Miguel Trujillo Lopez

Identificación de concentraciones de agua en caminos usando técnicas de inteligencia artificial

2026


**Universidad Autónoma de Querétaro**

**Facultad de Ingeniería**


# Identificación de concentraciones de agua en caminos usando técnicas de inteligencia artificial

## Tesis

Que como parte de los requisitos para obtener el Grado de

**Maestro en Ciencias en Inteligencia Artificial**

Presenta


**Miguel Trujillo Lopez**

**Dirigido por:**
**Dr. Juan Manuel Ramos Arreguin**


**Querétaro, Qro. a Febrero 2026**

**Universidad Autónoma de Querétaro**
**Facultad de Ingeniería**
**Maestría en Ciencias en Inteligencia Artificial**

# Identificación de concentraciones de agua en caminos usando técnicas de inteligencia artificial

## Tesis

Que como parte de los requisitos para obtener el Grado de

**Maestro en Ciencas en Inteligencia Artificial**

Presenta

**Miguel Trujillo Lopez**

Dirigido por

**Dr. Juan Manuel Ramos Arreguin**

Dr. Juan Manuel Ramos Arreguin
Presidente

Dr. Saul Tovar Arriaga
Secretario

Dr. Marco Antonio Aceves Fernández
Vocal

Dr.Jesús Carlos Pedraza Ortega
Suplente

MCIA. Karen Andrea Ramírez Arriaga
Suplente

Centro Universitario, Querétaro, Qro. México
Fecha de aprobación por el Consejo Universitario Febrero 2026

# Agradecimientos

Quiero agradecer primeramente a Dios, por ser mi guía constante y brindarme fortaleza, sabiduría y perseverancia a lo largo de este trayecto.

A mis papás y hermanos, por su amor incondicional, paciencia y apoyo permanente, por ser mi motivación diaria y el pilar fundamental que sostiene cada uno de mis logros.

A mi director de tesis Dr. Juan Manuel Ramos Arreguin y a todo el sínodo, por su orientación, dedicación y valiosas aportaciones durante el desarrollo de este trabajo. Su experiencia, compromiso y disposición fueron esenciales para fortalecer la calidad académica de esta investigación.

A la Secretaría de Ciencia, Humanidades, Tecnología e Innovación de México, por el apoyo otorgado para la realización de mis estudios de posgrado, el cual hizo posible mi formación académica y el desarrollo de este proyecto de investigación.

Finalmente, agradezco a todas las personas que de manera directa o indirecta, contribuyeron a la culminación de este trabajo.

## Resumen

La presencia de concentraciones de agua en caminos o carreteras puede ser una situación riesgosa para los automóviles y conductores de vehículos terrestres, estos pueden ocultar de la vista del conductor ciertos obstáculos o anomalías los cuales pueden causar accidentes. Tan solo en México el número de accidentes de tráfico aumenta considerablemente en temporada de lluvias. El desarrollo de tecnologías que permitan la detección de este tipo de fenómeno puede ayudar a la asistencia en la conducción, se han desarrollado soluciones bajo diversas metodologías como el uso de sensores o equipo óptico. Esta propuesta de trabajo tiene como objetivo la detección de acumulaciones de agua (charcos) en superficies implementando técnicas de inteligencia artificial en imágenes.



## Abstract

The presence of water on roads or highways can be dangerous for cars and drivers, as it can obscure certain obstacles or anomalies from the driver's view, which can cause accidents. In Mexico alone, the number of traffic accidents increases considerably during the rainy season. The development of technologies that enable the detection of this type of phenomenon can assist in driving. Solutions have been developed using various methodologies, such as sensors or optical equipment. This project aims to detect accumulations of water (puddles) on surfaces by implementing artificial intelligence techniques in images.

# DEFINICIONES

**AI** – Artificial Intelligence (Inteligencia Artificial)

**R-CNN** – Region-based Convolutional Neural Network (Red Neuronal Convolucional basada en regiones)

**SDD** – Single Shot Multibox Detector (Detector multicaja de disparo único)

**YOLO** – You Only Look Once (Solo miras una vez)

**SVM** – Support Vector Machine (Maquina de vectores de Soporte)

**GPS** – Global Position System (Sistema de posicionamiento global)

**GSM** - Global System for Mobile Communications (Sistema global de Comunicaciones móviles)

**LIDAR** – Light Detection and Ranging (Detector y medición de distancia por luz)

**ML** – Machine Learning (Aprendizaje Máquina)

**DL** – Deep Learning (Aprendizaje Profundo)

**RGB** – Red, Green, Blue (Rojo, Verde, Azul)

**ADAS** – Advanced Driver Assistance Systems (Sistemas avanzados de asistencia al conductor)

**ANN** – Artificial Neural Network (redes neuronales artificiales)

**CNN** – Convolutional Neural Network (Redes neuronales convolucionales)

**RoI** – Region of Interest (Región de Interés)

**AP** – Average Precision (Precisión Media)

**mAP** – Mean Average Precision (Precisión Media promedio)

**TP** – True Positive (Verdadero Positivo)

**FP** – False Positivo (Falso Positvio)

**FN** – False Negative (Falso Negativo)

**TN** – True Negative (Verdadero Negativo)

**IoU** – Intesection Over Union (Intersección sobre la unión)

# ÍNDICE

CAPITULO 1. INTRODUCCIÓN ..... 9
1.1 Introducción ..... 9
1.2 Antecedentes ..... 9
1.3 Estado del arte ..... 12
1.4 Descripción del problema ..... 17
1.5 Justificación ..... 18
1.6 Hipótesis ..... 19
1.7 Objetivo general ..... 19
1.8 Objetivos específicos ..... 19
1.9 Contenido ..... 19
CAPITULO 2. MARCO TEORICO ..... 20
2.1 Vehículos Autónomos ..... 20
2.2 Visión por Computadora ..... 21
2.3 Imagen ..... 21
2.4 Formatos de Imagen ..... 21
2.5 Tipo de imágenes ..... 22
2.6 Etapas y componentes de un sistema de Visión Artificial ..... 24
2.7 OpenCV ..... 24
2.8 Inteligencia Artificial ..... 25
2.9 Machine Learning ..... 25
2.10 Redes Neuronales ..... 25
2.11 Aprendizaje Profundo ..... 26
2.12 R-CNN ..... 27
2.12.1 Faster R-CNN ..... 28
2.13 SSD ..... 29
2.14 YOLO ..... 30

2.15 Comparativa de modelos ..... 33
2.16 Etiquetado ..... 34
2.17 Tareas de visión por computadora ..... 36
2.17.1 Reconocimiento en imágenes ..... 36
2.17.2 Clasificación de imágenes ..... 36
2.17.3 Localización ..... 37
2.17.4 Detección o identificación de objetos ..... 37
2.18 Métricas ..... 38
2.19 Aumento de datos ..... 40
2.20 Hiperparámetros ..... 41
CAPITULO 3. METODOLOGIA ..... 43
3.1 Búsqueda bibliográfica ..... 43
3.2 Recopilación de base de datos ..... 43
3.3 Recopilación de modelos usados ..... 48
3.4 Selección del modelo ..... 48
3.5 Entrenamiento del modelo ..... 49
3.6 Análisis del entrenamiento ..... 51
3.7 Análisis de los resultados ..... 56
CAPITULO 4. RESULTADOS Y DISCUSIÓN ..... 57
4.1 Evaluación de las métricas ..... 57
4.2 Comparativa estadística ..... 58
4.3 Resultados de detección en las imágenes ..... 58
4.4 Comparación entre versiones ..... 60
4.5 Comparación de desempeño ..... 64
4.6 4.2.6 Pruebas realizadas en video ..... 64
CAPITULO 5. CONCLUSIONES ..... 65
5.1 Conclusión ..... 65

5.2 Trabajo a Futuro ..... 66
ANEXOS..... 71

# ÍNDICE DE FIGURAS

Figura 1. Métodos de detección de agua [6]..... 10
Figura 2. Reflexión, refracción y dispersión de un cumulo de agua [17]..... 18
Figura 3. Waymo vehículo autónomo de Google [24]. ..... 20
Figura 4. Imagen Binaria [28]..... 23
Figura 5. Imagen en escala de Grises [25]. ..... 23
Figura 6. Imagen en RGB [25]. ..... 23
Figura 7. Neurona tipo Perceptrón [34]..... 26
Figura 8. Deep Learning Network [34]. ..... 27
Figura 9. Faster R-CNN red única y unificada para la detección de objetos [37]. .... 28
Figura 10. Arquitectura de SSD [38]. ..... 30
Figura 11. Funcionamiento de YOLO [39]. ..... 31
Figura 12. Non-Maximum Suppression (NMS) [40]..... 32
Figura 13. Arquitectura de YOLO [39]. ..... 32
Figura 14. Tipos de etiquetado espacial [43]..... 35
Figura 15. Procesos de etiquetado [43]. ..... 35
Figura 16. Tarea de detección [45]. ..... 36
Figura 17. Tarea de clasificación [45]. ..... 37

Figura 18. Tarea de detención [47]. ..... 38

Figura 19. Ilustración de Intersección sobre Unión (IoU) [50]. ..... 39

Figura 20. Aumento de datos usados en YOLOv8 [53]. ..... 41

Figura 21. Diagrama de flujo de la metodología. ..... 43

Figura 22. Repositorio del conjunto de datos ‘puddle-1000’’. ..... 44

Figura 23. Dispositivo de captura. ..... 45

Figura 24. Imagen con cúmulo de agua. ..... 45

Figura 25. Imagen sin cúmulo de agua. ..... 46

Figura 26. Condiciones de luz en el conjunto de datos ..... 46

Figura 27. Interfaz de etiquetado de CVAT para etiquetado de imágenes. ..... 47

Figura 28. Estructura del conjunto de datos ..... 48

Figura 29. Interfaz de Google colab Pro. ..... 49

Figura 30. Características del equipo. ..... 50

Figura 31. Esquema de entrenamiento. ..... 50

Figura 32. Validación K-fold. ..... 51

Figura 33. Aumento de datos ..... 53

Figura 34. Perdida durante el entrenamiento en YOLOv5. ..... 53

Figura 35. Métricas de validación en YOLOv5. ..... 54

Figura 36. Perdida durante el entrenamiento en YOLOv8. ..... 54

Figura 37. Métricas de validación en YOLOv8. ..... 55

Figura 38. Perdida durante el entrenamiento en YOLOv11. ..... 55

Figura 39. Métricas de validación en YOLOv11. ..... 56

Figura 40. Detección de YOLOV5..... 59

Figura 41. Detección de YOLOV8..... 59

Figura 42. Detección de YOLOV11 ..... 60

Figura 43. Desempeño del modelo en un día soleado y nublado. ..... 61

Figura 44. Desempeño del modelo en un atardecer y medio urbano. ..... 62

Figura 45. Detecciones faltantes con YOLO v8. ..... 63

Figura 46. Detecciones erróneas. ..... 63

# ÍNDICE DE TABLAS

Tabla 1. Resumen del estado del arte. ..... 16

Tabla 2. Formatos de Imagen [26]. ..... 22

Tabla 3. Comparación de los modelos ..... 33

Tabla 4. Definiciones de TP, FP, FN y TN [49]. ..... 38

Tabla 5. Resultados de YOLOv5 ..... 57

Tabla 6. Resultado de YOLOv8 ..... 57

Tabla 7. Resultados de YOLOv11 ..... 57

Tabla 8. Resultados de los modelos entrenados considerando media y desviación estándar..... 58

Tabla 9. Comparativa de los resultados con el estado del arte..... 64

# CAPITULO 1. INTRODUCCIÓN

## 1.1 Introducción

Las concentraciones de agua superficial en caminos es una condición que puede ser riesgosa para los conductores de vehículos terrestres, debido a que obstruyen la visibilidad de anomalías en el camino, como pueden ser agujeros, baches o algún obstáculo que pudieran existir en la ruta y provocar un accidente. Esta propuesta de trabajo tiene como objetivo la detección de acumulaciones de agua (charcos) en superficies implementando técnicas de inteligencia artificial en imágenes. Para esto, se propone el uso de modelos de inteligencia artificial como R-CNN (Region-based Convolutional Neuronal Network), SDD (Single Shot Multibox Detector) o YOLO (You Only Look Once), los cuales han sido usados en el estado del arte con diferentes aplicaciones como detección de charcos como criadero de mosquitos [1], la detección de charcos de agua en imágenes RGB en terreno de conducción [2], la detección de agua en caminos [3], o la detección de baches en caminos [4]. En nuestro caso, el principal objetivo es la detección de charcos para su aplicación en vehículos autónomos, donde al detectar el charco de agua, por seguridad se baje la velocidad o se realice alguna otra acción. Se busca emplear algunos de los métodos de Inteligencia Artificial (IA) mencionados previamente, así como algunos de los usados en [5].

## 1.2 Antecedentes

En este segmento se realiza una revisión de los métodos existentes para dar solución al problema de detección de concentraciones de agua superficial (charcos) en un camino. Esto con la finalidad de mejorar la auto navegación de vehículos autónomos terrestres. A través de los años han surgido diversas metodologías para poder dar solución a problemáticas similares como lo es la detección de agua. Dentro de los diferentes tipos de técnicas se puede resaltar el uso de radares, sensores, cámaras e imágenes de satélite, como se muestra en la Figura 1, incluyendo diferentes técnicas usadas [6][7].

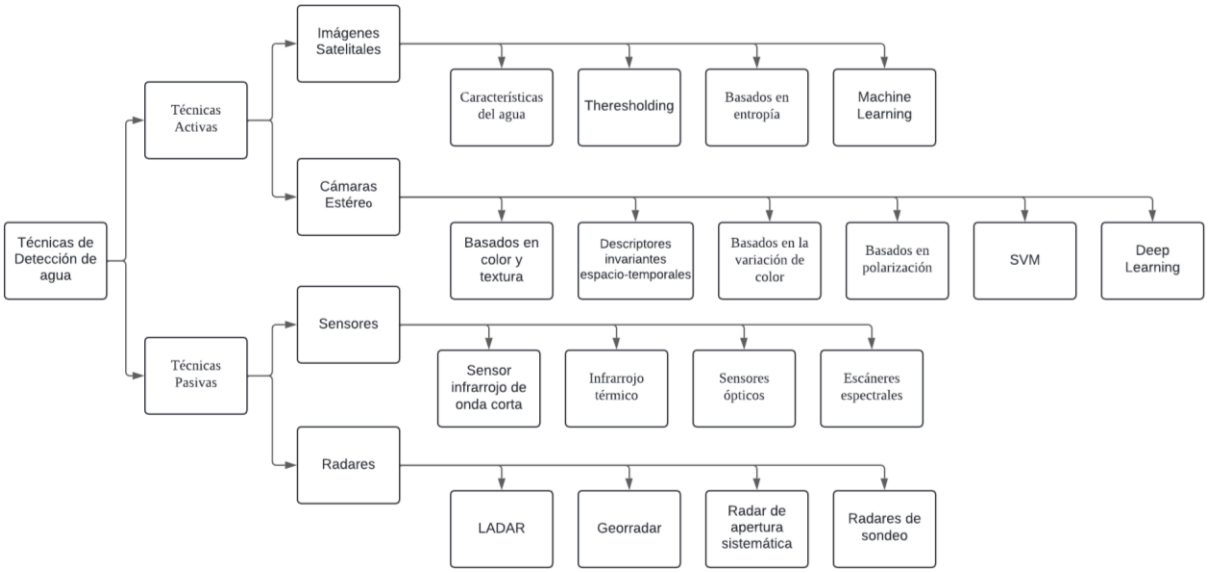



Figura 1. Métodos de detección de agua [6].

En el trabajo de Larry Matthies [8] cataloga diferentes variables las cuales afectan el trabajo de detección de obstáculos de agua, como lo son la diferencia entre el día y la noche, cuando el cumulo de agua refleja el cielo u objetos de su entorno, el tamaño del obstáculo y otros factores. En su trabajo resume la aplicación de diversos sensores que pueden llevar acabo esta tarea, LADAR, sensores infrarrojos y cámaras térmicas. También comparó la diferencia de brillo en imágenes y se mostró que el lugar que tiene un mayor brillo en las imágenes es el cielo, posterior el lugar donde se encuentra el charco y por último el resto de la imagen tiene menor valor de brillo. En sus resultados nos indica que en las imágenes es complicado detectar el charco cuando este refleja objetos de su entorno.

Al igual Arturo Rankin [9] concluye para su trabajo de “evaluación de rendimiento de vehículos no tripulados detección de agua” que la reflexión de objetos que se encuentra alrededor de la masa de agua pasa de forma desapercibida, igual se subestima cuando existen objetos en el contorno como rocas, vegetación o sedimentos.

Haiyan Shao [10] propone el uso de un sensor aplicando un método de luz estructural lineal para vehículos no tripulados, el cual puede detectar a un largo alcance (100 metros) en condiciones de luz del día. Para lograr estos resultados utilizó

un sensor que consta de una cámara multiespectral, una fuente puntual laser y un espejo piezoeléctrico.

Con la intención de detectar agua estancada como posible punto de criadero de mosquitos y otros insectos, debido a un incremento de casos de dengue y malaria Meghshyam G. [11] usó un cuadricóptero para la inspección y detección de dichas zonas de interés. En su trabajó enfatiza lo complicado que puede ser la tarea de identificar el agua debido a que actúa de formar similar a un espejo, reflejando lo que se encuentra a su alrededor, bajo condiciones de luz ambiental, los charcos son áreas de alto brillo y baja saturación. Se utilizó un clasificador basado en SVM (Support Vector Machine) y en un clasificador de flujo óptico.

La detección de agua no es solo un tema de interés para los vehículos autónomos si no también en área de cuidado, como el del trabajo presentado por Muhammad Siddique [12] el cual desarrolla un bastón inteligente para personas con discapacidad visual (invidentes) permitiendo la detección de obstáculos (como agua), para su desarrollo utiliza diversos materiales, como una tarjeta controladora Raspberry Pi, Motores de vibración, audífonos, módulos GPS y GSM, baterías entre otros. Para la detección de agua se usó la técnica de corto circuito de cobre, en el cual se tiene dos terminales de cobre conectadas a un receptor del sensor IR, en el momento que las terminales entren en contacto con el agua produce una salida en el pin del sensor lo cual indica que se encuentra en presencia de agua.

Minyoung Lee [13] propone el uso de un sensor LiDAR para la detección de charcos mediante la intensidad de reflexión del láser, ya que la intensidad es menos susceptible a cambios y esta varía dependiendo del material al que es dirigido (material de la superficie). Sus experimentos se realizaron simulando un charco de agua para el cual se cavó un agujero y se introdujo un cubo con agua, también se simulo un bache o agujero y una superficie plana. Según la intensidad de la salida se asocia con el tipo de ambiente en el que se encuentra el robot.

