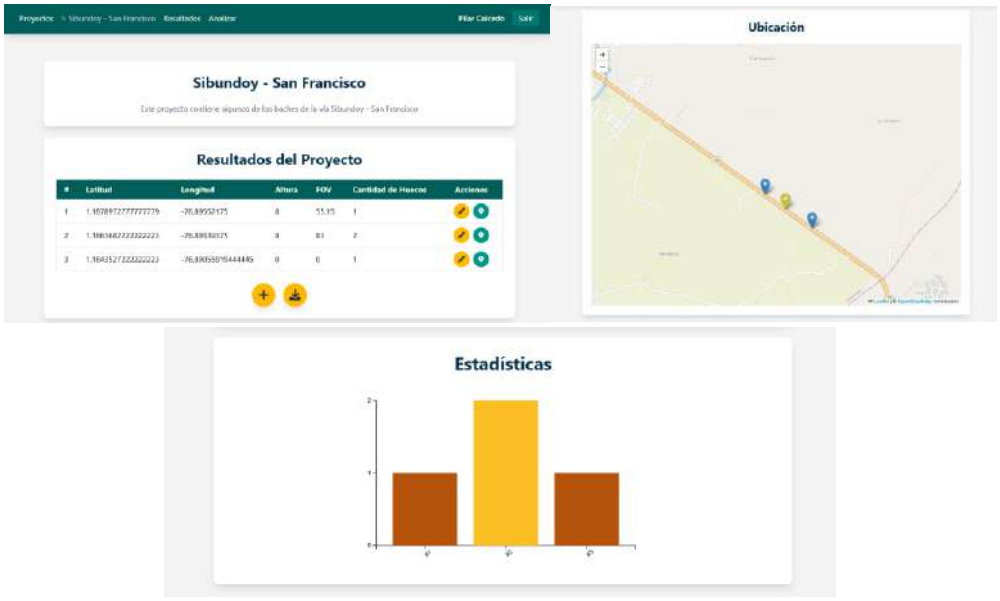


Figura 34: Resultados del proyecto que incluyen una tabla, un mapa y un gráfico de barras.

También hay que resaltar el botón de descarga que permite descargar la información de los baches e imágenes en un archivo de tipo CSV donde NA corresponde a parámetros no diligenciados o cifras no calculadas por falta de los mismos.

	A
1	id,img_lat,img_long,bache_id,fov,largo_m,ancho_m
2	101,1.1864264722222222,-76.89347355555556,158,83,0.98,0.63
3	101,1.1864264722222222,-76.89347355555556,159,83,0.3,0.41
4	102,1.1859128055555557,-76.89275347222222,160,83,0.89,0.59
5	102,1.1859128055555557,-76.89275347222222,161,83,1.12,0.65
6	103,1.1843527222222223,-76.89055919444445,NA,NA,NA,NA
7	

Figura 35: Resultados del proyecto en formato CSV.

Teniendo en cuenta que la aplicación web cuenta con la opción de obtener las medidas reales de los baches a continuación, se presentan los resultados obtenidos por la aplicación y los resultados reales tomados en campo:

La figura 36 corresponde a un bache con dimensiones de 1 x 1 metro aproximadamente y también se pueden ver las dimensiones calculadas por GeoBach donde estableció medidas de 0.94 x 1.19 metros.



Figura 36: Bache real con dimensiones de 1x1m y detección hecha por GeoBach con dimensiones de 0.94x1.19m.

La tabla 6 es una comparación de las medidas obtenidas por GeoBach con las medidas obtenidas en campo, en los anexos del 3 al 13 se encuentran los baches con sus dimensiones que hacen parte de este análisis; se realiza un cálculo del área para cada método y también se hace el cálculo de errores relativos. Como resumen de los valores obtenidos en la tabla 6 se encuentra la tabla 7 que indica los errores globales en el área, en el largo y ancho de los baches.

Tabla 6: Medidas obtenidas haciendo uso de GeoBach y medidas obtenidas en campo.

# Bache	GeoBach		Real		Diferencia Ancho	Diferencia Largo	Error Ancho	Error Largo	GeoBach Area	Real Area	Error Area
	Ancho	Largo	Ancho	Largo							
1	0,94	1,19	1	1	0,06	0,19	6%	19%	1,1186	1,19	6%
2	0,47	0,99	0,35	0,6	0,12	0,39	34%	65%	0,4653	0,3465	34%
3	0,49	0,57	0,55	0,5	0,06	0,07	11%	14%	0,2793	0,3135	11%
4	0,39	0,66	0,4	0,5	0,01	0,16	3%	32%	0,2574	0,264	3%
5	0,38	0,36	0,4	0,37	0,02	0,01	5%	3%	0,1368	0,144	5%
6	0,54	0,65	0,6	0,64	0,06	0,01	10%	2%	0,351	0,39	10%
7	0,63	0,58	0,64	0,56	0,01	0,02	2%	4%	0,3654	0,3712	2%
8	0,59	1,08	0,52	0,98	0,07	0,1	13%	10%	0,6372	0,5616	13%
9	0,39	0,49	0,4	0,6	0,01	0,11	3%	18%	0,1911	0,196	3%
10	0,33	0,33	0,33	0,22	0	0,11	0%	50%	0,1089	0,1089	0%
11	0,36	1,19	0,4	1,3	0,04	0,11	10%	8%	0,4284	0,476	10%
12	0,49	0,62	0,42	0,6	0,07	0,02	17%	3%	0,3038	0,2604	17%