En nuestro caso, para el desarrollo del proyecto se usa una cámara para detectar las concentraciones superficiales de agua y métodos de inteligencia artificial.

## 1.3 Estado del arte

En su trabajo Sai Swathi [2] busca la detección de agua en calles (water, no water), para el cual desarrolló un algoritmo de clasificación bajo diferentes condiciones de color y tamaño de imagen y a su vez adaptando un algoritmo de aprendizaje maquina (ML, machine Learning) proveniente del “Statistics and Machine Learning Toolbox” de MATLAB. Usando el método de ML logró clasificar correctamente las condiciones de “no water”. Sin embargo, para las condiciones de “water” se encontró con varios errores. El algoritmo de clasificación tuvo mejor rendimiento sobre el algoritmo de ML. En este trabajo se propone utilizar un base de datos de imágenes propia y combinado con el uso de métodos de aprendizaje profundo (DL, deep learning) realizar la detección de las concentraciones de agua.

Juntao Li y Chuong Nguyen [3] en su trabajo de detección de obstáculos de agua proponen una red temporal convolucional 3D (T3D-FCN) inspirado por C3D network una extensión del trabajo presentado por Jonathan Long, Evan Shelhamer Y Trevor Darrell [14]. La entrada de video pasa a través de bloques convolucionales 3D, donde cada bloque cuenta con dos capas convolucionales, normalización por bloques y una capa de agrupación 3D. El modelo presentado logra tener un mejor rendimiento en comparación de las arquitecturas las cuales no cuentan con información temporal.

En el trabajo presentado por Hirotaka, Ibuki, Kenichiro, Tatsuya y Masanobu [15] el cual consiste en la detección de charcos para la planificación de rutas por parte de un robot móvil, evitando las concentraciones de agua, mediante las medidas de intensidad de reflexión provenientes de un telémetro láser bidimensional (2D LRF). En este trabajo se descarta el uso de cámaras o sistemas con ML debido a que estos sistemas pueden llegar a tener un peor comportamiento en la oscuridad. El método presentado logró determinar la presencia de charcos y el tamaño del mismo, para charcos con una longitud de 200 cm se consiguió una precisión de ± 1cm y ± 5cm en ancho.

Las contribuciones de Jian-Jun, Jun-Yan He,Wei Li y Qiang Peng [16] consisten en el primer método de detección de agua salpicada basado en DL que lleva por nombre “SWNet”. Dicho método tiene por objetivo resolver el problema del pavimento mojado y agua estancada en carretera. SWNet es una red de segmentación eficiente en la cual consiste de forma básica una estructura de codificador-decodificador, decodificador liviano y reutilización de índices de agrupación. El modelo propuesto tiene una mejor eficiencia y efectividad.

Manideep Sai [5] en su trabajo de detectar charcos de agua en imágenes RGB usando técnicas de deep Learning identificó técnicas apropiadas que pueden llegar a ser usadas parta el reconocimiento de diversos tipos de obstáculos basados en agua. Evaluó y analizó los modelos R-CNN y SDD, concluyó que SDD es más rápido de configurar y construir. También SDD tiene un rendimiento relativamente mejor.

Sonali Bhutad, Kailas Patil y Nishad Khare [1] presenta un mecanismo para poder identificar posibles criaderos de mosquitos (estancamiento de agua). Se propone una técnica de anclaje de cajas para reducir la clasificación errónea en la detección de agua estancada. El trabajo se realizó utilizando YOLO V3. Consiguiendo de forma efectiva la distinción de las masas de agua de las superficies húmedas. Sin embargo, en casos donde la incidencia de la luz sobre el agua es mayor, existe una clasificación errónea.

Para detectar encharcamientos usando drones, se aplican diversos algoritmos como Fully Convolutional Networks with Reflection Attention Units (FCN-RUA) [17] para identificar de las propiedades del agua, así como la detección de superficies basado en ondas generadas por el impulso de un dron, y a partir del movimiento de la altura estática (SAM, Static Altitude Movent). Este trabajo es presentado por H.N. Samaranayake [18]. Donde se concluye que el modelo UNet-RUA tuvo un mejor rendimiento que el modelo FCN-RUA. Sin embargo, para que esto ocurra, la vista debe ser tomada de forma angular y debe estar estable el reflejo de la luz del medio ambiente. El rendimiento del modelo se ve afectado por las ondas generadas por las

hélices del dron cuando se encuentra a muy baja altura respecto al agua, o cuando el viendo distorsiona la reflexión del agua, así como objetos en el agua.

Los sistemas avanzados de asistencia al conductor o por sus siglas en inglés (ADAS) juegan un rol muy importante hoy en día, por lo que Ningbo Long, Han Yan, Liqiang Wang, Haifeng Li y Qing Yang [19]. En su trabajo presentan un sistema de función de datos multisensor basado en la cámara estéreo en color de polarización y LIDAR, para lograr la detección y reconocimiento de objetos. Se utiliza la información de la polarización a nivel de pixeles y la nube de puntos LIDAR para poder detectar superficies resbaladizas y charcos en un entorno urbano. Los resultados se mantienen estables en diversas condiciones de luz y rango.

La detección de baches en condiciones reales, trabajo presentado por Maroš Jakubec, Eva Lieskovská, Boris Bucko y Katarína Zábovská [4] comparan diversos modelos basados en redes neuronales convolucionales tradicionales, basados en región y YOLO Demostrando que YOLO supera a los modelos R-CNN en términos de precisión y velocidad de indiferencia en diferentes en condiciones como despejado, lluvia, atardecer y tarde. Mientras que los modelos R-CNN tienen un mejor rendimiento en condiciones peores de visibilidad como la noche

Aditya Kumar y Ayesha Choudhary [20] en su trabajo presentado para la segmentación de charcos de agua usando DP en un ambiente con condiciones no favorables, utilizan YOLO V8 como técnica basada en CNN. Para la creación del dataset en este trabajo se montó una cámara en el tablero de un vehículo el cual iba grabando diferentes secciones de ruta de forma continua, las condiciones de captura fueron tomadas en una hora específica del día y bajo condiciones climáticas específicas para que la captura contase con un gran nivel de detalle. Después del entrenamiento los resultados de este trabajo nos indica que el modelo es capaz de detectar y segmentar en un 63% del charco con más del 50% de confianza.

En la Tabla 1, se muestra un resumen de diversos artículos relacionados con el estado del arte, los cuales son tomados en cuenta para el desarrollo del presente trabajo.

| Autores | Métodos | Resultados | Características | Dataset |
|---|---|---|---|---|
| Sai Swathi [2] (2019) | Toolbox de machine learning de MATLAB | Accuracy Gray: 80%<br>Accuracy Color: 85%<br>Accuracy Machine Learning: 75% | La cámara de teléfono se encuentra cerca de las luces delanteras del carro en un ángulo de 75 grados. | 288 imágenes de soleado, lluvioso y nublado de la misma ruta. |
| Juntao Li [3] (2019) | Red temporal convolucional 3D (T3D-FCN) | **En ruta (ONR)**<br>F1: 0.68<br>Precission:0.79<br>Recall: 0.62<br>Time: 0.23<br>**Fuera de ruta (OFR)**<br>F1: 0.73<br>Precission:0.87<br>Recall: 0.63<br>Time: 0.23 | Todas las imágenes fueron reajustadas a un tamaño de 240x320 pixeles. | Puddle-1000 dataset dividido en 2, ONR y OFR. |
| Hirotaka [15] (2019) | Laser Reflection Intensity | Logró determinar la presencia de charcos y el tamaño del mismo, para charcos con una longitud de 200 cm se consiguió una precisión de ± 1cm y ± 5cm en ancho | Utilizó sensores para la resolución del problema. | N/A |
| Jian-Jun [16] (2022) | SWNet | Pixel Accuracy: 99.8<br>Mean Pixel Accuracy: 87.2<br>Mean Intersection Over Union: 78.5<br>Frecuency Weighted Intersection over Union: 99.6 | Se etiqueta las imágenes usando LabelMe de forma manual | El dataset cuenta con 7,387 imágenes primer conjunto de datos de agua salpicada. |
| Manideep Sai [5] (2020) | R-CNN y SDD | **R-CNN**<br>Accuracy: 87.48<br>Precision:0.964<br>Recall: 0.885<br>F1-Score: 0.922<br>Training Time: 9 hrs<br>**SDD**<br>Accuracy: 76.46<br>Precision:0.98<br>Recall: 0.739<br>F1-Score: 0.842<br>Training Time: 15 hrs | Se etiquetó el dataset de forma manual usando LabelImg | Puddle-1000 |
| Sonali Bhutad [1] (2023) | YOLO V3 | MAP: 85.67<br>Precision: 0.82<br>Recall: 0.86<br>Time: 8s | Las imágenes fueron reajustadas a 256x256 | 3812 imágenes, 1200 son de Google para entrenamiento, 1976 tomadas por |

| Autores | Métodos | Resultados | Características | Dataset |
|---|---|---|---|---|
| | | | | un celular, 100 imágenes rotadas a 90 grados y 536 imágenes de prueba de Google |
| Harin Samaranayake [18] (2020) | Unet-Only<br><br>FCN-RUA | **UNet-8-RUA**<br>ONR<br>Accuracy: 99.2<br>Precision: 78.2<br>Recall: 82.11<br>F1-Measure: 80.11<br>**OFR**<br>Accuracy: 99.34<br>Precision: 88.98<br>Recall: 79.56<br>F1-Measure: 84.0<br>**DWF**<br>Accuracy: 99.23<br>Precision: 79.44<br>Recall: 82.57<br>F1-Measure: 80.97 | Las imágenes capturadas tienen una resolución de 1920x1080 | Puddle-1000 y dataset creado por el autor: Drone-Water Front y Down contiene 119 imágenes |
| Ningbo Long [19] (2022) | Información de polarización a nivel de pixeles y nube de puntos de sensor LIDAR | Accuracy: 98.71 | Se consigue buenos resultados en diversas condiciones de luz. | N/A |
| Maroš Jakubec [4] (2023) | R-CNN<br><br>YOLO | **Sparse R-CNN**<br>Mean Precision: 55.4%<br>Mean Recall: 46.6%<br>Mean mAP: 51.4<br>Nº Parámetros: 77.8M<br>**YOLO V7**<br>Mean Precision: 73.2%<br>Mean Recall: 56.3%<br>Mean mAP: 55.6<br>Nº Parametros: 36.9M | Cuenta con una resolución de 1920x1080 | Creado por el autor con un total de 2099 fotos en diversas condiciones climatológicas |
| Aditya Kumar [20] (2023) | YOLO V8 | **Sin Aumento**<br>mAP50: 0.559<br>mAP50-90: 0.321<br><br>**Con aumento**<br>mAP50: 0.626<br>mAP50-90: 0.496 | Cámara montada en el tablero del carro, captura bajo cierta hora del día y ciertas condiciones climatológicas, se aumentó la base de datos modificando las imágenes originales para añadir más complejidad y mejorar el rendimiento. | Creado por el autor |

Tabla 1. Resumen del estado del arte.

## 1.4 Descripción del problema

Las concentraciones de agua superficial o también conocidos comúnmente como charcos de agua es un fenómeno que ocurre cuando el suelo cuenta con un nivel diferente o el suelo no tiene un escape de agua, logrando así que exista un estancamiento de agua. La profundidad del charco dependerá de la diferencia de altura o nivel que exista.

Los charcos pueden llegar a ser inadvertidos debido a muchos factores como lo pueden ser el tamaño del cúmulo de agua, al ángulo de visión del conductor y el ángulo de incidencia de la luz siendo estos los más relevantes. Por lo cual, existe una gran aplicabilidad en los vehículos autónomos los cuales cuentan con diversos dispositivos para la detección de obstáculos. Se han desarrollado diversos algoritmos, pero algunos tienen problemas con la iluminación, la calidad es baja, especialmente los desarrollados para trabajar con imágenes. En algunos casos, se usa la cámara de un teléfono inteligente [2], pero el mismo autor considera que se puede mejorar usando una cámara digital dedicada a fotos y video. En la Figura 2 se muestra el proceso de reflexión, refracción y dispersión en un charco de agua presentado en el trabajo de Han [17].

En el estado del arte, se realiza la detección de charcos con diferentes propósitos, por ejemplo: detección de charcos como criadero de mosquitos [1], la detección de charcos de agua en imágenes RGB en terreno de conducción [2], la detección de agua en caminos [3], la detección de agua salpicada en carreteras [16], la detección de baches en el mundo real [4] e incluso simplemente se detectan las concentraciones de agua [16]. En el caso de [16], la detección se realiza en caminos de terracería, con una exactitud del 87% con el modelo R-CNN y del 76% con SDD, por lo que en el presente trabajo se va a realizar la detección en imágenes correspondientes a carreteras o calles urbanas de México, usando al menos uno de los métodos mencionados previamente. En este caso, al cambiar el ambiente de trabajo, se deben realizar adecuaciones a los métodos empleados, debido a los cambios de iluminación, tipo de piso, entre otros.

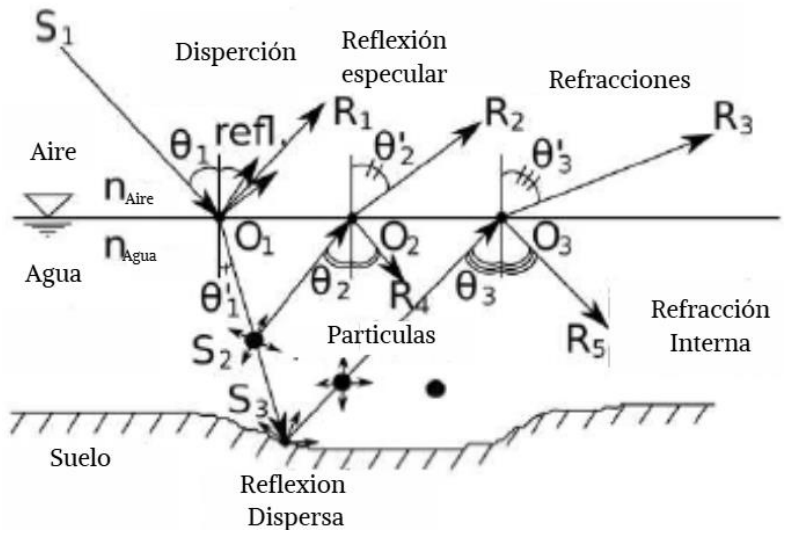



Figura 2. Reflexión, refracción y dispersión de un cumulo de agua [17].

## 1.5 Justificación

Manejar bajo la lluvia o después de esta, es una tarea que requiere mucha precaución de parte del conductor, debido a la inestabilidad del suelo y los inconvenientes que pueda haber en la ruta. En México entre el año 2021 al 2022 los accidentes causados por lluvias incrementaron en un 13%, y en los últimos 5 años alrededor del 25% de accidentes reportados se presentan en temporada de lluvias [21]. A consecuencia de las lluvias, la presencia de charcos de agua en las calles puede evitar que se vean algunos tipos de obstáculos, baches, alcantarillas sin tapa, entre otros. Por lo que, sin la precaución o alerta necesaria esto podría causar daños a la estructura del vehículo o se podría ver afectada la integridad física de los pasajeros.

Existen trabajos los cuales abordan esta problemática mediante el desarrollo de sistemas con el uso de diferentes tipos de sensores [15] [22], este tipo de desarrollo suelen ser costoso, por otro lado, las técnicas que implementan inteligencia artificial suelen ser más eficientes y de menor costo. Por lo tanto, es importante el desarrollo de trabajos relacionados con la detección de concentraciones de agua superficial (charcos de agua). En nuestro caso particular, se propone trabajar en la detección de charcos de agua usando algoritmos de inteligencia artificial como podría ser YOLO, R-CNN o SWNet basados en imágenes y con buena iluminación.

## 1.6 Hipótesis

Se puede detectar concentraciones de agua en caminos usando técnicas de inteligencia artificial, considerando diversos tipos de camino, y el uso de la métrica de precisión para igualar o mejorar el 76% mencionado en el estado del arte.

## 1.7 Objetivo general

Aplicar modelos de inteligencia artificial para la detección de charcos de agua en imágenes de caminos, mediante la implementación de al menos un algoritmo basado en el estado del arte.

## 1.8 Objetivos específicos

- Identificar un modelo de inteligencia artificial adecuado para la identificación de concentraciones de agua, realizando pruebas en imágenes.
- Generar una base de datos para usarse en el entrenamiento del algoritmo, para la identificación de charcos de agua mediante algoritmos de inteligencia artificial elegido.
- Analizar el rendimiento del algoritmo usado, comparándolo con resultado del estado del arte para determinar su eficiencia en la identificación de las concentraciones de agua.

## 1.9 Contenido

El contenido de este reporte se dividirá en los siguientes capítulos: Capítulo 2: marco teórico, en el cual se aborda todos los conceptos teóricos para llevar a cabo la resolución del problema; Capítulo 3: Metodología, en este capítulo se explica el proceso que se llevó a cabo para la solución a la problemática; Capítulo 4: se expone y se discute los resultados obtenidos. Por último, se dan las conclusiones obtenidas en este trabajo.

# CAPITULO 2. MARCO TEORICO

## 2.1 Vehículos Autónomos

Anteriormente los vehículos solían ser maquinas completamente mecánicas. Sin embargo, con el paso de los años y el crecimiento de la tecnología, el Internet de las cosas (IoT) y los sistemas embebidos, los vehículos autónomos cuentan con una gran variedad de herramientas de hardware y software que permiten implementar la conducción autónoma en cierto grado. De esta manera, los vehículos autónomos son máquinas funcionales e inteligentes, dando paso a un gran campo para la investigación y la innovación. Una de las principales motivaciones de los investigadores se enfoca en la seguridad de los tripulantes y personas que se encuentren alrededor del mismo, y es por esto que se implementan algoritmos para la detección de peatones, señales de tráfico, estacionado automático, hasta la implementación de algoritmos que hagan posible la conducción autónoma [23]. La Figura 3 es una muestra de un vehículo autónomo desarrollado por Google.

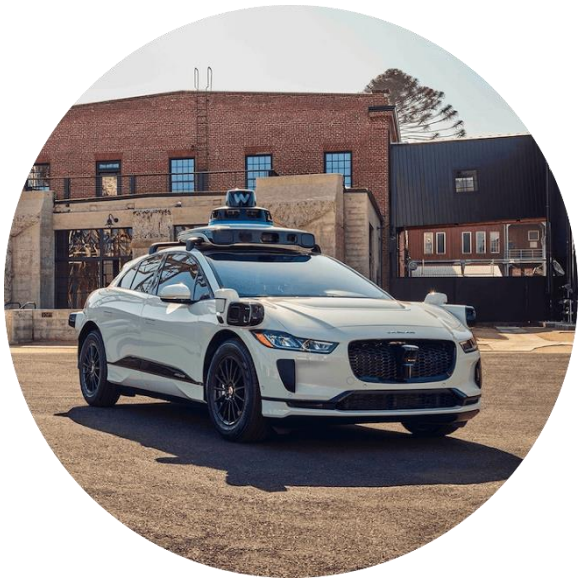


Figura 3. Waymo vehículo autónomo de Google [24].

## 2.2 Visión por Computadora

El área encarga de brindar a la computadora la capacidad de “ver” u “observar” es la llamada visión artificial o visión por computadora. Dicha área permite que la computadora obtenga datos de imágenes digitales o videos y así poder tomar decisiones. Se puede encontrar similitudes entre el proceso de análisis realizado por el del ser humano y el de las computadoras. Sin embargo, la visión humana cuenta con ciertas ventajas como lo es el contexto, mientras que las computadoras necesitan de cámaras, datos, algoritmos complejos para realizar tareas de mayor precisión y rapidez, como la inspección o detección de objetos o patrones [25].

## 2.3 Imagen

Puede ser considerada una imagen digital como una representación visual discreta de datos que poseen tanto información espacial (diseño) como de intensidad (color), debido a una fuente radiante de luz. Una imagen también puede ser definida como una función bidimensional *f(x,y)*, y se compone de un número finito de elementos donde cada uno cuenta con una ubicación y valor particular, véase la ecuación (1). Estos elementos se les denomina elementos de una imagen, pixeles. Píxel es el termino más utilizado para referirse a los elementos de una imagen digital [25] [26] [27].

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \dots & f(0,N-1) \\ f(1,0) & f(1,1) & \dots & f(1,N-1) \\ \vdots & \vdots & \ddots & \vdots \\ f(M-1,0) & f(M-1,1) & \cdots & f(M-1,N-1) \end{bmatrix} \quad (1)$$

## 2.4 Formatos de Imagen

En la práctica es necesario almacenar y mostrar las imágenes, de ahí surgió el desarrollo de formatos estandarizados de imágenes digitales, a continuación, en la

Tabla 2 se mencionan los principales formatos que se manejan en la actualidad [26].

| Acrónimo | Formato | Propiedades |
|---|---|---|
| GIF | Graphics Interchange Format | Limitado a 256 colores (8-bits) compresión sin perdida. |
| JPEG | Joint Photographic Experts Group | Compresión con perdida, hay variantes sin perdida. |
| BMP | Bit Map Picture | Formato de imagen básico, generalmente limitado, existe perdida con compresión, hay variantes sin perdida. |
| PNG | Portable Network Graphics | Formato sin perdidas de compresión, busca remplazar a GIF. |
| TIF/TIFF | Tagged Image (File) Format | Muy flexible, formato detallado y adaptable, existen variante con compresión y sin compresión. |

Tabla 2. Formatos de Imagen [26].

## 2.5 Tipo de imágenes

Imágenes Binarias también conocidas como imágenes lógicas son aquellas imágenes en las cuales sus pixeles están representados por valores de {0,1}, donde el valor de '1' corresponde al color blanco y el '0' al color negro. Este tipo de imágenes pueden ser interpretada por una simple cadena de bits, como se muestra en la Figura 4 [26].

Imágenes en escala de grises, al igual que el tipo de imagen anterior cada pixel tiene un valor único, en este caso el valor del pixel representa la intensidad de gris en dicho punto Figura 5. Dicha intensidad está delimitada con respecto a la resolución de bit que tenga el formato de imagen a trabajar [26].

RGB o color verdadero, en la cual cada pixel toma tres tipos de valor los cuales corresponden a Rojo(R), Verde(G) y Azul(B). Comúnmente este tipo de imagen se almacenan de la siguiente manera, (R1G1B1, R2G2B2, … RnGnBn) [26]. Observe la Figura 6.

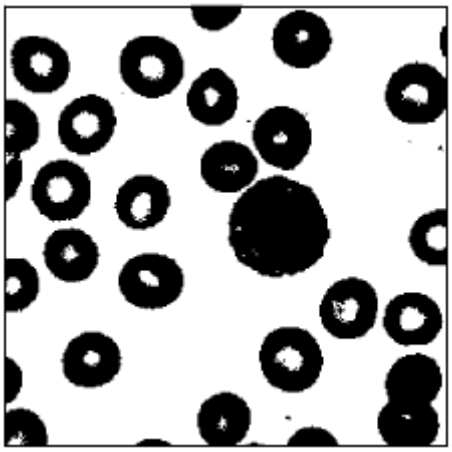


Figura 4. Imagen Binaria [28].

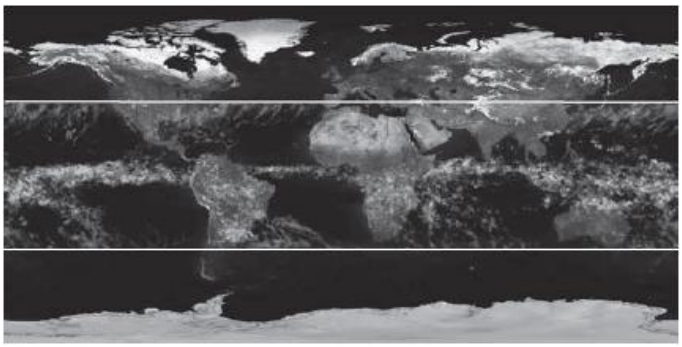


Figura 5. Imagen en escala de Grises [25].

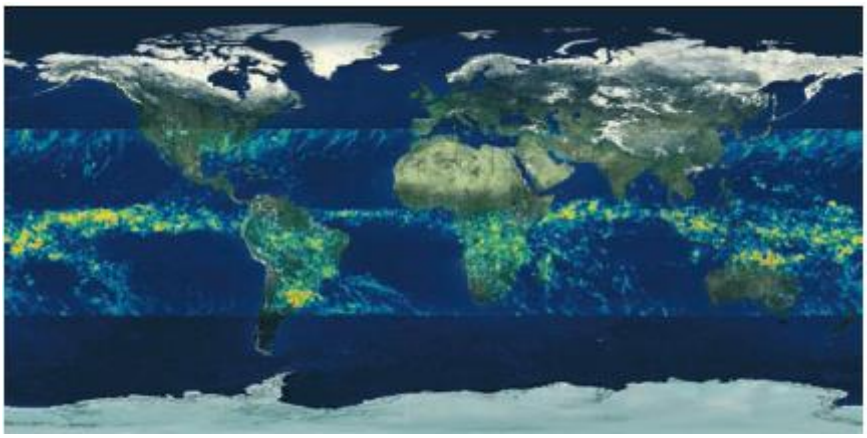


Figura 6. Imagen en RGB [25].

Punto Flotante a diferencia de los anteriores tipos de datos los cuales los valores de cada pixel contenían valores enteros, en este tipo de dato guarda valores de tipo flotante, comúnmente pueden representar otro valor de medición más que simplemente la intensidad de color en un pixel. El uso de imágenes de punto flotante ha incrementado debido al uso de rango dinámico y fotografía estéro como se muestra

en la figura 6. Este tipo de imágenes suelen ser almacenadas en los formatos tipo TIF. Sin embargo, aún hay mucha limitación en los formatos que permiten su almacenamiento [26].

## 2.6 Etapas y componentes de un sistema de Visión Artificial

La visión artificial cuenta con varias etapas y componentes, tanto de software como hardware para poder llevarse a cabo. La primera etapa de este proceso es la adquisición de las imágenes mediante el uso de sensores ópticos como las cámaras. El tipo de sensor dependerá de la aplicación, una tarjeta de adquisición permite digitalizar la señal, la cual es almacenada en memoria y posteriormente se muestra en monitores. La segunda etapa consiste en el preprocesamiento de la imagen con la finalidad de mejorarla y llegar a un resultado con mayor probabilidad de éxito. La etapa de la segmentación consiste en dividir la imagen en parte o en objetos que la forman, la segmentación facilita mucha la solución del problema si esta se realiza de forma adecuada y llevará a su falla si esta es realizada de forma errónea. La parametrización es la etapa en la cual se extraen los rasgos que producen alguna información cuantitativa de interés o aquellos rasgos que son fundamentales para diferencias un objeto de otro. Y por último la clasificación el cual consiste en el reconocimiento e interpretación de la imagen para la cual se les asigna una etiqueta a los objetos reconocidos [27].

## 2.7 OpenCV

La idea de construir una infraestructura de visión por computadora disponible a todos fue concebida en los laboratorios de investigación de Intel. OpenCV viene del término en inglés “Open Source Computer Vision”, esto es un framework el cual contienen herramientas, librerías y módulos permitiendo la implementación de aplicaciones de visión por computadora. Actualmente es una de las herramientas más ampliamente adoptadas por los desarrolladores debido a su escalabilidad a entornos reales y tener disponibilidad en varios lenguajes de programación como Python, C++ y Java, y a su vez en distintos sistemas operativos, Windows, Linux, macOS, Android y iOS [29].

## 2.8 Inteligencia Artificial

Inteligencia artificial es concepto el cual ha ido evolucionando a lo largo de los años y aún hoy en día no sé tiene una definición que agrade a todos, debido a que aún se debate en la definición propia de “inteligencia”. Se define Inteligencia Artificial como el área encargada de hacer que las computadoras, robots o máquinas repliquen el proceso de “pensar” y procesar información mediante el uso de modelos matemáticos, con la finalidad de resolver problemas como la toma de decisiones, la clasificación, la predicción, entre otros [30]. Dentro de la inteligencia artificial existen muchos campos de aplicación, y la visión por computadora es una de ellos.

## 2.9 Machine Learning

Aprendizaje Maquina (ML) es una rama de la inteligencia artificial que puede resolver problemas como la toma de decisiones, mediante la generalización de ejemplos dados, recomendación de artículos, predicción de futuros valores, entre otros. Dos de los métodos más usados dentro de ML es el aprendizaje supervisado el cual lleva su nombre debido a que un “maestro” proporciona la supervisión de los resultados esperados, por lo cual estos datos deben de contar con una etiqueta. En este método los resultados permiten que sean fáciles de entender y de medir sus rendimientos. Por otro parte, el aprendizaje No-Supervisado es aquel que no cuenta con etiquetas o “maestro”, solamente se le proporcionan datos de entrada y nos devuelve datos de salida. Para usar este método es necesario contar con el entendimiento de sus resultados y es difícil de evaluar [31].

## 2.10 Redes Neuronales

Las redes neuronales artificiales (ANN) es un subconjunto de ML y su nombre y estructura se inspira del mismo cerebro humano, tratando de imitar la forma en la que las neuronas transmiten la información. Las ANN se componen por capas llenas de nodos las cuales se encargan de transmitir capa por capa con el uso de pesos y umbrales, asociando información hacia una capa de salida. Si la salida de un nodo sobrepasa el umbral especificado este se activa y permite el paso de la información a la siguiente capa. De lo contrario la información no fluye (Figura 7). Las redes

neuronales se basan en entrenamiento para poder aprender y mejorar su rendimiento y precisión, dicho entrenamiento requiere cierta inversión de tiempo, pero una vez entrenadas de forma correcta estos algoritmos se convierten en herramientas muy rápidas y precisas [32], [33].

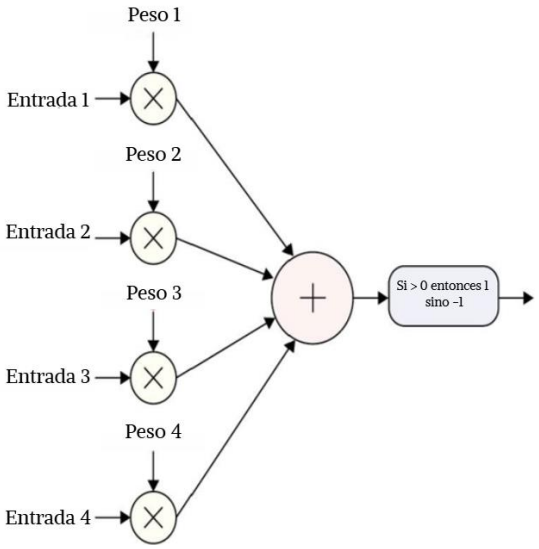



Figura 7. Neurona tipo Perceptrón [34].

## 2.11 Aprendizaje Profundo

El deep Learning (DL) es un área de las ANN el cual consiste en una red neuronal con tres o más capas, el termino profundo hace referencia al número de capas ocultas que tiene el modelo (Figura 8). Por lo cual permite introducir datos no estructurados como lo pueden ser el uso de texto o imágenes, extrayendo las características principales eliminado de esta manera el uso de supervisores o expertos humanos. Mediante los procesos de gradiente y retro-propagación, el algoritmo de DL puede ajustarse y adaptarse para obtener una mejor precisión [35].

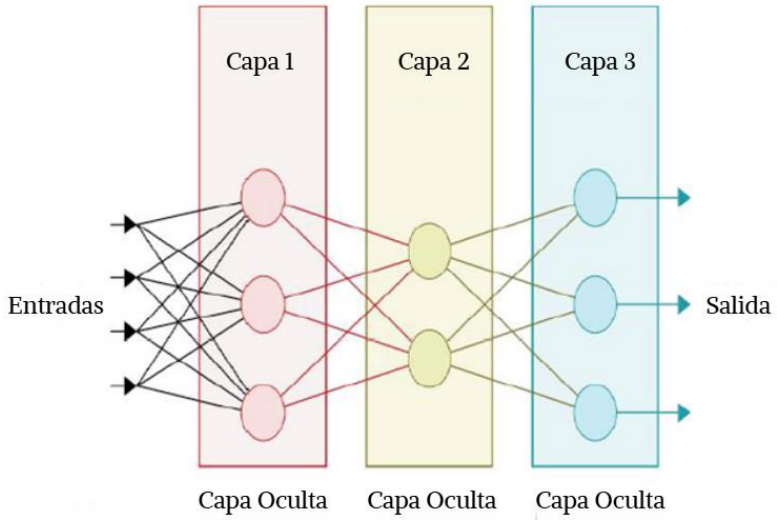



Figura 8. Deep Learning Network [34].

## 2.12 R-CNN

Las redes neuronales convolucionales (CNN) es un proceso que tiene inspiración biológica, este es un tipo de red neuronal la cual es comúnmente usada en visión artificial. La arquitectura de este tipo de red consiste en distintas capas, más específicamente en cuatro:

- Capa convolucional
- Unidad lineal rectificadora (ReLu)
- Capas de agrupamiento
- Capas completamente conectadas (FC)

Con los avances en las técnicas de DL, especialmente aquellas basadas en región (R-CNN,Region Based CNN), han tenido gran éxito en diferentes áreas de ML, estadística y visión por computadora, en esta última en tareas como detección de objetos, categorización y segmentación de imágenes. Esto se debe a que la red solo identifica la clase del objeto mas no la ubicación de este en la imagen. Especialmente cuando se encuentran más de un objeto en la imagen esta no suele rendir bien debido a la interferencia que puede existir entre los objetos. R-CNN identifica regiones de interés (RoI) mediante búsqueda selectiva y utilizando una red convolucional las

clasifica ya sea como primero o segundo plano. R-CNN es considera el mejor detector de objetos basados en CNN [36].

### 2.12.1 Faster R-CNN

Faster R-CNN es una arquitectura basada en R-CNN desarrollada por Shaoquing Ren [37], este modelo mejoró significativamente sus predecesores R-CNN y Fast R-CNN debido a la integración de “propuestas” las cuales son una generación de regiones. Faster R-CNN busca unificar dos tareas que se hacían por separado, las cuales eran las propuestas de regiones de interés (ROIs) y la clasificación y ajuste (refinamiento) de dichas regiones. Para poder realizar esta tarea se cuenta con dos módulos, en el primero se cuenta con una red totalmente convolucional profunda la cual se encarga de proponer regiones, mientras que en el segundo modulo se encuentra el detector en el cual se usa las propuestas de regiones, el módulo llamado Red de Propuestas Regionales (RPN) le dice al módulo de Fast R-CNN hacía donde observar. En la Figura 9 puede observar un diagrama donde se observan los módulos. [37]

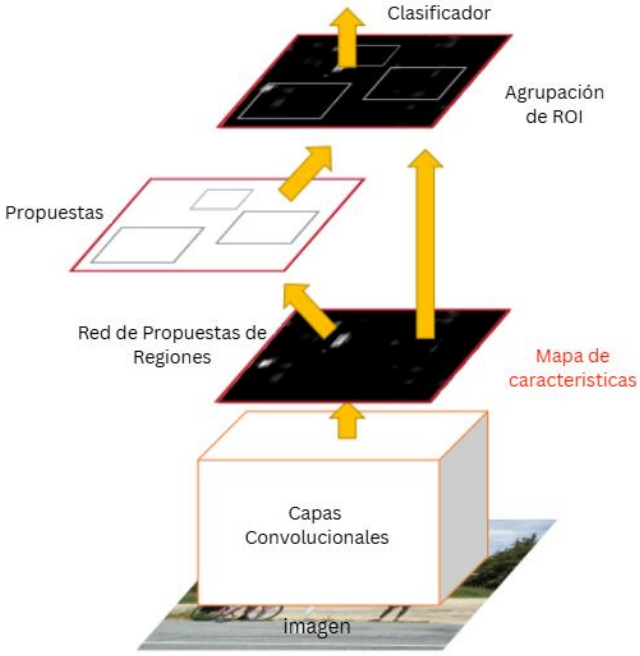



Figura 9. Faster R-CNN red única y unificada para la detección de objetos [37].

La arquitectura de Faster R-CNN se puede describir en las siguiente cuatro partes. Red Convolucional Base, en esta sección se usa una CNN pre-entrenada la cual se encuentra de extraer un mapa de características de la imagen. Red de Propuestas Regionales es la parte principal de Faster R-CNN donde se busca predecir regiones en donde pueda haber objetos directamente desde el mapa de características, como parte del funcionamiento se desliza una ventada de 3x3 sobre el mapa de características, para cada posición se genera “anclas” las cuales son cajas de diversos tamaños y proporciones, para cada una de estas “anclas” se predice la probabilidad de contener algún objeto de interés y se ajusta la caja. La sección de agrupamiento de ROI conecta RPN con la parte final del detector, en el cual las regiones propuestas de RPN se aplican sobre el mapa de características, cada región e interés se corta a un tamaño fijo, permitiendo suministrar diferentes tamaños de regiones a una misma red completamente conectada. El siguiente proceso consiste en los procesos de predicción de la clase del objeto y regresión en el cual se ajustan las cajas delimitadoras para mejorar la precisión.

## 2.13 SSD

R-CNN ha sido modificado y existen diversas variantes fast R-CNN, faster R-CNN, Region based Fully Convolutional Networks (R-FNC), Single Shot Dectetor (SDD), You Only Look Once (YOLO), entre otros.