Tabla 7: Errores globales

Error Area	Error Ancho	Error Largo
7.27%	7.27%	11.3%

En la figura 37 se observa el gráfico de dispersión de las medidas obtenidas para el ancho por GeoBach con las medidas reales capturadas en campo, donde se obtuvo la siguiente ecuación correspondiente a la recta:

$$y = 0.8193x + 0.1013 \quad (26)$$

El valor de 0.8193 pertenece a la variable de la pendiente (m), esto significa que por cada aumento de 1 en la medida real (x), la medida de GeoBach (y) aumenta en 0.8193 unidades; esto indica que GeoBach tiende a subestimar el ancho del bache. En cuanto al valor de $R^2 = 0.872$ se puede decir que el 87.2% de la variabilidad de las mediciones se debe a las variaciones en las mediciones reales, el 12.8% restante implica que existen otros factores que no se justifican con la relación lineal entre GeoBach y las medidas reales como puede ser la delimitación de la caja que identifica el bache.

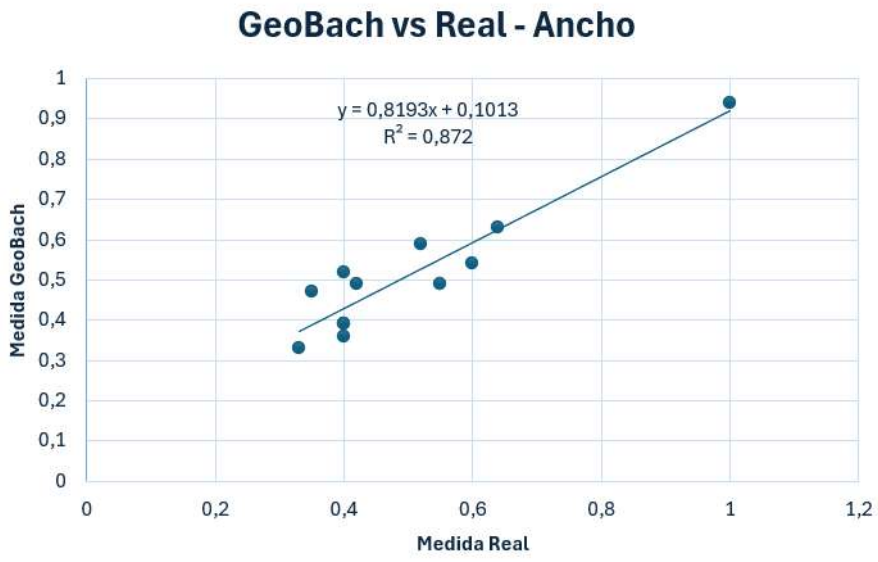


Figura 37: Grafico de dispersión medidas de GeoBach con medidas reales (Ancho).

En la figura 38 se observa el grafico de dispersión de las medidas obtenidas para el ancho por GeoBach con las medidas reales capturadas en campo, donde se obtuvo la siguiente ecuación correspondiente a la recta:

$$y = 0.9255x + 0.1188 \tag{27}$$

El valor de 0.9255 pertenece a la variable de la pendiente (m), esto significa que por cada aumento de 1 en la medida real (x), la medida de GeoBach (y) aumenta en 0.9255 unidades; esto indica que GeoBach tiende a subestimar el largo del bache. En cuanto al valor de $R^2 = 0.8058$ se puede decir que el 80.58% de la variabilidad de las mediciones se debe a las variaciones en las mediciones reales, el 19.42% restante implica que existen otros factores que no se justifican con la relación lineal entre GeoBach y las medidas reales como puede ser la delimitación de la caja que identifica el bache.