Tanto en la R-CNN y las R-FCN, estos dos modelos llevan a cabo una propuesta regional mediante el uso de RPN para generar regiones de interés y clasificación de zonas de interés, mediante el uso de capas completamente conectadas (FCL) o por medio de capas convolucionales sensibles, SDD se encarga de realizas ambas actividades en una sola toma, a medida que se procesa la imagen se predice el cuadro limitador y la clase simultánea [36].

Wei Liu propuso SSD [38] con el objetivo de combinar precisión y una alta velocidad de respuesta en tiempo real, algo que en los modelos R-CNN es difícil de lograr. El objetivo de SSD es realizar la tarea de detección en una sola instancia o pasada mediante una red convolucional completamente integrada, mientras que en los modelos basados en R-CNN realizan la tarea en dos etapas, la propuesta de regiones y la clasificación de clases, SSD realiza ambas tareas en solo paso sobre el mapa de características [38].

La estructura de SSD consiste en una Red Convolucional Base pre-entrenada encargada de generar los mapas de características de la imagen de entrada. Posteriormente, se encuentran las capas de detección multiescala, las cuales son el núcleo de SSD, en el cual se utiliza diversos mapas de características. Cada uno de estos mapas tienen diferente dimensión y es encargado de detectar objetos de distintos tamaños. Enseguida, el modelo realiza las predicciones usando cajas predeterminadas, en este proceso, cada celda del mapa de características genera predicciones usando cajas predeterminadas o cajas ancla, similar al proceso usado en Faster R-CNN. Para cada posición en el mapa de características se define un numero K de cajas, se predice la confianza o probabilidad de que se encuentre un objeto de interés y se ajusta el tamaño de la caja delimitadora. Como último paso a todas las predicciones del mapa de características se les aplica un ajuste donde se descartan las cajas con poca confianza y un filtro de supresión de No Máximos, el cual elimina las cajas que se solapan unas a otras. En la Figura 10 se muestra la composición de la arquitectura del modelo [38].

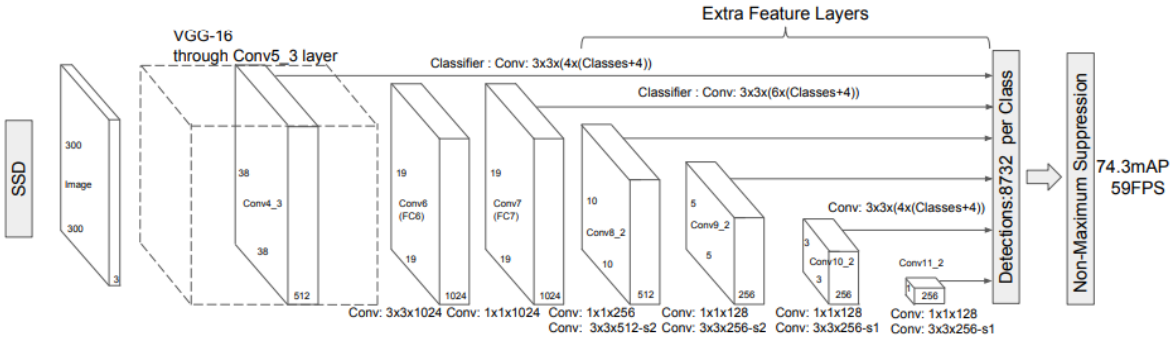



Figura 10. Arquitectura de SSD [38].

## 2.14 YOLO

YOLO (You only Look Once) es un modelo de inteligencia artificial el cual es ampliamente usado en tareas de detección de objetos, es popular por su rapidez y alta precisión, fue introducido por Joseph Redmon [39] en el año 2016. Se replantea la detección de objetos como un problema de regresión, directamente desde los pixeles de la imagen hasta las coordenadas de los cuadros delimitadores y las probabilidades de clase.

YOLO es simple y rápido debido a que tratan el problema como una regresión, con una sola red convolucional simultáneamente predice múltiples posibles cajas delimitadoras y calcula las probabilidades de clase para cada una de ellas. En comparación con otro tipo de técnicas basadas en ventanas deslizantes YOLO analiza la imagen completa durante el entrenamiento y prueba, lo cual le permite codificar información contextual sobre la clase y su apariencia de forma implícita.

YOLO unificó los pasos de detección de objetos al detectar todos los cuadros delimitadores de forma simultánea. Para lograr esto, YOLO divide la imagen de entrada en una cuadricula S × S y predice B número de cajas delimitadoras de la misma clase, junto con la probabilidad de que cada recuadro pertenezca a una de las C clases posibles para cada celda de la cuadricula. La predicción consta de cinco valores: Pc, bx, by, bh, bw, donde Pc es la confianza de predicción de que el rectángulo contenga un objeto y se representa la intersección sobre la unión entre la caja predicha y una anotación verdadera; bx y by corresponden a las coordenadas del centro del cuadro en relación con la celda de la matricula; mientras que bh y bw representan la altura y ancho de del recuadro en relación con la imagen completa. La salida de YOLO es un tensor de tamaño S x S (B x 5 + C), al que opcionalmente se le aplica la supresión de no-maximos (NMS), para eliminar las detecciones duplicadas [39] [40], esto se puede observar en la Figura 11.

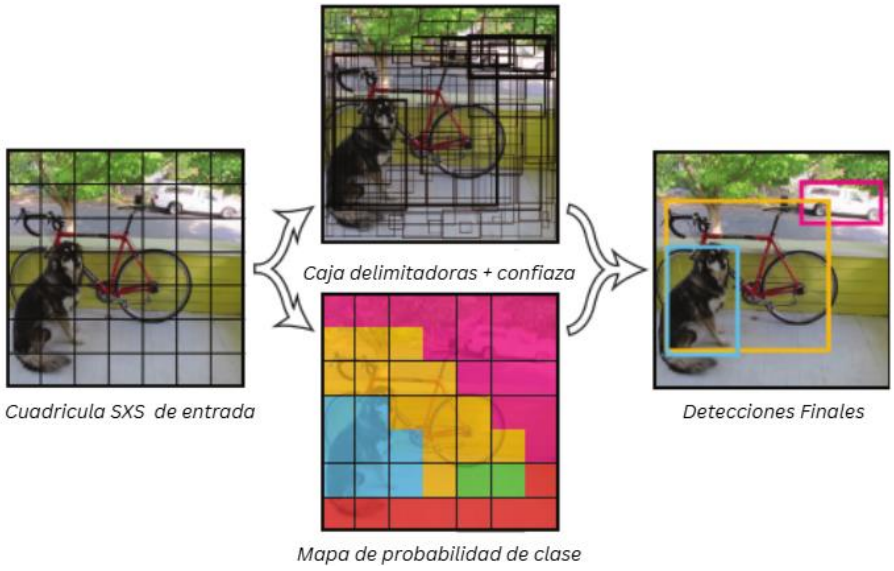



Figura 11. Funcionamiento de YOLO [39].

La supresión de no-máximos es una técnica de post-procesamiento usada en tareas de detección de objetos, en este procedimiento se reduce la cantidad de cajas delimitadoras propuestas las cuales se encuentran sobrepuestas, esto permite mejorar la calidad en la detección. Debido a que como en el caso de YOLO los modelos de detección generan varias propuestas de cuadros delimitadores alrededor de un objeto con diferentes valores de confianza. Por lo que NMS sirve como filtro, eliminando aquellas propuestas de recuadros los cuales son irrelevantes o redundantes y dejan aquel con mayor confianza, la Figura 12 nos muestra este proceso.


Figura 12. Non-Maximum Suppression (NMS) [40].

La arquitectura del modelo de YOLO se muestra en la Figura 13, las primeras capas convolucionales extraen las características de la imagen mientras que las capas completamente conectadas predicen las probabilidades y las coordenadas. La red tiene 24 capas convolucionales seguidas de dos capas completamente conectadas, se usa una capa de reducción de 1x1 seguido de una capa convolucional de 3x3.

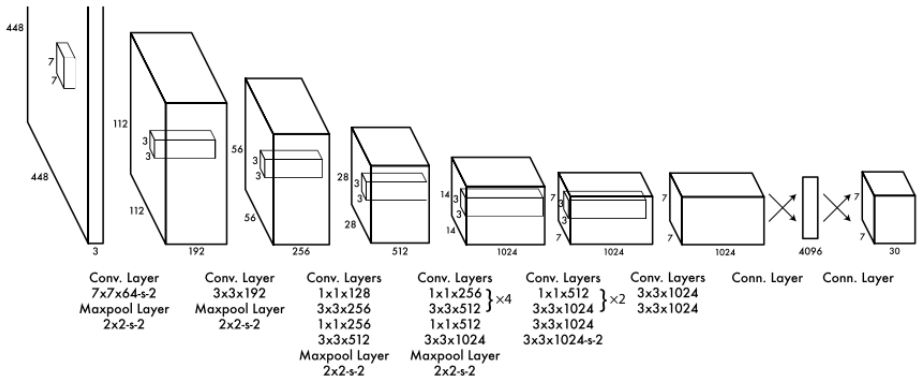



Figura 13. Arquitectura de YOLO [39].

## 2.15 Comparativa de modelos

En los sistemas de conducción autónoma o asistencia en la conducción los puntos a considerar en un modelo de detección de objetos es la precisión y una rápida repuesta del mismo ante la presencia de un objeto de interés. Esto debido a que en un entorno o aplicación real los modelos estarán expuestos a múltiples imágenes por segundo con baja latencia, por lo que el modelo debe ser capaz de procesarlas y dar una rápida respuesta. Las propuestas mostradas con anterioridad tienen diferente comportamiento en cuanto a precisión, velocidad y eficiencia.

En la Tabla 3 se presenta una comparativa general donde se abordan las diversas características de los modelos usados en tareas de detección de objetos.

| Criterio | Faster R-CNN | SSD | YOLO |
|---|---|---|---|
| Tipo de arquitectura | Dos etapas | Una etapa | Una etapa |
| Velocidad (FPS) | 5-7 | 22-59 | **60-150** |
| Precisión ($_{mAP@0.5, COCO}$) | **75-80%** | 72-77% | **77-83%** |
| Latencia | >120 ms | 46-60 ms | **< 20 ms** |
| Objetos Pequeños | **Excelente** | Bueno | Bueno |
| Escalabilidad (multiplataforma) | Limitada | Media | **Excelente** |
| Adecuado para tiempo real. | Poco | Parcialmente | **Alta** |
| Complejidad de entrenamiento | Alta | **Media** | **Media** |

Tabla 3. Comparación de los modelos

Un punto crítico dentro la conducción autónoma es la latencia de respuesta, debido a que un retraso de 100 ms en un vehículo que va a 80km/h equivale a recorrer prácticamente 2 metros sin recibir respuesta de parte del sistema, por lo cual la elección del modelo requiere de un equilibrio entre una alta precisión y una rápida respuesta de parte del mismo. El modelo Faster R-CNN tiene una alta precisión, pero es muy lento para aplicaciones en tiempo real. SSD tiene una mayor rapidez en la respuesta, pero no es tan preciso para objetos pequeños. YOLO por su parte tiene un

gran balance entre la velocidad la cual puede ser casi instantánea y una gran precisión en diversos escenarios [41].

YOLO tiene una mayor facilidad de escalabilidad lo cual permite que se pueda ejecutar en sistemas embebidos con menores recursos computacionales, esto se debe a que YOLO cuenta con versiones más pequeñas del modelo como YOLO5nano o YOLO8nano lo cual permite la transición a este tipo de sistemas embebidos, manteniendo alrededor de 30FPS de inferencia, mientras que los otros modelos requieren mayor consumo computacional y una mejor optimización para sistemas embebidos [42].

Las versiones de YOLO(V5-V8) indican que mantienen una buena estabilidad en sus resultados bajo diversas condiciones de luz, variabilidad en el clima y en la obstrucción parcial de los objetos. Por lo que YOLO es una gran herramienta en cuanto a tareas de detección de objetos para conducción en áreas urbanas.[42]

## 2.16 Etiquetado

El proceso para entrenar un modelo supervisado de ML en el área de visión por computadora requiere una gran cantidad de información (imágenes) etiquetada, este proceso es requiera muchas veces la mayor cantidad de recursos en su desarrollo (tiempo y dinero), así como también se precisa de una muy alta calidad para obtener los mejores resultados [43].

El etiquetado de imágenes consiste en la construcción de un mapeo de características visuales con etiquetas semánticas el cual nos describe lo que se encuentra dentro de la imagen. Dentro de la literatura se usan dos términos de forma indistintamente para referirse a las etiquetas, la primera es etiqueta y el otro termino es anotación [43].

El proceso de anotación de imágenes contiene los siguientes cinco pasos: colección del material, etiquetado, post-procesamiento, evaluación de calidad y exportación de los datos. En el primer paso se recopilan los datos a etiquetar (imágenes o videos), el segundo paso consiste en etiquetar la imagen dependiendo del objetivo a realizar, como tercer paso se suele hacer procesamiento de imágenes en caso de ser necesario, y seguido se somete a una evaluación para ver la calidad del etiquetado y de la imagen, en la evaluación se busca que lo realice una persona

externa la cual no haya etiquetado la imagen, y por último se exporta las etiquetas en el formato requerido [43].

Muchos de las herramientas de etiquetado permiten el etiquetado con recuadros delimitadores (bounding boxes) en el cual se encierra el objeto de interés, pero también hay otras formas como los polígonos, segmentos/mascaras o puntos y líneas como se observa en la Figura 14 [43].


Figura 14. Tipos de etiquetado espacial [43].

Existen diversos procesos de etiquetado, el manual es aquel donde una persona va creando las etiquetas una a una. El proceso semiautomático se apoya de un algoritmo el cual hace el etiquetado y posteriormente el proceso es corregido o avalado por una persona. El etiquetado automático se deja al algoritmo realizar el etiquetado de forma independiente, aquí se puede correr el riesgo de algún error de etiquetado. Mientras que el etiquetado interactivo es un ciclo donde constantemente se le da retroalimentación al algoritmo y así mejore al momento de etiquetar. Esto se visualiza en la Figura 15.

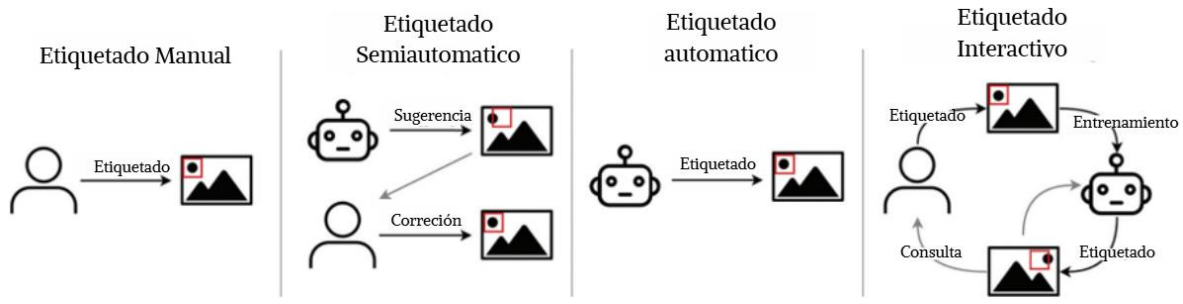



Figura 15. Procesos de etiquetado [43].

## 2.17 Tareas de visión por computadora

El área de visión por computadora tiene diversas aplicaciones o labores y cada una tiene objetivos diferentes y por ende sus metodologías varían, por lo cual a continuación se describirá cada una de ellas y lo que conlleva.

### 2.17.1 Reconocimiento en imágenes

El reconocimiento en imágenes consiste en determinar si el objeto es el objetivo o no. Es decir, si el objeto se encuentra en la imagen del video o no. La ecuación (2) nos describe esto y la Figura 16 visualiza esto [44].

$$Y \in \{\text{objeto o no} - \text{objeto}\} \tag{2}$$

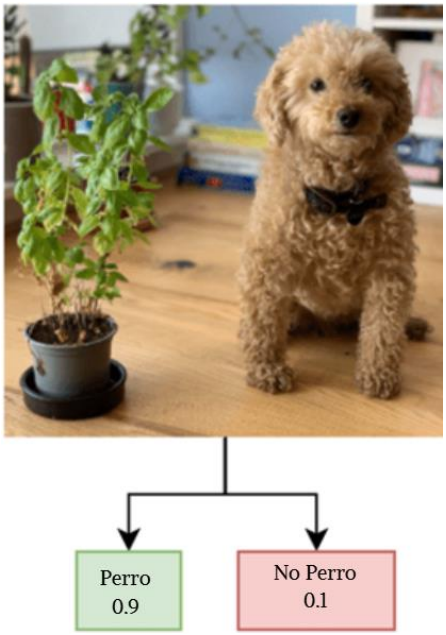



Figura 16. Tarea de detección [45].

### 2.17.2 Clasificación de imágenes

Para las tareas de clasificación lo que se busca es determinar una clase y buscar si el objeto dentro de la imagen pertenece a alguna de las clases creadas. La ecuación (3) describe esto y la Figura 17 lo muestra [44], [46].

$$Y \in \{\text{clase A}, \text{clase B}, \text{clase C}, \text{no} - \text{objeto} \tag{3}$$

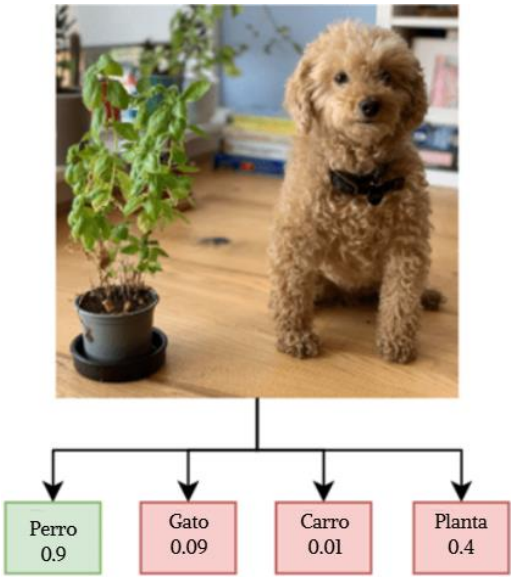


Figura 17. Tarea de clasificación [45].

### 2.17.3 Localización

Lo que se busca en este tipo de tareas es la locación del objeto de interés, la salida es una matriz de probabilidades de cada característica correspondiente al objeto. La ecuación (4) describe dicha matriz [44].

$$\mathrm{Y} = \{a_{i,j}\}_{mxn} \tag{4}$$

### 2.17.4 Detección o identificación de objetos

La tarea de detección consiste en buscar el objeto objetivo dentro de la imagen ubicándolo con una caja delimitadora, este proceso es un conjunto entre aspectos de clasificación y localización ya que una vez localizado el objeto este se asocia con una clase definida. La ecuación (5) es una representación de la colección de cajas delimitadoras y la Figura 18 lo ejemplifica [44], [46].

$$\mathrm{Y} = \{y_1, y_2, y_3 \cdots y_n | y_i = (x, y, w, h)\} \tag{5}$$

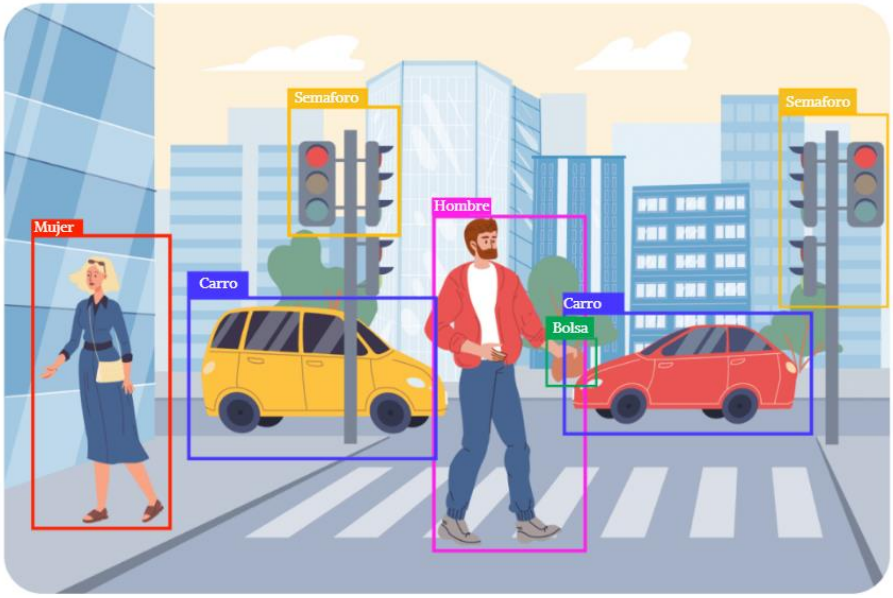



Figura 18. Tarea de detención [47].

## 2.18 Métricas

Las métricas de evaluación dentro de las tareas de detección sirven para poder cuantificar el desempeño del modelo, esto nos permite saber que tan bien detecta el sistema de forma numérica y no depender de nuestra percepción, nos indica que tan cerca es la predicción hecha por el modelo y la verdadera etiqueta. La métrica mayormente usa dentro del campo es AP (Average Presicion) o mAP (mean Average Presicion) la cual es la precisión media, pero para poder entender a qué se refiere es necesario comprender los siguientes conceptos [48], [49], [50].

Verdadero Positivo (TP): Cuando la predicción del modelo coincide con la etiqueta verdadera establecida.

Falso Positivo (FP): Cuando la detección es incorrecta de un objeto inexistente o cuando realiza una detección en un lugar equivocado.

Falso Negativo (FN): Cuando el modelo indica que no existe el objeto, pero en realidad si esta. La Tabla 4 nos resume lo mencionado [48].

| | Etiqueta verdadera | Etiqueta Falsa |
|---|---|---|
| Predicción verdadera | Verdadero Positivo | Falso Positivo |
| Predicción falsa | Falso Negativo | Verdadero Negativo |

Tabla 4. Definiciones de TP, FP, FN y TN [49].

Nótese que en las definiciones se omitió los verdaderos negativos, esto debido a que en el contexto de detección de objetos esto no aplica ya que habría un gran número de cuadros delimitadores que no deberías detectarse [48].

Para determinar cuándo es un TP, FP, FN o TN es necesario establecer a que se refiere cuando se habla de “detección correcta” o “detección Incorrecta” [48]. Esto se puede establecer con la intersección sobre la unión (IoU), el cual es una métrica basada en el índice de Jaccard, un coeficiente de similitud entre dos conjuntos de datos (etiquetas/anotaciones). El IoU corresponde al área de superposición (intersección) entre el área predicha $B_p$ y la anotación verdadera $B_{pt}$ y dividido entre el área de su unión, esto se describe en la ecuación (6) y en la Figura 19 se encuentra una ilustración de la ecuación [50].

$$J(B_p, B_{gt}) = IoU = \frac{area(B_p \cap B_{gt})}{area(B_p \cup B_{gt})} \tag{6}$$

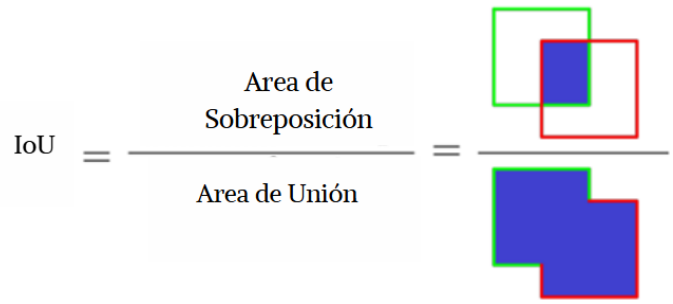



Figura 19. Ilustración de Intersección sobre Unión (IoU) [50].

Cuando el valor de IoU = 1 se dice considera que es una detección perfecta, mientras que es cuando IoU = 0 es una detección nula. Por lo cual cuanto más cercano se IoU a 1 la detección es mejor. Lo que se suele hacer es ajustar un límite o umbral *t* para poder definir cuando ya es una detección acertada. Generalmente este valor suele ser *t = 0.5,* por lo cual si el IoU ≥ *t* la detección es considerada correcta [48], [50].

La precisión (P) es la habilidad del modelo de identificar objetos relevantes, es el porcentaje de predicción correctas. Por su parte la Recuperación (R) es la capacidad del modelo de recuperar todos los casos relevantes, el porcentaje de predicciones positivas correctas entre todas las etiquetas verdaderas dadas. Las ecuaciones (7) y (8) nos muestran el cálculo de estas métricas [48].

$$P = \frac{TP}{TP + FP} = \frac{TP}{Todas\ las\ detecciones} \tag{7}$$

$$R = \frac{TP}{TP + FN} = \frac{TP}{Todas\ las\ verdades} \tag{8}$$

También existen otro tipo de evaluaciones, las cuales son variantes de AP, estas se avalúan con un valor diferentes y varían de rango. Generalmente se pueden calcular para 10 IoUs entre el rango de 0.5 a 0.95 con pasos de un 5%. O simplemente se puede reportar con cierto valor como son mAP50, mAP75 o mAP50:95 [48].

## 2.19 Aumento de datos

Muchas veces uno de los principales problemas dentro del DL es la limitante cantidad de datos para poder entrenar los modelos, si se mejora la cantidad y diversidad de los datos los resultados mejoran, por lo cual el aumento de datos se ha convertido en algo prácticamente inevitable para desarrollar modelos de inteligencia artificial [51], [52].

Lo que busca el aumento de datos es aumentar la cantidad de muestras y la variedad con las que nuestro modelo aprenda para que de esa forma tenga un mejor desempeño al momento de ser probado. Una de las principales razones por la cuales se usa el aumento de datos es debido a que es posible que no se tenga la suficiente cantidad de datos y con poca diversidad y el hecho de obtener nuevos datos puede ser una tarea de alto costo o demandante. A continuación, se describen las principales técnicas de aumento de datos en imágenes, la Figura 20 muestra un resumen de técnicas de aumento de datos utilizadas en YOLOv8.

Voltear. La imagen se voltea vertical u horizontalmente.

Rotar. Se puede cambiar el ángulo de inclinación de la imagen.

Escalar. Se modifica el tamaño de la imagen ya sea en aumento o disminución.

Ruido. Se puede inyectar diversos tipos de ruido a la imagen (ruido Gausiano, efecto lluvia, neblina, entre otros.)

Espacio de Color. Se puede modificar los canales de color.

Contraste. Se puede cambiar el contraste de la imagen y brillo de la imagen.

Enfoque. Se puede variar la nitidez de la imagen.

Traslación. Mover la imagen vertical u horizontalmente.

Recortar. Recortar una zona de la imagen.

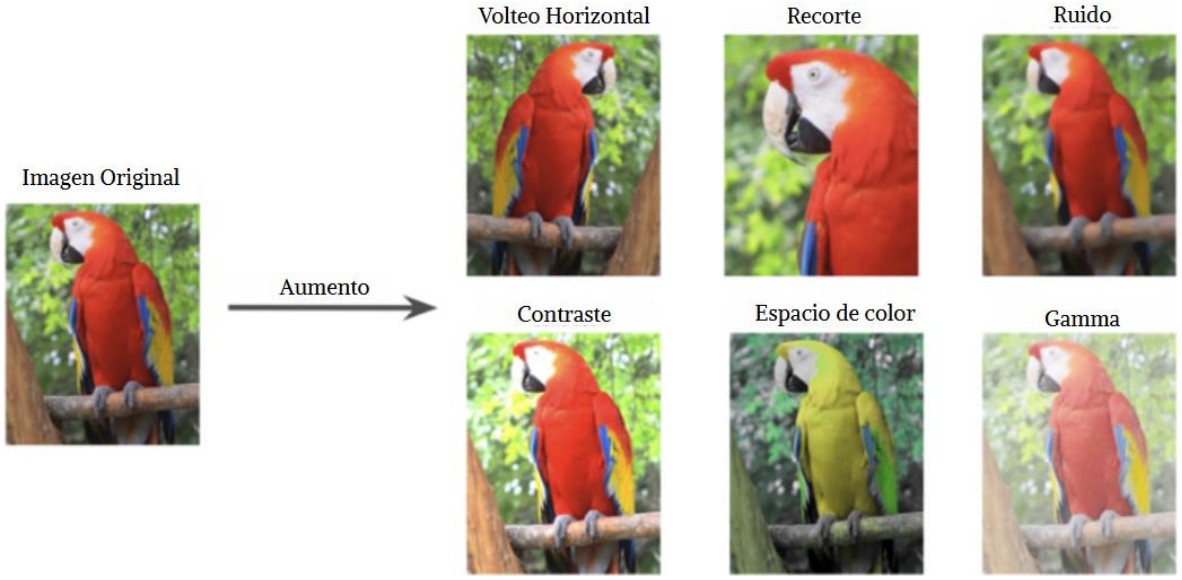



Figura 20. Aumento de datos usados en YOLOv8 [53].

## 2.20 Hiperparámetros

Casi todos los algoritmos de modelos de aprendizaje cuentan con un conjunto de Hiperparámetros de ajuste a los datos. Estos se deben de distinguir entre de los parámetros internos del modelo, aquellos como pesos que se actualizan de forma adaptativa para cada problema. Los Hiperparámetros de las redes neuronales suelen especificar la arquitectura de dicha red, como número y tipo de capas, número y tipos de nodos, entre otros [54]. Son variables que controlan diferentes aspectos del entrenamiento, estas se pueden controlar mientras que los parámetros internos del

modelo se ajustan en medio del entramiento. El objetivo de los hiperpárametros de optimización es encontrar aquellas variables que obtengan la mejor salida o rendimiento del modelo, es decir disminuir en la menor cantidad la tasa de error. Los hiperpárametros más comunes son [55]:

- Tasa de aprendizaje
- Tamaño del lote
- Épocas

# CAPITULO 3. METODOLOGIA

Para la detección de concentraciones de agua, se presenta la metodología mostrada en la Figura 21 y posterior a ello se describirá cada una de las etapas.

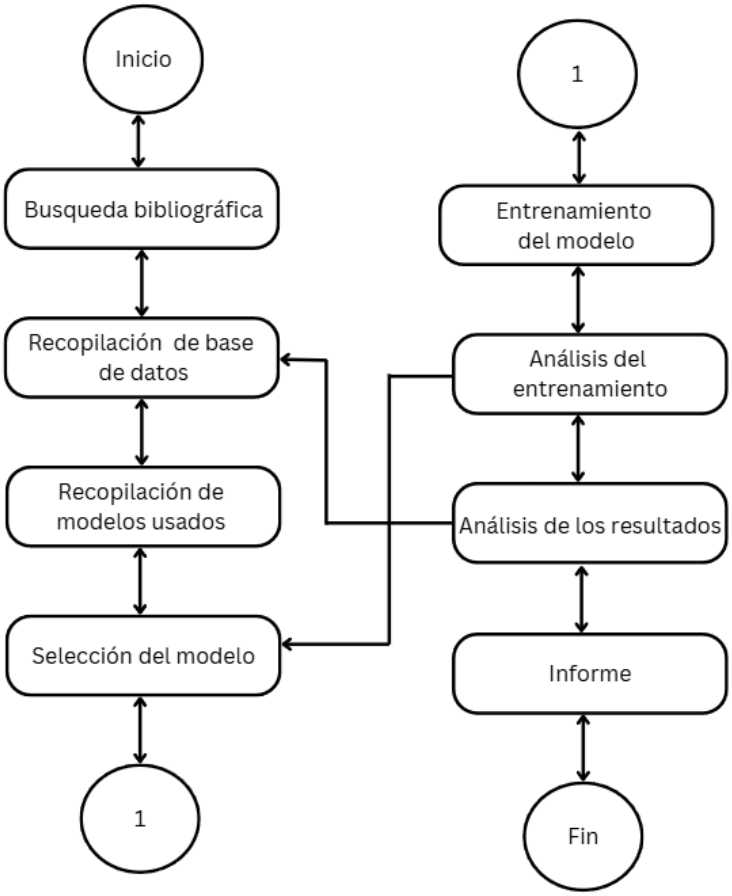



Figura 21. Diagrama de flujo de la metodología.

## 3.1 Búsqueda bibliográfica

El primer paso para la resolución de este problema consistió en la recopilación de trabajos relacionados con la problemática a resolver, esta información fue mostrada anteriormente en el primer capítulo donde se desglosa cada uno de los trabajos relacionados. Nuestra metodología se basas en métodos de IA la cual se explicará en breves instantes.

## 3.2 Recopilación de base de datos

Para la obtención de la base de datos se consultó los diversos trabajos que se encuentran en el estado del arte, dentro de los cuales en la mayoría de los trabajos

los autores creaban su propia base de datos, sin embargo, algunos trabajos disponían de un conjunto de datos en común, dicho conjunto lleva por nombre "*puddle100*" [17], en esta base de datos los autores ampliaron un antiguo conjunto de datos [56] para lo cual montaron un cámara ZED la cual sirve para capturar de forma más profunda la información en movimiento y del ambiente, esta cámara tiene dos lentes la cual imita los ojos humanos y la visión estereotípica, el conjunto de imágenes está dividido en segmento dentro de ruta (ONR) que cuenta con 357 capturas (frames) y fuera de ruta (OFR) con 628 capturas, dentro de la imágenes de ruta se encuentra una carretera de asfalto y carros en movimiento, mientras que las que no están en ruta es un camino de terracería mojado con obstáculos como árboles, montículos, materiales de construcción y vallas, las tomas obtenidas fueron en un día en particular.

Se buscaba utilizar este conjunto de imágenes para realizar pruebas, sin embargo, no se logró obtener dichas imágenes, en el repositorio del trabajo se encuentra un enlace de descarga y no fue posible acceder a este en la Figura 22 se muestra el enlace de descarga para las imágenes y la pagina que carga al intentar ingresar.

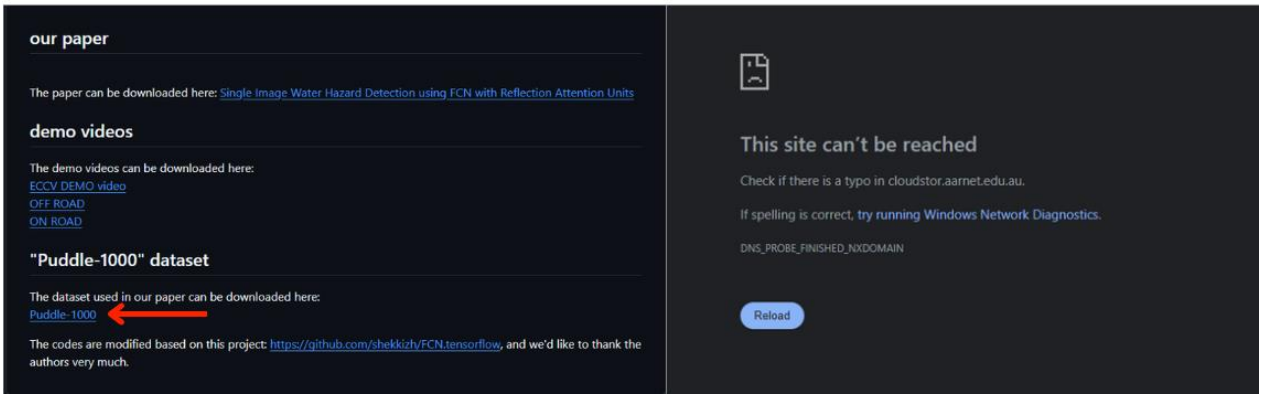



Figura 22. Repositorio del conjunto de datos 'puddle-1000''.

Debido a que no fue posible el uso de bases de datos para el desarrollo de este trabajo, se creó una base de datos propia, a partir de videos existentes previos al inicio del proyecto. Estos videos son de segmentos de ruta en el cual se conducía de punto A al punto B, sin tomar en cuenta la condición del día ni la ruta a seguir. Podía ser un día despejado, por la mañana, tarde o noche, podía estar nublado o con lluvia. El dispositivo de captura se muestra en la Figura 23.

Figura 23. Dispositivo de captura

Se extrajo imágenes de los videos, de las cuales un grupo contienen muestras positivas (con charco) y muestras negativas (sin charco). En la Figura 24 y Figura 25 se visualiza una muestra de este tipo de imágenes, donde se busca obtener la mayor similitud en cuanto a tipo de luz, camino y ambiente en las imágenes que tienen cúmulos de agua y las que no.


Figura 24. Imagen con cúmulo de agua.

Figura 25. Imagen sin cúmulo de agua.

Las imágenes recopiladas son diversas, ya que estas se encuentran en distintas condiciones de luz y distintos tipos de camino que pueden ser estructurado (calles pavimentadas) y no estructurado (calles sin pavimentar o en malas condiciones). En la Figura 26 se puede ver la variedad de condiciones en las que se obtuvieron las muestras.