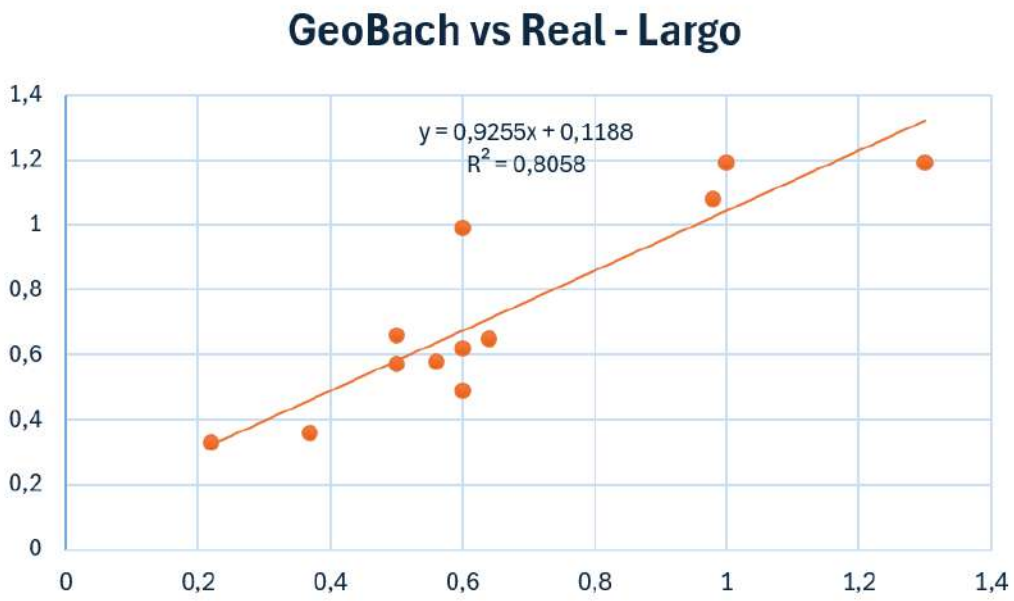


Figura 38: Grafico de dispersión medidas de GeoBach con medidas reales (Largo).

DISCUSIÓN

La aplicación de la arquitectura YOLO ha demostrado ser eficaz para la detección de baches. Chitale, et al., (2020) afirman que a diferencia de arquitecturas de dos etapas como Faster R-CNN, YOLO no requiere un paso separado de propuesta de región, lo que elimina la complejidad de la red, al hacer uso de las versiones 3 y 4 obtuvieron un mAP @0.5 del 88.9% y 93.3% respectivamente. Por su parte, Chavan, et al., (2022) al usar la versión 4, consideran que la principal ventaja es su velocidad superior que es capaz de procesar hasta 45 cuadros por segundo, en su análisis comparativo YOLO V4 superó a los modelos basados en CNN, alcanzando una precisión promedio de 85% - 90% para la detección de baches en tiempo real, mientras que los modelos CNN, aunque lograban una precisión de 80-82%, tardaban mucho más tiempo en detectar el bache; Aware, et al. (2023) propusieron un sistema automatizado para la detección y estimación del área de baches utilizando Mask R-CNN, donde obtuvieron un mAP@0.5 del 48%, este valor es 22 puntos porcentuales menor al mAP@0.5 del 70.7% que se obtuvo con el uso de la arquitectura YOLO en el desarrollo de este trabajo; esta brecha se puede atribuir directamente a la elección de la arquitectura del modelo, la evolución de la arquitectura YOLO ha demostrado superar a otros modelos, incluso a modelos anteriores de sí mismo en términos de precisión y eficiencia (Wang, Bochkovskiy y Liao, 2023).

El uso de modelos tiny de la arquitectura YOLO son una alternativa viable para reducir el costo computacional y obtener buenos resultados. Dharneeshkar, et al., (2020) trabajaron con las versiones 2, 3 y 3 tiny de YOLO, de acuerdo con las métricas resultantes del entrenamiento el mejor modelo fue el de la versión 3 tiny que obtuvo un mAP @ 0.5 del 49.51% y una precisión del 76%, lo que sugiere que, a pesar de ser un modelo con menor complejidad, logró un rendimiento mayor al de las versiones 2 y 3 evaluadas en este documento. Para este trabajo se usó la versión de YOLO V7 tiny debido a que la versión 7 requería mayor costo computacional del que se tenía disponible; la versión tiny se alineó de mejor manera a la técnica de entrenamiento multi-escala utilizada; al comparar los resultados de este estudio que obtuvo un mAP@0.5 del 70.7%, una precisión del 70.17% y un recall del 69.7% con estudios como el de Dharneeshkar, et al., (2020) que obtuvieron una precisión ligeramente superior, se infiere que pudo ser causado por la diferencia de dataset entre los dos estudios ya que Dharneeshkar, et al., (2020) usaron un total de 1500 imágenes y en este estudio se usaron 1058; aun así el modelo propuesto en este trabajo fue superior al de Dharneeshkar, et al., (2020) en otras métricas como el mAP@ 0.5 por 21 puntos porcentuales y en el recall por 29.7 puntos porcentuales.

La forma irregular y la variación del tamaño de baches afectaron la precisión en su identificación. Zhong, et al., (2025) afirman que la presión prolongada de los vehículos dificulta la localización de los límites de los baches lo que genera mayores errores en la detección, que la precisión de la detección se ve influenciada por la claridad del límite del bache y la rugosidad de la carretera; por ejemplo, los baches con contorno de límites ambiguos mostraron un error de superficie del 5.84%, los baches ubicados en áreas con mayor rugosidad de la carretera y perfiles de limite menos distintos tuvieron errores de superficie del 13% y de perímetro alrededor del 11%. Baek & Chung, (2020) afirman que los baches se caracterizan por sus diversas formas y tamaños, que el contorno de los baches puede ser difícil de encontrar debido a la diferencia de contraste en ambientes nocturnos y lluviosos; identificaron que los baches con muchas grietas tenían una precisión baja, que en algunos casos fueron del 60.5% y 58.9%; además afirman que otra de las dificultades para la detección de baches es que estos no cuentan con un color RGB fijo (Baek and Chung, 2020). Para el presente trabajo a pesar del entrenamiento multi-escala, la forma irregular de los baches presentó un desafío en la estimación precisa de sus dimensiones ya que las cajas delimitadoras rectangulares generadas por el modelo, al intentar abarcar baches con contornos irregulares, en

ocasiones incluían áreas que no correspondían al bache real, llevando a mediciones subestimadas lo que generó un error global en el área del 7.27%.

El aprendizaje por transferencia como metodología para el entrenamiento de un modelo resulta efectivo. Silva, et al., (2020) basaron su trabajo en el modelo YOLOv4-tiny con pesos pre-entrenados, alcanzaron una precisión del 96.25% y un promedio de precisión (AP) del 98.46% para la clasificación de baches. Gujar, et al., (2024) al utilizar un modelo pre-entrenado de YOLO V8, obtuvieron un mAP del 95.82% y una precisión de 95%. El producto final de este trabajo denominado GeoBach, al aplicar un proceso de doble entrenamiento, obtuvo un mAP@0.5 del 70.7%, una precisión del 70.17% y un recall del 69.7%. Dharneeshkar, et al., (2020) lograron un mAP@0.5 de 49.71%, una precisión de 76% y un recall de 40% con Tiny YOLOv3; esto significa que GeoBach superó a Dharneeshkar et al. en mAP@0.5 por 20.99 puntos porcentuales y en recall por 29.7 puntos porcentuales. Aware, et al., (2023) obtuvieron un mAP@0.5 del 48%, lo cual es considerablemente inferior al rendimiento de GeoBach.

CONCLUSIONES

En el desarrollo de este proyecto se obtuvo como resultado a GeoBach, una aplicación diseñada como solución para la inspección y cuantificación automatizada de baches en vías de tipo asfaltico haciendo uso de técnicas de inteligencia artificial; esta aplicación permite que los usuarios puedan gestionar proyectos que contengan imágenes aéreas capturadas por VANT´S de una forma más segura con una menor exposición en las vías, dichas imágenes son procesadas por un modelo de YOLOV7 Tiny que cuenta con la capacidad de identificar y cuantificar los baches presentes. Mediante el uso de parámetros como el campo de visión (FOV), GeoBach es capaz de calcular las dimensiones de los baches lo que facilita obtener un análisis más detallado. Dentro de cada proyecto, los resultados se presentan en tablas que contienen la cantidad de baches por imagen con sus respectivas coordenadas y parámetros registrados por el usuario; también, cuenta con un componente geoespacial que es un mapa interactivo donde se visualiza la ubicación georreferenciada de las imágenes procesadas; si el usuario tiene la necesidad de descargar la información, también es posible por medio de un botón que exporta dicha información en un archivo de tipo CSV. Teniendo en cuenta lo anterior, GeoBach se define como una herramienta robusta para la toma de decisiones relacionadas con procesos de inspección y mantenimiento vial.

A pesar de las limitaciones computacionales y la carencia de un dataset robusto, los resultados obtenidos en este estudio demuestran ser altamente competitivos. En métricas clave como el mAP@0.5, este modelo no solo se alinea con el estado del arte, sino que supera significativamente los resultados reportados por trabajos como los de Dharneeshkar, et al., (2020), Suong, et al., (2018) y Aware, et al., (2023) que usaron modelos como mask R-CNN, YOLOV3, YOLOV3-Tiny, YOLOV2 y YOLOV2 Tiny; lo que valida el potencial de esta metodología para aplicarse en la detección de baches en pavimento asfaltico; En cuanto a las medidas reales y a las medidas calculadas por GeoBach se puede deducir que las medidas del ancho son muy cercanas a las reales con un error en la media del 7.27%, para las medidas del largo se presentan algunas sobrestimaciones por lo que su error en la media es de 11.3% que son causa en su mayoría por las delimitaciones de la caja de detección.

FUTUROS TRABAJOS

Con el fin de complementar este proyecto, se plantean para trabajos futuros los siguientes puntos:

Detección de diversos tipos de afectaciones en pavimento flexible y rígido: El pavimento de tipo flexible puede presentar otras afectaciones como fisuras dentro de las que existen de diferentes tipos, también deformaciones y perdidas de capas estructurales; así mismo, el pavimento de tipo rígido puede presentar afectaciones como las grietas, fisuras y deterioro superficial, por lo que con un dataset más variado se podría cambiar el enfoque de la aplicación a uno más amplio donde se pueda realizar la detección de diversos tipos de afectaciones en los dos tipos de pavimento más frecuentes.

Permitir la medición de la profundidad de baches: Teniendo en cuenta que en la actualidad existen drones equipados con tecnología LiDAR, resulta importante que la aplicación pueda trabajar con este tipo de datos para obtener información sobre la profundidad de los baches con lo que se podría determinar hasta que capa del pavimento se encuentra la afectación.

Comparación de rutas similares: Se propone la comparación de mismos tramos de vías en fechas distintas para obtener la diferencia entre las épocas y determinar si dicho tramo ha mejorado o empeorado.