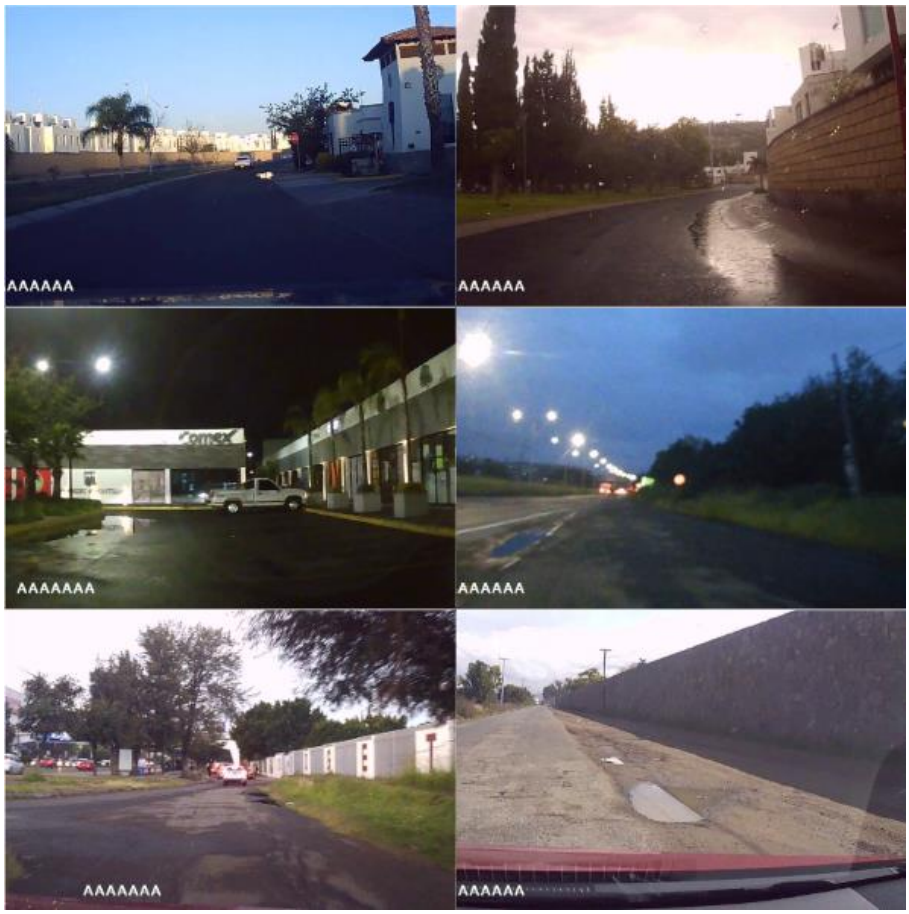


Figura 26. Condiciones de luz en el conjunto de datos

El conjunto de datos creado consiste de 2866 imágenes de las cuales 1320 no contienen cúmulos de agua y las 1546 imágenes restante tienen cúmulos de agua en ellas. Estas imágenes pertenecen a la ciudad de Querétaro y Guanajuato (México).

Una vez que se agrupó los dos tipos de imagen se procedió a realizar las anotaciones o etiquetado de imágenes, para realizar este paso se utilizó la plataforma de etiquetado CVAT [57], la cual tiene distintas herramientas que ayudan a realizar de forma más practica y precisa, la Figura 27 muestra el entorno de etiquetado que utilizado para la realizar las anotaciones pertinentes en las imágenes.

Cuando se realizó el etiquetado de todas las imágenes, se procedió a realizar su descarga de las etiquetas en formato “.txt” las cuales contienen las coordenadas del objeto de interés. Para las imágenes que no tienen charco se genera una etiqueta, pero en este caso el archivo “.txt” se encuentra vació, esto indica que la imagen no contiene el objeto de interés.

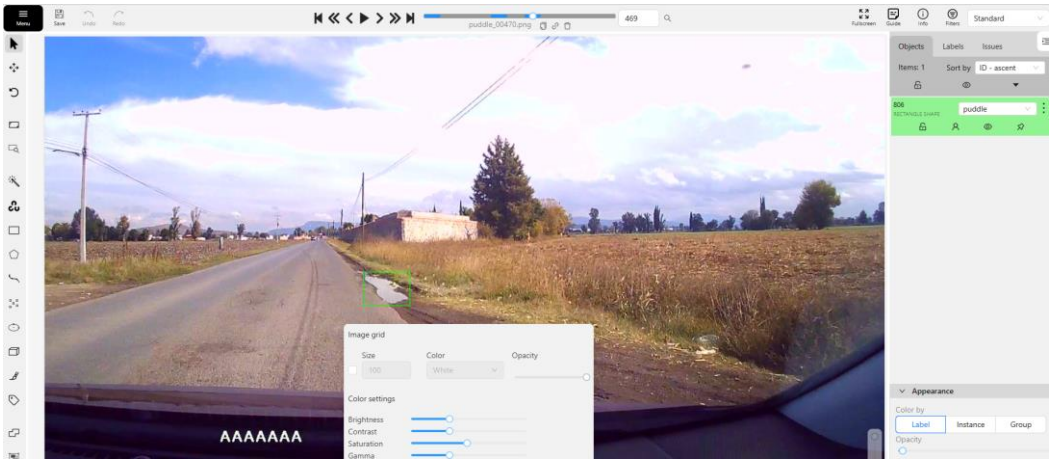



Figura 27. Interfaz de etiquetado de CVAT para etiquetado de imágenes.

El siguiente paso que se realizó fue la creación de los directorios y la división del conjunto de datos. Para poder dividir la base de datos en conjunto de entrenamiento, validación y prueba, se creó cuatro directorios uno para las imágenes que no contienen charcos, otro para las etiquetas vacías, las imágenes con charco y las etiquetas de los charcos. Estos directorios servirán para crear los tres conjuntos mencionados anteriormente. Dentro de estos conjuntos se encontrarán en cada uno de ellos imágenes con sin charcos, etiquetas vacías, imágenes con charco y las etiquetas. Las etiquetas deben contener el mismo nombre que la imagen para el modelo asocie la etiqueta con la imagen. Esto se puede ver de forma visual en la Figura 28 la cual es una representación gráfica de la estructura del directorio.

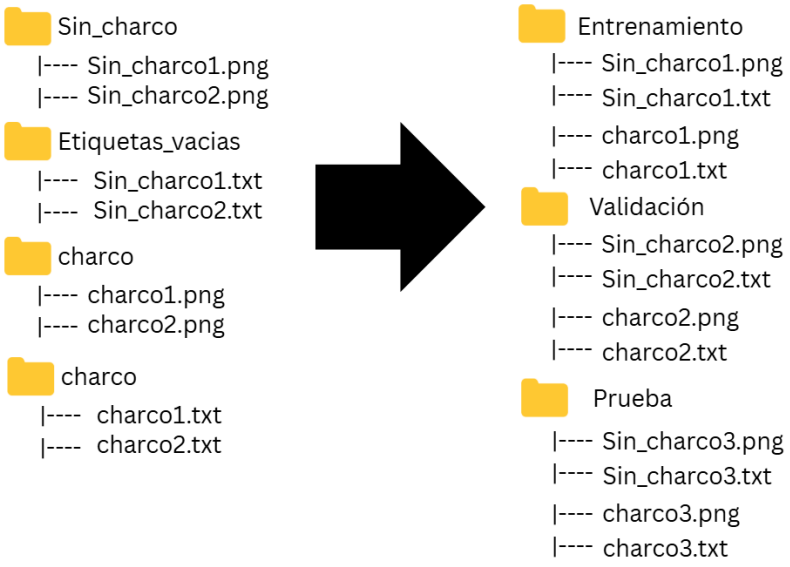



Figura 28. Estructura del conjunto de datos

## 3.3 Recopilación de modelos usados

Los modelos usados con mayor frecuencia en trabajos relacionados con la detección de charcos y de objetos en general son faster R-CNN, SDD y YOLO. YOLO tiene como característica el procesamiento de imágenes a alta velocidad y la capacidad de respuesta en tiempo real, lo que lo hace destacar en tareas que se requiera una rápida respuesta, con las constantes actualizaciones que ha tenido el modelo a mejorar su rendimiento y robustes en diversas condiciones de luz y ruido. Faste R-CNN es una mejora al algoritmo que lleva por nombre R-CNN y fast R-CNN, su principal aporte consta en la inclusión de la red de propuesta de región (RPN) lo que permite agilizar las propuestas de región y mejorar la velocidad de detección. También enfocado en la velocidad del procesado de imágenes SDD utiliza un mapa de características de multiresolución para detectar objetos, logrando así una alta precisión a gran velocidad [58].

## 3.4 Selección del modelo

Debido a en la aplicación del problema, lo que se busca es un modelo que tenga una rapidez en la respuesta de la detección en tiempo real y un buen desempeño. Según Nataliya Bilous [58] pese a que otras redes neuronales obtienen mejores resultados en ciertas métricas YOLO las supera en todas, mostrando los mejores resultados en mAP-0.88, F1-0.88 y FPS-48. Las características de este modelo lo convierten en la opción preferida para tareas de detección en tiempo real como el monitoreo de seguridad en vehículos autónomos. Además, que muestra que

el modelo es robusto para entornos donde el agua se refleja y las condiciones de luz son variadas. A partir de la cuarta versión se introdujeron varias optimizaciones para la mejora del rendimiento, como el aumento de datos, técnicas de regularización y estrategias avanzadas de entrenamiento [41]. YOLO es el modelo más eficaz para la detección de objetos en diversos entornos debido a su alta precisión, velocidad de procesamiento y adaptabilidad [58].

## 3.5 Entrenamiento del modelo

El entrenamiento del modelo se lleva a cabo en la plataforma Google colab [59], con el plan pro, que es un entorno de desarrollo en la nube, en el cual se pueden desarrollar proyectos de inteligencia artificial. En la Figura 29 se muestra la interfaz con la que se está trabajando.

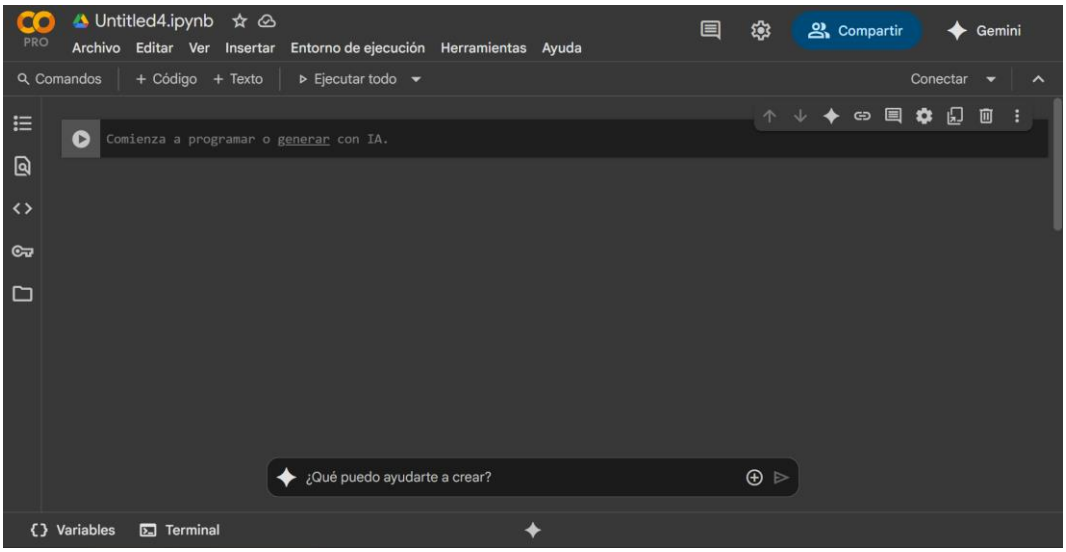



Figura 29. Interfaz de Google colab Pro.

El equipo de cómputo con el que se realizaron las pruebas tiene las siguientes características, Figura 30:

- CPU: Intel(R) Xenon (R) @ 2.20 GHz, 12 núcleos lógicos (6 físicos, 2 por núcleo)
- Memoria RAM: 52.96
- GPU: NVIDIA L4
- Almacenamiento disponible: 236 GB (39 GB usados, 198 GB libres).

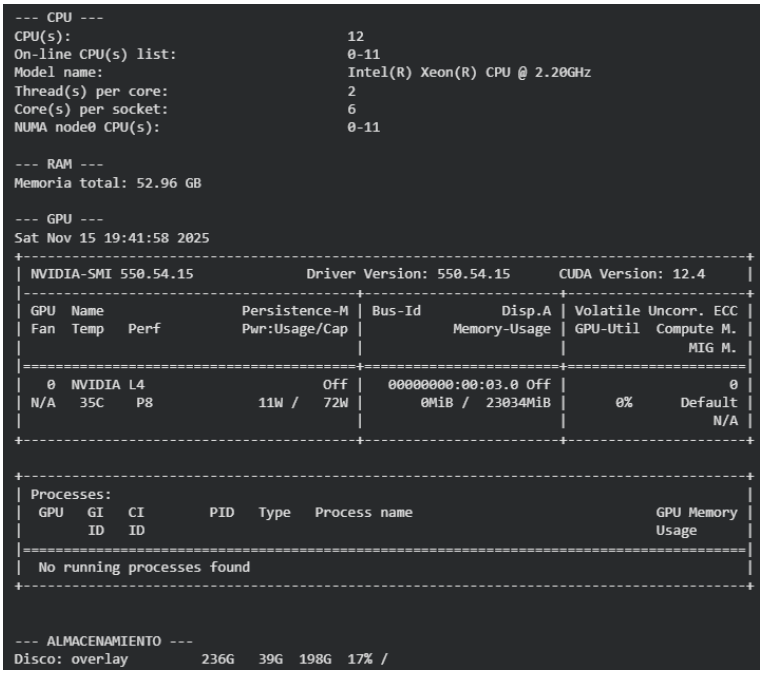



Figura 30. Características del equipo.

El entrenamiento se llevará a cabo bajo un esquema 70/15/15 [60], [61] esto quiere indicar que el 70% (2030 imágenes y etiquetas) del conjunto de datos se utilizará para el entrenamiento, un 15% (406 imágenes y etiquetas) será para la validación y el 15% (430 imágenes y etiquetas) restante para pruebas del modelo. El conjunto de entrenamiento consta de 2006 muestras. La Figura 31 es un esquema grafico de esta división.

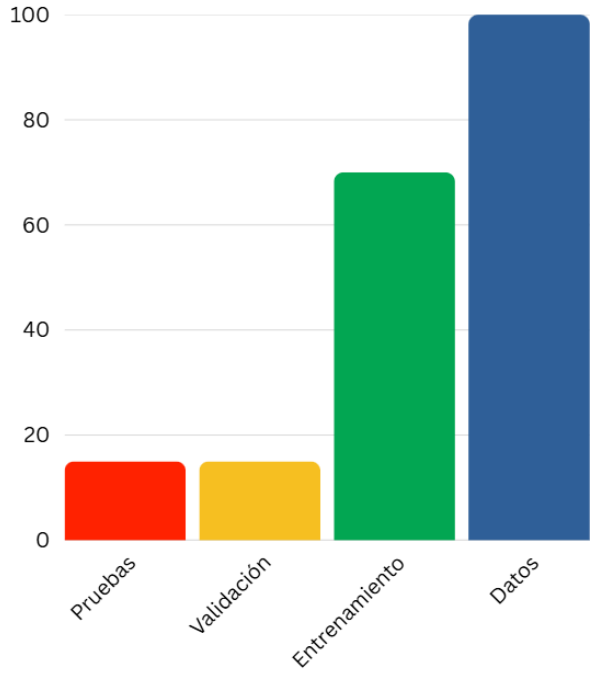



Figura 31. Esquema de entrenamiento.

Para obtener resultados más robustos y representativos del modelo, se implementó la validación cruzada K-fold, esta técnica permite evaluar el desempeño del modelo de una forma más confiable. En donde se divide el conjunto de datos en *K número* de subconjuntos, donde por cada *K* se entrena y se valida con diferentes tipos de muestra, esto permite que el modelo sea entrenado y validado con diferentes tipos de imágenes. La Figura 33 muestra de forma visual como la proporción 70/15/15 se respeta, lo único que se modifica es que en cada fold los datos de entrenamiento y validación cambian, mientras que el subconjunto de pruebas se mantiene igual.

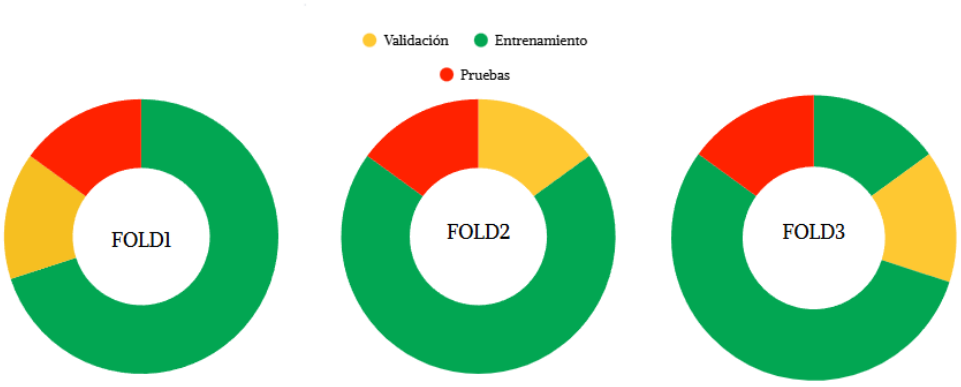



Figura 32. Validación K-fold.

El modelo a usar para el entrenamiento es YOLO, en sus versiones 5, 8 y 11. La versión 5 tiene 241 capas, 53,225,024 parámetros, 0 gradientes y 135.6 GFLOPS. Por otro lado, la versión 8 tiene 209 capas, 43,691,520 parámetros, 0 gradientes y 165.7 GFLOPS. Finalmente, la versión 11 tiene 257 capas, 25,372,160 parámetros, 0 gradientes y 87.6 GFLOPS. Estas versiones tienen mejoras en la detección de objetos pequeños, ocluidos y distantes en diversas condiciones de luz y de clima, además que la ultralytics mantiene soporte, documentación y las métricas son replicables [42].

## 3.6 Análisis del entrenamiento

El entrenamiento a realizar se llevará acabo usando la siguiente configuración de hiperparámetros para las 3 versiones de YOLO. La Figura 33 muestra una visualización de cómo se verían aplicados los hiperparámetros que permiten el aumento de datos, si bien no se pueden obtener estos aumentos directamente del modelo, se puede realizar una visualización de cómo se verían.