Actualización y migración hacia versiones más recientes de Yolo: Dado el avance constante de las arquitecturas de detección de objetos, un trabajo futuro de alta relevancia consiste en actualizar el modelo implementado hacia versiones más recientes de la familia Yolo o variantes optimizadas para entornos ligeros (Nano, Small o Mobile). Estas versiones incorporan mejoras en precisión, estabilidad del entrenamiento, capacidad de generalización y eficiencia computacional que podrían superar los resultados obtenidos con YOLOv7-Tiny.

Implementar la aplicación para dispositivos móviles: Dado que esta aplicación fue implementada como una aplicación web, también sería importante desarrollarla como aplicación móvil para tabletas o celulares con el fin de que se puedan usar en campo de manera más práctica.

Referencias

Alshehhi, Rasha, Prashanth Reddy, Wei Lee, and Mauro Dalla. "Simultaneous extraction of roads and buildings in remote sensing imagery with convolutional neural networks." *ISPRS Journal of Photogrammetry and Remote Sensing*, no. 130 (2017): 139-149.

Autofact. *Huecos en las calles ¿Qué hacer?* Autofact. n.d. https://www.autofact.com.co/blog/mi-carro/conduccion/hoyos-en-las-calles.

Aware, Sakshi, Tejal Deore, Sakshi Khatale, Kshitija Malode, and Chaudhari. "Pothole Detection and Area Estimation System Using Mask R-CNN." *International Journal of Creative Reseach Thoughts* XI (2023): 262-266.

Baek, Ji-Won, and Kyungyong Chung. "Pothole classification model using edge detection in road image." *Applied Sciences* X, no. 19 (2020): 6662.

Barris, Wes. *Converting Diagonal Field of View (FOV) to Horizontal FOV.* n.d. https://www.litchiutilities.com/docs/fov.php?utm_source=chatgpt.com.

Callari, Francesco. *Relation between horizontal, vertical and diagonal Field-of-View.* Julio 7, 2015. https://stackoverflow.com/questions/31257664/relation-between-horizontal-vertical-and-diagonal-field-of-view/31258926?utm_source=chatgpt.com.

Chavan, Kshitija, Chinmay Chawathe, Vatsal Dhabalia, and Sankhe Amruta. "POTHOLE DETECTION SYSTEM USING YOLO v4 ALGORITHM ." *International Research Journal of Engineering and Technology (IRJET)* , 2022: 2777-2782.

Chitale, Pranjal, Hrishikesh Shenai, Gala, Jay, Kaustubh Kekre, and Ruhina Karani. "Pothole Detection and Dimension Estimation System Using Deep Learning (YOLO) and Image Processing." *IEEE*, 2020: 01-06.

Cortés, Erika. "Levantamientos topográficos mediante el uso de drones ." (Ingeniería: Ciencia Tecnologia e Innovación) VIII, no. 2 (2021).

Daley, Sam. *What is a Drone? Uses of Drones and Definition.* Builtin. Marzo 23, 2023. https://web.archive.org/web/20230603173852/https://builtin.com/drones.

Delgado, Efredy. "Comparación de clasificadores de género: Redes Neuronales MLP y Teoría de la Información –ITL–." *AVANCES Investigación en Ingeniería*, no. 13 (2010): 87-97.

Dharneeshkar, J, S. Aniruthan, and R Karthika. "Deep Learning based Detection of potholes in Indian roads using YOLO." *En 2020 international conference on inventive computation technologies (ICICT).* 2020.

Fonter. *El estado de las vías en Colombia: ¿Cómo nos comparamos con Sudamérica?* Fonter. 2023. https://fonter.co/el-estado-de-las-vias-en-colombia-como-nos-comparamos-con-sudamerica/ (accessed Enero 27, 2024).

Garzón, Allison, and Leicy Hernández. "Cartilla-Guia ilustrativa del proceso constructivo de un pavimento flexible para bajos volumenes de tránsito." 2018.

Gil, Juan Pablo Latorre. *¿Quién responde cuando su vehículo se daña por un hueco o el mal estado de las vías?* RCNRadio. Junio 24, 2023. https://www.rcnradio.com/colombia/quien-responde-cuando-su-vehiculo-se-dana-por-un-hueco-o-el-mal-estado-de-las-vias.

Gujar, S, Prathmesh Shinde, Shlok Parihar, Aditya Pawar, and Saikrishna Yemul. "POTHOLE DETECTION SYSTEM USING YOLO V8 ." *International Research Journal of Modernization in Engineering Technology and Science* , 2024: 9715-9719.

HemaMalini, B. H., A. Padesur, V. Manoj, and A Shet. "Detection of Potholes on Roads using a Drone." *EAI Endorsed Transactions on Energy Web*, 2021.

Henderson, P, and V Ferrari. "End-to-end training of object class detectors for mean average precision." *Computer Vision–ACCV 2016: 13th Asian Conference on Computer Vision.* Taipei: Springer International Publishing, 2017. 198-213.

Hossain, Ali, Ahmed Boshir, and Ariful Islam. "Automatic Extractions of Road Intersections from Satellite Imagery in Urban Areas." *6th International Confference on Electrical and Computer Engineering.* Dhaka, 2010.

Infobae. *El ocaso de la Ruta del Sol: colombianos se quejan en redes sociales del mal estado de la vía.* Infobae. Enero 7, 2023. https://www.infobae.com/america/colombia/2023/01/07/el-ocaso-de-la-ruta-del-sol-colombianos-se-quejan-en-redes-sociales-del-mal-estado-de-la-via/.

Instituto Nacional de Vías. *Administradores viales.* Enero 25, 2013. https://www.invias.gov.co/index.php/informacion-institucional/123-seguimiento-a-la-inversion/proyectos-invias/80-administradores-viales (accessed Octubre 12, 2022).

Instituto Nacional De Vias. "Estado del la red vial." *Invias.* Agosto 08, 2022. https://www.invias.gov.co/index.php/informacion-institucional/2-principal/57-estado-de-la-red-vial.

Instituto Nacional de Vias; Universidad Nacional de Colombia. "Manual para la Inspeccion Visual de Pavimentos Flexibles." Bogotá, 2006.

KANG, Byeong-ho, and Su-il CHOI. "Pothole detection system using 2D LiDAR and camera." *Ninth international conference on ubiquitous and future networks (ICUFN).* 2017. 744-746.

Koch, Christian, and Ioannis Brilakis. "Pothole detection in asphalt pavement images." *Advanced Engineering Informatics* xxv, no. 3 (2011): 507-515.

M, Garay, et al. "Detección inteligente de baches y estimación de su profundidad con Aprendizaje Profundo para la conservación de carreteras." *CIENCIA ergo-sum*, 2025: 20.

Macea, Luis, Luis Morales, and Luis Márquez. "Un sistema de gestión de pavimentos basado en nuevas tecnologías para paises en vía de desarrollo." (Ingeniería, investigación y tecnología) XVII, no. 2 (2016).

Medium. *Key Components of Neural Network Architecture.* Medium. Noviembre 13, 2023. https://baotramduong.medium.com/neural-network-architecture-08479e5cc2d8.

Minaee, Shervin, Yuri Boykov, Fatih Porikli, Antonio Plaza, Nasser Kehtarnavaz, and Demetri Terzopoulos. "Image Segmentation Using Deep Learning: A Survey." *IEEE transactions on pattern analysis and machine intelligence* XLIV, no. 7 (2021): 3523-3542.

Ministerio De Transporte Instituto Nacional De Vias. "Estudio E Investigacion Del Estado Actual De Las Obras De La Red Nacional De Carreteras." Bogotá, 2006.

O´Shea, Keiron, and Ryan Nash. "An Introduction to Convolutional Neural Networks." *Cornell University*, 2015: 1-11.

Potrimba, Petru. *What is Mask R-CNN? The Ultimate Guide.* Roboflow Blog. Agosto 09, 2023. https://blog.roboflow.com/mask-rcnn/#:~:text=Mask%20R%2DCNN%2C%20short%20for,segmentation%20tasks%20in%20computer%20vision. (accessed Abril 02, 2024).

Pragati, Baheti. *Activation Functions in Neural Networks [12 Types & Uses Cases].* v7labs. Mayo 27, 2021. https://www.v7labs.com/blog/neural-networks-activation-functions (accessed Marzo 13, 2024).

Pucha, Pablo, and Belizario Zárate. "Evaluación superficial de pavimentos rígidos en carreteras mediante ortoimágenes obtenidas mediante un vehículo aéreo no tripulado." (AVANCES: Investigación en ingeniería) XVII, no. 2 (2020).

Redmon, Joseph, and Ali Farhadi. "YOLO9000: Better, Faster, Stronger." *IEEE*, 2017: 6517-6525.

Redmon, Joseph, and Ali Farhadi. "Yolov3: An incremental improvement." *arXiv*, 2018.

Redmon, Joseph, Santosh Divvala, Ross Girshick, and Ali Farhadi. "You Only Look Once: Unified, Real-Time Object Detection." *IEEE*, 2016.

RMS. *¿Cómo ayuda la tecnología LiDAR a la topografía?* Abril 29, 2021. https://rmsgeoespacial.com/como-ayuda-la-tecnologia-lidar-a-la-topografia/#:~:text=La%20tecnolog%C3%ADa%20LiDAR%20se%20incorpora,las%20caracter%C3%ADsticas%20de%20dicha%20superficie. (accessed Octubre 11, 2022).

Saha, Sumit. *A Comprehensive Guide to Convolutional Neural Networks.* Medium. Diciembre 15, 2018. https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53.

Shah, Deval. *Intersection over Union (IoU): Definition, Calculation, Code.* v7 Labs. Mayo 30, 2023. https://www.v7labs.com/blog/intersection-over-union-guide (accessed Mayo 10, 2024).

Sharma, Rajan. *Yolo (You Only Look Once).* Medium. Marzo 21, 2020. https://rajansharma9467.medium.com/yolo-you-only-look-once-2afdba1f6a32 (accessed Abril 18, 2024).