- Optimizador: AdamW
- Tasa de aprendizaje inicial: 0.0001
- Tasa de aprendizaje final: 0.001
- Caída de peso: 0.0005
- Épocas=150 [20]
- Lote: 32
- Tamaño de imagen: 640
- Aumento de datos:
    - hsv_h=0.03
    - hsv_s=0.7
    - hsv_v=0.4
    - grados=8
    - translación=0.15
    - escalado=0.35
    - shear=3.0
    - voltear=0.5
    - mosaico=0.5
    - mezclar=0.1
- pretrained=True
- single_cls=True

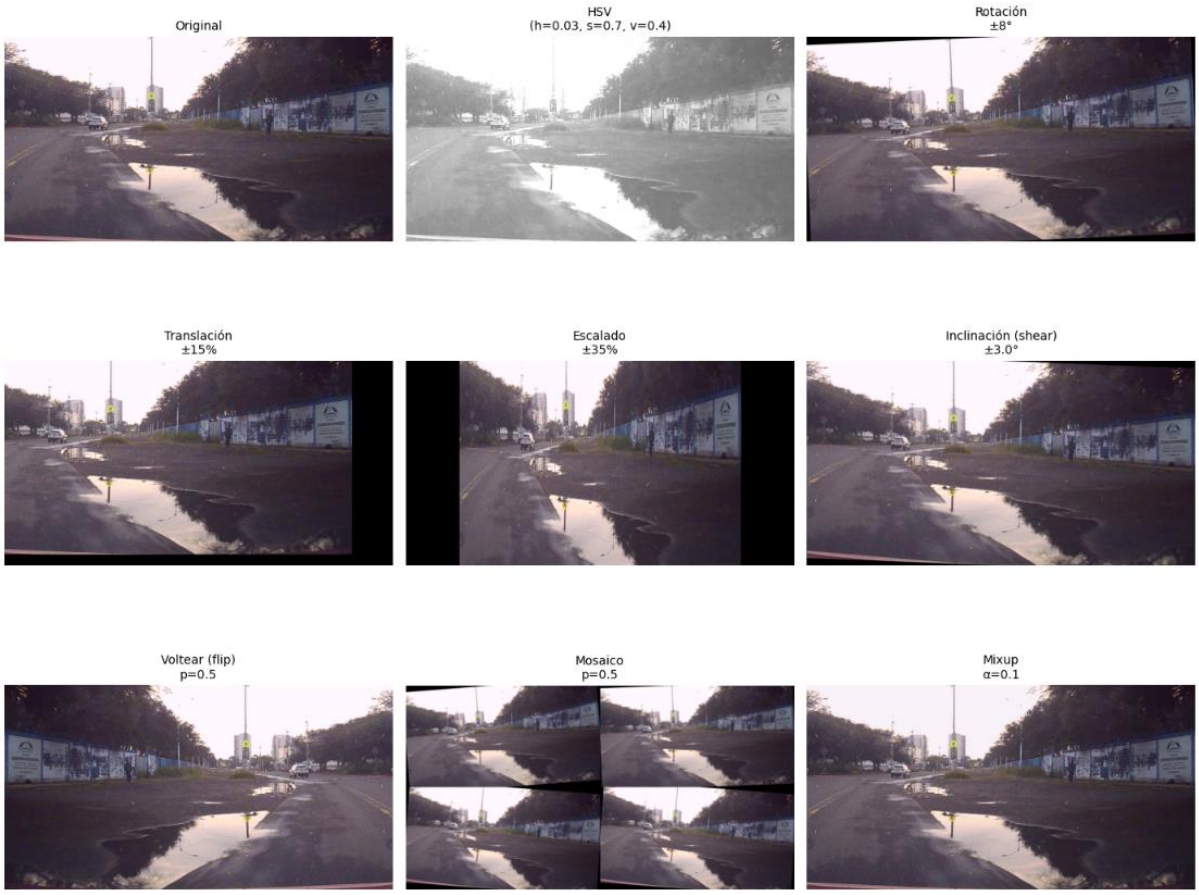



Figura 33. Aumento de datos

La figura 34 y 35 muestra el comportamiento del modelo en la versión 5 durante el entrenamiento, la figura 34 muestra la perdida durante el entrenamiento y las figura 35 muestras el desempeño de las métricas durante el proceso de entrenamiento.

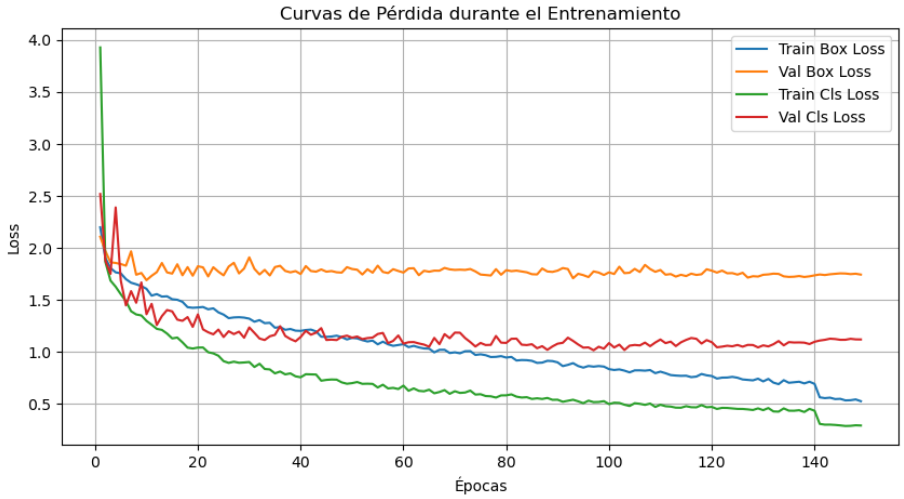



Figura 34. Perdida durante el entrenamiento en YOLOv5.

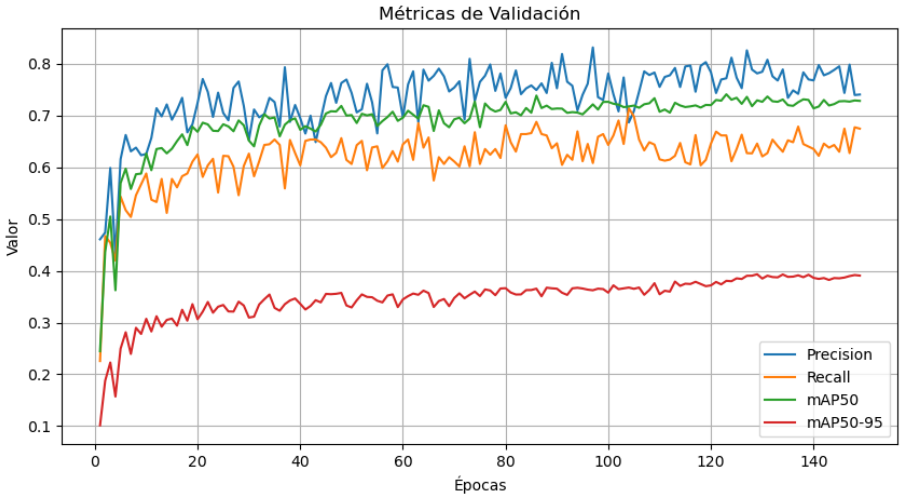



Figura 35. Métricas de validación en YOLOv5.

En la figura 36 se muestra la evolución de perdida durante el entrenamiento y en la figura 37 las métricas de validación para versión 8 de YOLO.

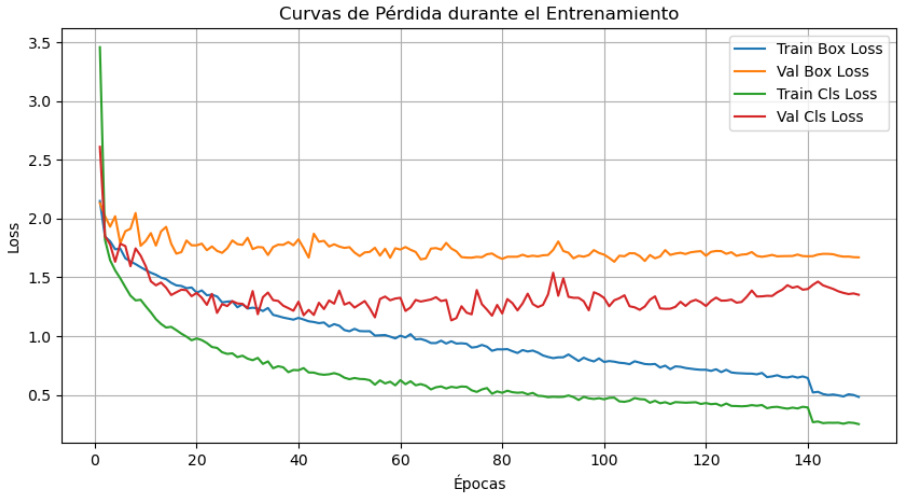



Figura 36. Perdida durante el entrenamiento en YOLOv8.

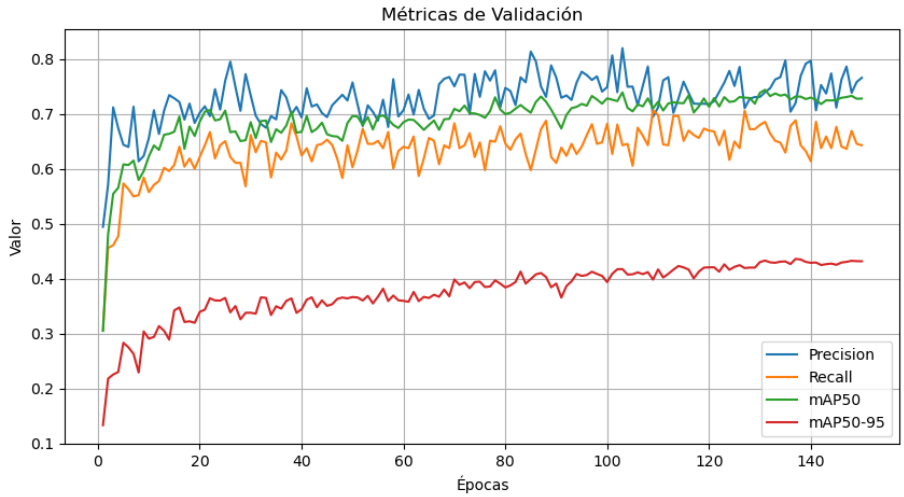



Figura 37. Métricas de validación en YOLOv8.

El desempeño de la versión 11 de YOLO se muestra en la figura 38 y 39, donde de igual forma se muestra la evolución de la perdida y las métricas obtenidas en el proceso de validación.

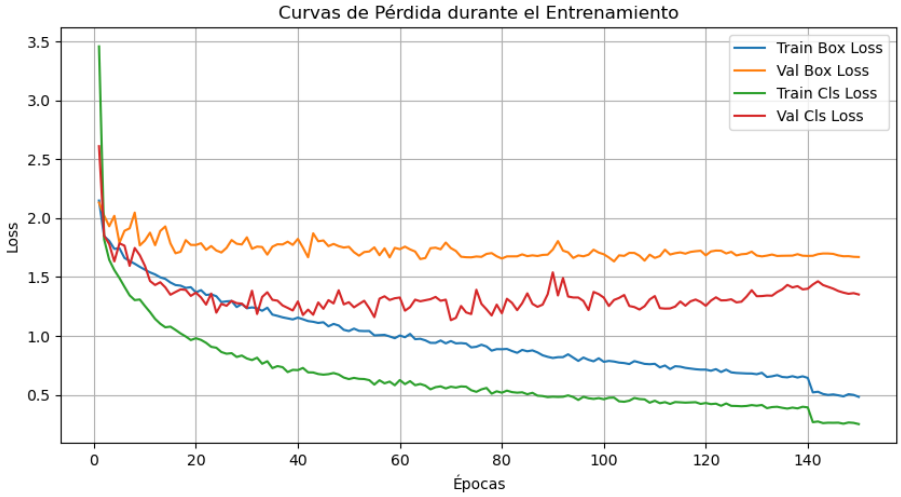



Figura 38. Perdida durante el entrenamiento en YOLOv11.

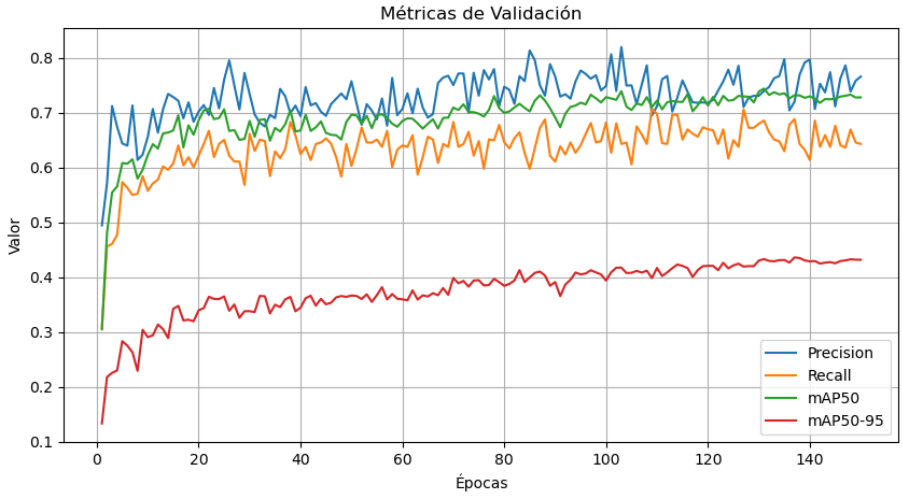



Figura 39. Métricas de validación en YOLOv11.

## 3.7 Análisis de los resultados

En base a los resultados obtenidos en el entrenamiento, se realizará un análisis para evaluar el desempeño del modelo entrenado. Todo este proceso se presenta en el capítulo 4.

# CAPITULO 4. RESULTADOS Y DISCUSIÓN

## 4.1 Evaluación de las métricas

Los resultados obtenidos con el modelo YOLO en las versiones 5, 8 y 11 se muestran en las tablas 5 a 7. Cada tabla muestra el resultado de cada una de las métricas usadas para evaluar el desempeño del algoritmo, para cada una de las versiones de YOLO para cada división (fold).

| Fold | Precisión | Recall | mAP50 | mAP50-95 |
|---|---|---|---|---|
| 1 | 69.85 | 66.87 | 69.15 | 38.25 |
| 2 | **78.79** | 61.01 | 68.60 | 37.18 |
| 3 | 76.53 | 68.13 | **75.17** | 42.86 |
| 4 | 76.32 | 68.40 | 74.18 | **42.95** |
| 5 | 73.61 | 66.91 | 73.44 | 38.41 |
| 6 | 74.09 | **69.02** | 73.77 | 42.85 |

Tabla 5. Resultados de YOLOv5

| Fold | Precisión | Recall | mAP50 | mAP50-95 |
|---|---|---|---|---|
| 1 | **81.02** | 66.91 | **76.39** | **44.87** |
| 2 | 76.79 | 62.55 | 70.62 | 36.20 |
| 3 | 78.22 | 69.72 | 72.34 | 39.74 |
| 4 | 71.07 | **69.90** | 75.16 | 40.04 |
| 5 | 74.70 | 65.42 | 69.98 | 37.94 |
| 6 | 75.85 | 67.16 | 73.01 | 39.41 |

Tabla 6. Resultado de YOLOv8

| Fold | Precisión | Recall | mAP50 | mAP50-95 |
|---|---|---|---|---|
| 1 | **79.18** | 66.23 | 73.51 | **42.49** |
| 2 | 74.13 | 64.87 | 70.85 | 36.56 |
| 3 | 75.90 | 67.91 | 73.86 | 39.85 |
| 4 | 77.19 | 67.41 | **74.36** | 42.20 |
| 5 | 76.81 | **70.06** | 73.64 | 38.76 |
| 6 | 72.96 | 68.48 | 73.35 | 39.30 |

Tabla 7. Resultados de YOLOv11

## 4.2 Comparativa estadística

En la tabla 8 se muestra una comparativa de los resultados obtenidos en las tres versiones, en la cual se reporta la media obtenida y la desviación estándar de los valores reportados en las tablas anteriores.

| Versión | Precisión | Recall | mAP50 | mAP50-95 |
|---|---|---|---|---|
| 5 | 76.05 ± 2.23 | 67.50 ± 1.80 | 72.39± 2.78 | 40.42 ± 2.74 |
| 8 | 76.28 ± 3.36 | 66.95 ± 2.76 | 72.92 ± 2.51 | 39.70 ± 2.91 |
| 11 | 74.87 ± 3.09 | 66.73 ± 2.93 | 73.27 ± 1.23 | 39.87 ± 2.23 |

Tabla 8. Resultados de los modelos entrenados considerando media y desviación estándar

Como se puede observar en la tabla 8, las tres versiones de YOLO tienen un rendimiento similar, en todos los casos se presenta un buen rendimiento en la tarea de detección de cúmulos de agua. El modelo de la versión 5 y 8 muestran una leve mejoría en la precisión y recuperación con respecto a la versión 11, pero en todas las versiones se garantiza una alta fiabilidad de detecciones positivas y la capacidad de identificar objetos reales en las imágenes, lo cual permite que el modelo sepa reconocer los cúmulos de agua. Mientras que la versión 11 tiene un rendimiento ligeramente mejor en las métricas de mAP, lo cual nos indica que es un poco mejor en la detección general del objeto y la localización del mismo.

## 4.3 Resultados de detección en las imágenes

En la Figura 40 se muestra algunas detecciones realizadas por la versión 5 de YOLO bajo diversas condiciones de luz.

Los ejemplos de las predicciones hechas por la versión 8 se presentan en la figura 41.

De igual manera en la figura 42 se muestran las predicciones realizadas por la versión 11, en todas se está mostrando las mismas imágenes para tener una comparativa igual entre las tres versiones.


Figura 40. Detección de YOLOV5


Figura 41. Detección de YOLOV8.


Figura 42. Detección de YOLOV11

### 4.4 Comparación entre versiones

En la figura 43 se compara el rendimiento de las tres versiones en un día con una buena iluminación (a), (b) y (c), las tres versiones tienen un muy desempeño en detectar el cumulo de agua. Mientras que en los puntos (d), (f) y (e) se muestra el desempeño en un día nublado, para el cual de igual manera las tres versiones tienen un buen desempeño.

En la figura 44 se hace una comparativa del rendimiento bajo una condición de luz diferente, que sería un atardecer, en donde la luz incide de tal forma que el cumulo de agua cambia ligeramente su aspecto (a), (b) y (c), para los siguientes incisos que serían (d), (e) y (f) las imágenes se encuentran en un medio urbano y con cúmulos más pequeños esparcidos por la calle. Las tres versiones cuentan con un buen desempeño, pero la versión 11 en ciertos cúmulos se ajusta y detecta mejor aquellos que son más pequeños.


Figura 43. Desempeño del modelo en un día soleado y nublado.