Silva, L, H Sanchez San Blas, Peral García, D, A. Sales Mendes, and G Villarubia González. "An architectural multi-agent system for a pavement monitoring system with pothole recognition in UAV images." *Sensors* XX, no. 21 (2020): 6205.

Solawetz, Jacob. *What is Mean Average Precision (mAP) in Object Detection?* Roboflow. Mayo 6, 2020. https://blog.roboflow.com/mean-average-precision/ (accessed Mayo 10, 2024).

Steward, Matthew. *Advanced Topics in Neural Networks.* MEDIUM. Agosto 20, 2019. https://towardsdatascience.com/advanced-topics-in-neural-networks-f27fbcc638ae.

Suong, Lim Kuoy, and Kwon Jangwoo. "Detection of potholes using a deep convolutional neural network." *Journal of universal computer science* XXIV, no. 9 (2018): 1244-1257.

The VRguy's Blog. *Converting diagonal field of view and aspect ratio to horizontal and vertical field of view.* Blogspot. Abril 23, 2013. https://vrguy.blogspot.com/2013/04/converting-diagonal-field-of-view-and.html?utm_source=chatgpt.com.

Thenoux, Guillermo, and Rodrigo Gaete. "Evaluación técnica del pavimento y comparación de métodos de diseño de capas de refuerzo asfáltico." 1995.

Ventura, Daniele, Michele Bruno, Giovanna Jona, and Andrea Belluscio. "A low-cost drone based application for identifying and mapping of coastal fish nursery grounds." *Estuarine, Coastal and Shelf Science* CLXXI (2016): 85-98.

Volkswagen. *¿Baches en las calles? Evítalos con estos consejos de seguridad.* Octubre 20, 2021. https://www.vw.com.mx/es/experiencia/tips/como-cuidar-autos-de-baches-en-calles.html (accessed Diciembre 12, 2022).

Wang, Chien-Yao, Alexey Bochkovskiy, y Hong-Yuan Mark Liao. «YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors.» *In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition.* 2023.

Wikimedia Foundation Inc. *Angle of view (photography).* Wikipedia. Febrero 5, 2025. https://en.wikipedia.org/wiki/Angle_of_view_%28photography%29?utm_source=chatgpt.com.

Yurdakul, Mustafa, and Sakir Tasdemir. "An Enhanched YOLOv8 Model for Real-Time and Accurate Pothole Detection and Measurement." *arXiv preprint arXiv:2505*, 2025.

Zhong, Junkui, Deyi Kong, Yuliang Wei, and Bin Pan. "YOLOv8 and point cloud fusion for enhanced road pothole detection and quantification." *Scientific Reports*, 2025: 11260.

Zonalogistica. *Atraso vial le cuesta unos $5,6 billones al sector transporte.* Septiembre 14, 2018. https://zonalogistica.com/atraso-vial-le-cuesta-unos-56-billones-al-sector-transporte/.

Anexos

Anexo 1: Comando utilizado para el primer entrenamiento con el modelo YoloV7 Tiny.

```
python train.py --workers 4 --device 0 --weights None --cfg
cfg/training/yolov7-tiny.yaml --data data/coco.yaml --epochs 300
--batch-size 4 --hyp data/hyp.scratch.tiny.yaml --name yolov7-
tiny-custom --sync-bn --multi-scale
```

Anexo 2: Comando utilizado para el segundo entrenamiento con el modelo YoloV7 Tiny.

```
python train.py --batch-size 16 --img 640 640 --epochs 100 --data
coco.yaml --cfg cfg/training/yolov7-tiny.yaml --weights
runs/train/yolov7-tiny-custom5/weights/best.pt --hyp
data/hyp.scratch.tiny.yaml -freeze 10 --multiscale --name yolov7-
tiny-finetuned
```

Anexo 3: Bache del área de estudio con dimensiones reales de 0.35 x 0.6.

Bache con dimensiones de 0.35 x 0.6 metros aproximadamente y también se pueden ver las dimensiones calculadas por GeoBach donde estableció medidas de 0.47 x 0.99 metros.



Figura 39: Bache real con dimensiones de 0.35 x 0.6m y detección hecha por GeoBach con dimensiones de 0.47x0.99m.

Anexo 4: Bache con dimensiones reales de 0.55 x 0.5

Bache con dimensiones de 0.55 x 0.5 metros aproximadamente y también se pueden ver las dimensiones calculadas por GeoBach donde estableció medidas de 0.49 x 0.57 metros.



Figura 40: Bache real con dimensiones de 0.55x0.5m y detección hecha por GeoBach con dimensiones de 0.49x0.57m.

Anexo 5: Bache con dimensiones de 0.4 x 0.5.

Bache con dimensiones de 0.4 x 0.5 metros aproximadamente y también se pueden ver las dimensiones calculadas por GeoBach donde estableció medidas de 0.39 x 0.66 metros.



Figura 41: Bache real con dimensiones de 0.4x0.5 y detección hecha por GeoBach con dimensiones de 0.39x0.66m.

Anexo 6: Bache con dimensiones de 0.4 x 0.37.

Bache con dimensiones de 0.4 x 0.37 metros aproximadamente y también se pueden ver las dimensiones calculadas por GeoBach donde estableció medidas de 0.38 x 0.36 metros.



Figura 42: Bache real con dimensiones de 0.4x0.37 y detección hecha por GeoBach con dimensiones de 0.38x0.36m.

Anexo 7: Bache con dimensiones de 0.6 x 0.64.

Bache con dimensiones de 0.6 x 0.64 metros aproximadamente y también se pueden ver las dimensiones calculadas por GeoBach donde estableció medidas de 0.54 x 0.65 metros.

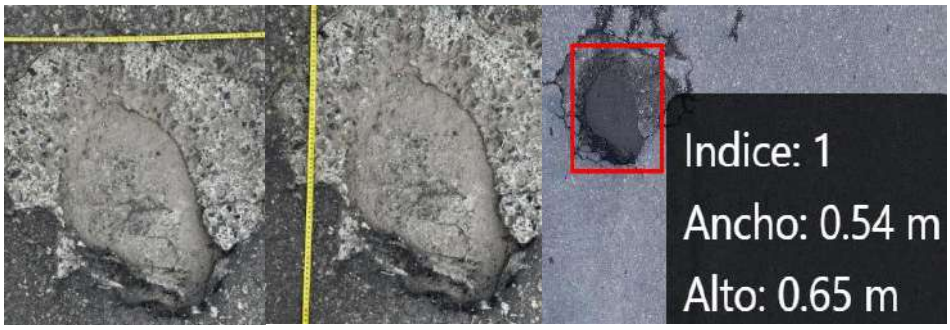


Figura 43: Bache real con dimensiones de 0.6x0.64m y detección hecha por GeoBach con dimensiones de 0.54x0.65m.

Anexo 8: Bache con dimensiones de 0.64 x 0.56.

Bache con dimensiones de 0.64 x 0.56 metros aproximadamente y también se pueden ver las dimensiones calculadas por GeoBach donde estableció medidas de 0.63 x 0.58 metros.



Figura 44: Bache real con dimensiones de 0.64x0.56m y detección hecha por GeoBach con dimensiones de 0.63x0.58m.

Anexo 9: Bache con dimensiones de 0.52 x 0.98.

Bache con dimensiones de 0.52 x 0.98 metros aproximadamente y también se pueden ver las dimensiones calculadas por GeoBach donde estableció medidas de 0.59 x 1.08 metros.

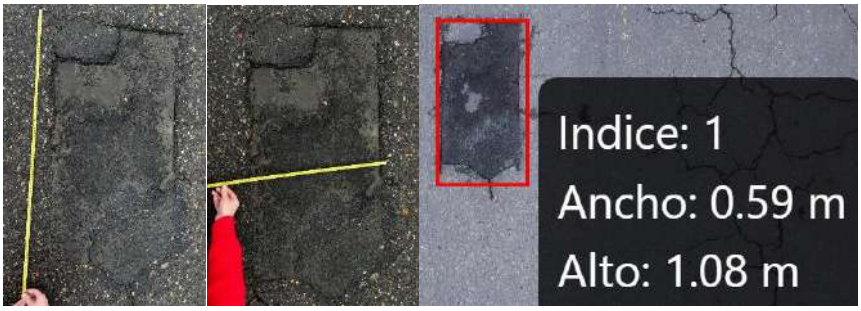


Figura 45: Bache real con dimensiones de 0.52x0.98m y detección hecha por GeoBach con dimensiones de 0.59x1.08m.

Anexo 10: Bache con dimensiones de 0.4 x 0.6

Bache con dimensiones de 0.4 x 0.6 metros aproximadamente y también se pueden ver las dimensiones calculadas por GeoBach donde estableció medidas de 0.39 x 0.49 metros.



Figura 46: Bache real con dimensiones de 0.4x0.6m y detección hecha por GeoBach con dimensiones de 0.39x0.49m.

Anexo 11: Bache con dimensiones de 0.33 x 0.22.

Bache con dimensiones de 0.33 x 0.22 metros aproximadamente y también se pueden ver las dimensiones calculadas por GeoBach donde estableció medidas de 0.33 x 0.33 metros.



Figura 47: Bache real con dimensiones de 0.33x0.22m y detección hecha por GeoBach con dimensiones de 0.33x0.33m.

Anexo 12: Bache con dimensiones de 0.4 x 1.33.

Bache con dimensiones de 0.4 x 1.33 metros aproximadamente y también se pueden ver las dimensiones calculadas por GeoBach donde estableció medidas de 0.36 x 1.19 metros.



Figura 48: Bache real con dimensiones de 0.4x1.33m y detección hecha por GeoBach con dimensiones de 0.36x1.19m

Anexo 13: Bache con dimensiones de 0.42 x 0.6.

Bache con dimensiones de 0.42 x 0.6 metros aproximadamente y también se pueden ver las dimensiones calculadas por GeoBach donde estableció medidas de 0.49 x 0.62 metros.



Figura 49: Bache real con dimensiones de 0.42x0.6m y detección hecha por GeoBach con dimensiones de 0.49x0.62m