(a) Versión 5, (b) versión 8 y (c) versión 11

(d) Versión 5, (e) versión 8 y (f) versión 11

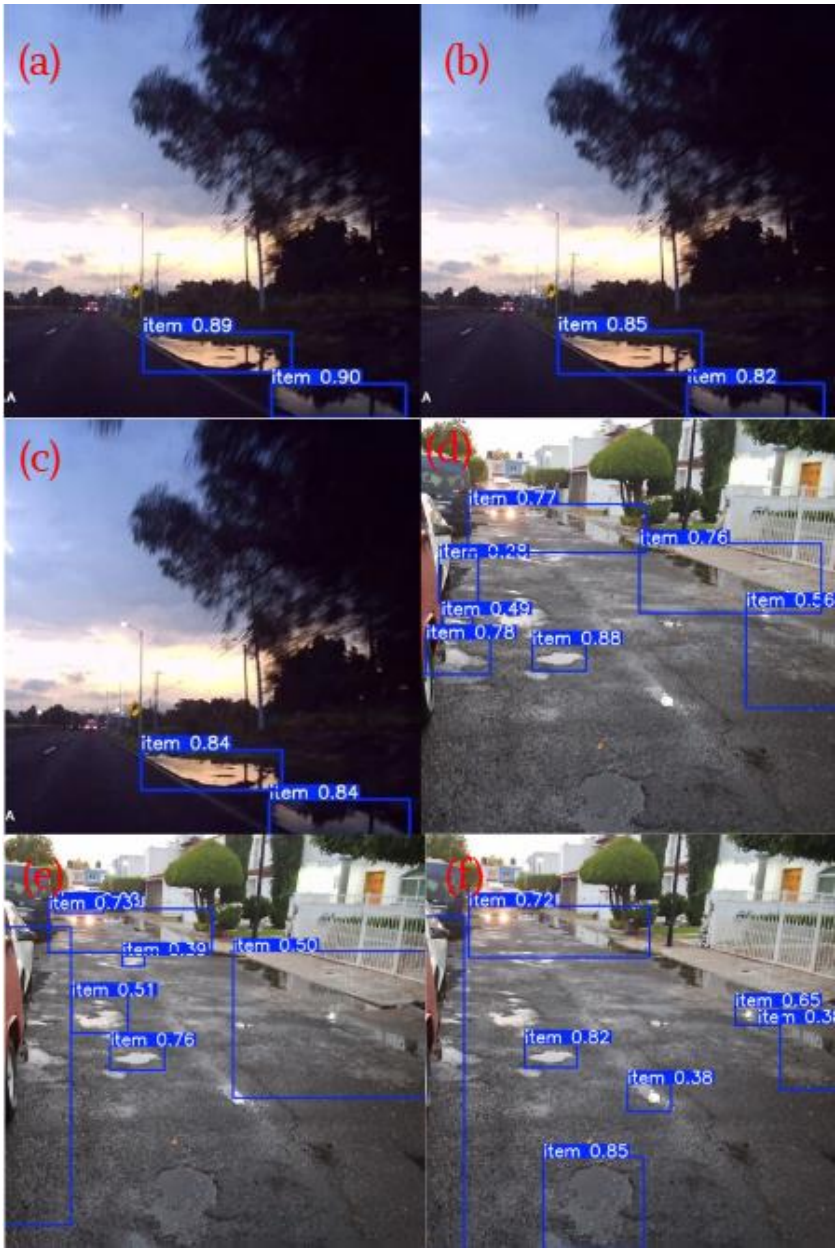



Figura 44. Desempeño del modelo en un atardecer y medio urbano.
(a) Versión 5, (b) versión 8 y (c) versión 11
(d) Versión 5, (e) versión 8 y (f) versión 11

La detección de cúmulos de agua es una tarea compleja, debido a que la imagen tomada al charco puede tener variaciones dependiendo de la luz que le incida y el ángulo en que se observa. Si bien el modelo presentado tiene un buen rendimiento en diversas condiciones de luz, tiene ciertas fallas en las cuales no logra detectar ciertas acumulaciones de agua. En la figura 45 se muestra como el modelo falla al no detectar algunas de estas acumulaciones, estas se encuentran encerradas con verde.


Figura 45. Detecciones faltantes con YOLO v8.

También el modelo realiza, aunque muy pocas veces detecciones en imágenes que no contienen cúmulos de agua, en la figura 46 se llega a visualizar esto, esto suele ocurrir con objetos que en ocasiones llegan a reflejar luz como un charco, en ambientes oscuros o zonas que tienen el camino mojado, pero no necesariamente se encuentra encharca la zona.


Figura 46. Detecciones erróneas.

## 4.5 Comparación de desempeño

En la tabla 9 se muestra un resumen comparativo de los resultados obtenidos en este trabajo y trabajos similares presentados en el estado del arte. Los resultados obtenidos son similares a los presentados por otros autores y en algunas métricas se alcanzó a superar los valores reportados.

| Modelo | Precisión | Recall | mAP50 | mAP50-95 |
|---|---|---|---|---|
| Nuestro<br>YOLOv5<br>YOLO v8<br>YOLOv11 | <br>76.05 ± 2.23<br>76.28 ± 3.36<br>74.87 ± 3.09 | <br>**67.50 ± 1.80**<br>**66.95 ± 2.76**<br>**66.73 ± 2.93** | <br>**72.39± 2.78**<br>**72.92 ± 2.51**<br>**73.27 ± 1.23** | <br>40.42 ± 2.74<br>39.70 ± 2.91<br>39.87 ± 2.23 |
| Aditya Kumar<br>YOLOv8<br>Segmentación | | | Sin aumento<br>55.9<br>Con aumento<br>62.5 | Sin aumento<br>32.1<br>Con aumento<br>49.3 |
| Maros Jakubec<br>Sparse R-CNN<br>YOLO7<br>Baches | <br>R-CNN<br>55.4<br>YOLOv7<br>73.2 | <br>R-CNN<br>46.6<br>YOLOv7<br>56.3 | <br>R-CNN<br>51.4<br>YOLOv7<br>55.6 | |
| Juntao Li<br>(T3D-FCN)<br>Segmentación | ONR<br>79<br>OFR<br>87 | ONR<br>62<br>OFR<br>63 | | |

Tabla 9. Comparativa de los resultados con el estado del arte

## 4.6 4.2.6 Pruebas realizadas en video

Se realizaron pruebas en video para ver el desempeño del modelo en video, para visualizar los videos, puede acceder a las siguientes ligas. En el primer video se la detección de cúmulos de agua se realiza mientras el automóvil se encuentra a baja velocidad, hipervínculo del video: https://youtu.be/xHIOYmtsA_w. Mientras que en la segunda prueba que se realizó el auto se encontraba a alta velocidad en carretera, hipervínculo del video: https://youtu.be/Ch83Bdx5qps.

# CAPITULO 5. CONCLUSIONES

## 5.1 Conclusión

La tarea de detectar cúmulos de agua en imágenes es compleja debido a la naturaleza del problema, la variabilidad de la luz y las diversas condiciones climáticas hacen que la detección en este tipo de tareas sea complicada. En este trabajo, el modelo implementado fue YOLO, en sus versiones 5, 8 y 11, para esta tarea, debido a su buen desempeño que esta cuenta en diversas condiciones del ambiente y el balance que tienen entre precisión y velocidad. Las tres versiones tuvieron un buen desempeño, considerando las diferentes condiciones de luz, de clima y de ambiente.

Se entrenó el modelo con una base de datos propia, para poder realizar el reconocimiento de las acumulaciones de agua bajo diversas condiciones de luz, diversos tipos de caminos, diferentes condiciones climáticas y con a topología característica de México.

Se realizó la creación de una base de datos la cual contiene muestras con cúmulos de agua, considerando diversas condiciones de luz, climatológicas y distintos tipos de camino con una topología mexicana.

Para el entrenamiento y pruebas realizadas, se usaron las tres versiones de YOLO, bajo las mismas condiciones. En general, todas tuvieron buen desempeño. Esta precisión es comparable con las reportadas por otros autores en el estado del arte. La diferencia principal es que las bases de datos usadas por ellos no son accesibles, por lo que no se pudo hacer pruebas con esas bases de datos, para tener una comparación confiable.

Finalmente, es posible la detección de cúmulos de agua en caminos con el uso de algoritmos de inteligencia artificial con un gran nivel de precisión. Si bien, aún el modelo presenta predicciones faltantes, es posible mejorar el desempeño del modelo, si se expande el tamaño y diversidad del conjunto de datos, añadiendo más muestras. Estas muestras deben ser obtenidas en diversos lapsos del día y bajo diversas condiciones climáticas.

Este trabajo abre la posibilidad de que en un futuro se realice la implementación de YOLO en vehículos autónomos terrestres con la ayuda de sistemas embebidos como NVIDIA Jetson o Raspberry Pi para poder evaluar la eficacia del modelo

entrenado en tiempo real. Esto es gracias a que YOLO cuenta con versiones ligeras (lite) que permiten su implementación en este tipo de sistemas embebidos.

## 5.2 Trabajo a Futuro

En este trabajo se presenta el desarrollo de un conjunto de datos para poder lograr la detección de cúmulos de agua, este conjunto de imágenes y etiquetas permite ser expandido añadiendo más muestras, estás pueden contener diferentes condiciones climáticas, de luz y de caminos, lo cual permitiría al modelo obtener mejores métricas y desempeño bajo escenarios reales. El conjunto de datos permitiría realizar un trabajo de segmentación con las imágenes que se cuentan actualmente o aumentar el número de clases que se desea detectar, por ejemplo, seco, mojado, charco o inundado. También se abre la oportunidad de implementar o probar dicho algoritmo en robots terrestres autónomos para poder medir el desempeño del modelo en condiciones reales.

# REFERENCIAS


[1] S. Bhutad, K. Patil, and N. Khare, “Differentiating Stagnant Water from Wet Surface for Detecting Potential Mosquito Breeding Sites in Real Time,” *Indian J Sci Technol*, vol. 16, no. 5, pp. 331–338, Feb. 2023, doi: 10.17485/IJST/v16i5.2111.

[2] S. S. Mekala, G. Koutitas, S. Aslan, and W. Stapleton, “RESEARCH OF WATER DETECTION IN AUTONOMOUS VEHICLES,” Texas State University, Texas, 2019.

[3] L. Juntao, N. Chuong, and Y. Shaodi, “Temporal 3D fully connected network for water-hazard detection,” *Digital Image Computing: Techniques and Applications (DICTA)*, pp. 1–5, 2019, doi: https://ieeexplore.ieee.org/abstract/document/8945849.

[4] M. Jakubec, E. Lieskovská, B. Bučko, and K. Zábovská, “Comparison of CNN-Based Models for Pothole Detection in Real-World Adverse Conditions: Overview and Evaluation,” *Applied Sciences (Switzerland)*, vol. 13, no. 9, May 2023, doi: 10.3390/app13095810.

[5] M. S. Yadav Muske, “To Detect Water-Puddle On Driving Terrain From RGB Imagery Using Deep Learning Algorithms,” Blekinge Institute of Technology, Karlskrona, 2020. [Online]. Available: www.bth.se

[6] M. Abdullah Sandhu, M. Ali Qureshi, and A. Amin, “Water Detection in an Open Environment: A Comprehensive Review,” *IJCSNS International Journal of Computer Science and Network Security*, vol. 23, no. 1, Jan. 2023, doi: 10.22937/IJCSNS.2023.23.1.1.

[7] X. Guo *et al.*, “Water hazard detection: A 20-year review,” Feb. 01, 2023, *Elsevier Ltd*. doi: 10.1016/j.jterra.2022.11.002.

[8] L. Matthies, P. Bellutta, and M. Mchenry, “Detecting water hazards for autonomous off-road navigation,” *Proceedings of SPIE - The International Society for Optical Engineering*, pp. 231–242.

[9] A. Rankin, T. Ivanov, and S. Brennan, “Evaluating the Performance of Unmanned Ground Vehicle Water Detection,” *Proceedings of the 10th Performance Metrics for Intelligent Systems Workshop*, Sep. 2010.

[10] H. Shao, Z. Zhang, and K. Li, “Research on Water Hazard Detection Based on Line Structured Light Sensor for Long-Distance All Day,” *Mechatronics and Automation (ICMA)*, Aug. 2015.

[11] M. G. Prasad, A. Chakraborty, R. Chalasani, and S. Chandran, “Quadcopter-based Stagnant Water Identification,” *Conference on Computer Vision, Pattern Recognition, Image Processing and Graphics (NCVPRIPG)*, p. 1, Dec. 2015.

[12] M. S. Farooq *et al.*, “IoT Enabled Intelligent Stick for Visually Impaired People for Obstacle Recognition,” *Sensors*, vol. 22, no. 22, Nov. 2022, doi: 10.3390/s22228914.

[13] M. Lee, J. C. Kim, M. H. Cha, H. Lee, and S. Lee, “Identifying Puddles based on Intensity Measurement using LiDAR,” *Journal of Sensor Science and Technology*, vol. 32, no. 5, pp. 267–274, Sep. 2023, doi: 10.46670/JSST.2023.32.5.267.

[14] J. Long, E. Shelhamer, and T. Darrell, “Fully Convolutional Networks for Semantic Segmentation,” Mar. 2015, [Online]. Available: http://arxiv.org/abs/1411.4038

[15] H. Tahara, I. Ikegami, K. Takakura, T. Kato, and M. Nagata, “Puddle Detection for Avoidance Path Planning of Wheeled Mobile Robot Using Laser Reflection Intensity,” *IECON 2019 - 45th Annual Conference of the IEEE Industrial Electronics Society*, vol. 1, pp. 699–704, Dec. 2019.

[16] J. J. Qiao, X. Wu, J. Y. He, W. Li, and Q. Peng, “SWNet: A Deep Learning Based Approach for Splashed Water Detection on Road,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 4, pp. 3012–3025, Apr. 2022, doi: 10.1109/TITS.2020.3029006.

[17] X. Han, C. Nguyen, S. You, and J. Lu, “Single Image Water Hazard Detection using FCN with Reflection Attention Units,” *Computer Vision – ECCV 2018*, vol. 11210, 2018.

[18] H. N. Samaranayake, “Detecting Water in Visual Image Streams from UAV with Flight Restrictions,” University of Colombo School of Computing, 2020.

[19] N. Long, H. Yan, L. Wang, H. Li, and Q. Yang, “Unifying Obstacle Detection, Recognition, and Fusion Based on the Polarization Color Stereo Camera and LiDAR for the ADAS,” *Sensors*, vol. 22, no. 7, Apr. 2022, doi: 10.3390/s22072453.

[20] A. Kumar and A. Choudhary, “Water-Puddle Segmentation Using Deep Learning in Unstructured Environments,” in *Proceedings of the 17th IEEE International Conference on Service Operations and Logistics, and Informatics, SOLI 2023*, Institute of Electrical and Electronics Engineers Inc., 2023. doi: 10.1109/SOLI60636.2023.10425657.

[21] EXPANSION, “Los siniestros de autos repuntan en México por las lluvias,” *EXPANSION*, México, Jul. 2023.

[22] L. Pascual, *Development of Puddle Detection System*. University of Auckland, 2020.

[23] D. Parekh *et al.*, “A Review on Autonomous Vehicles: Progress, Methods and Challenges,” *Electronics (Switzerland)*, vol. 11, no. 14, Jul. 2022, doi: 10.3390/electronics11142162.

[24] Waymo, “Waymo,” https://waymo.com/intl/es/.

[25] R. C. . Gonzalez and R. E. . Woods, *Digital image processing*, 4th ed. Pearson, 2018.

[26] C. Solomon and T. Breckon, *Fundamentals of Digital Image Processing A Practical Approach with Examples in Matlab*. John Wiley & Sons, Ltd, 2011.

[27] A. Gonzáles Marcos *et al.*, *VisionArtificial tecnicas y algoritmos basicos*. Universidad de La Rioja. Servicio de Publicaciones, 2006.

[28] A. C. . Bovik, *The essential guide to image processing*. San Diego: Academic Press, 2009.

[29] S. Gollapudi and V. Laxmikanth, *Learn Computer Vision Using OpenCV: With Deep Learning CNNs and RNNs*. Apress Media LLC, 2019. doi: 10.1007/978-1-4842-4261-2.

[30] Hala. Nelson, *Essential math for AI : next-level mathematics for efficient and successful AI systems*, 1st ed. O’Reilly, 2023.

[31] A. C. Müller and S. Guido, *Introduction to Machine Learning with Python A GUIDE FOR DATA SCIENTISTS Introduction to Machine Learning with Python*. O’Reilly, 2017.

[32] IBM, “What is a neural network?,” IBM.

[33] A. Krogh, “What are artificial neural networks?,” *Nat Biotechnol*, vol. 26, Feb. 2008, [Online]. Available: http://www.r-project.org/

[34] A. Glassner, *Deep Learning A Visual Approach*. San Francisco: William Pollock, 2021.

[35] IBM, “What is deep learning?,” IBM.

[36] J. Asit Kumar Das, B. N. S. Nayak, and D. P. Kumar Pati, *Computational Intelligence in Pattern Recognition*, 1st ed. Springer Singapore, 2019.

[37] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” Jan. 2016, [Online]. Available: http://arxiv.org/abs/1506.01497

[38] W. Liu *et al.*, “SSD: Single Shot MultiBox Detector,” Dec. 2016, doi: 10.1007/978-3-319-46448-0_2.

[39] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, IEEE Computer Society, Dec. 2016, pp. 779–788. doi: 10.1109/CVPR.2016.91.

[40] J. Terven and D. Cordova-Esparza, “A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS,” Feb. 2024, doi: 10.3390/make5040083.

[41] D. Dakari Aboyomi and C. Daniel, “A Comparative Analysis of Modern Object Detection Algorithms: YOLO vs. SSD vs. Faster R-CNN,” *ITEJ Information Technology Engineering Journals*, vol. 8, pp. 96–106, 2023, [Online]. Available: https://syekhnurjati.ac.id/journal/index.php/itej

[42] G. Jocher, “Ultralytics open-source object detection architecture.,” https://github.com/ultralytics/ultralytics.

[43] C. Sager, C. Janiesch, and P. Zschech, “A survey of image labelling for computer vision applications,” *Journal of Business Analytics*, vol. 4, no. 2, pp. 91–110, 2021, doi: 10.1080/2573234X.2021.1908861.

[44] Z. Su, A. Adam, M. F. Nasrudin, M. Ayob, and G. Punganan, “Skeletal Fracture Detection with Deep Learning: A Comprehensive Review,” Oct. 01, 2023, *Multidisciplinary Digital Publishing Institute (MDPI)*. doi: 10.3390/diagnostics13203245.

[45] MATLAB, “Multilabel Image Classification Using Deep Learning,” https://www.mathworks.com/help/deeplearning/ug/multilabel-image-classification-using-deep-learning.html.

[46] R. K. Sinha, R. Pandey, and R. Pattnaik, “Deep Learning For Computer Vision Tasks: A review,” *International Conference on Intelligent Computing and Control (I2C2)*, 2017.

[47] Mostafa Ibrahim, “Enhancing real estate with Object Detection and Computer Vision,” https://www.ultralytics.com/blog/enhancing-real-estate-with-object-detection-and-computer-vision.

[48] R. Padilla, S. L. Netto, and E. A. B. da Silva, “A Survey on Performance Metrics for Object-Detection Algorithms,” *International Conference on Systems, Signals and Image Processing (IWSSIP*, pp. 237–242, 2020.

[49] H. Zhu, H. Wei, B. Li, X. Yuan, and N. Kehtarnavaz, “A review of video object detection: Datasets, metrics and methods,” Nov. 01, 2020, *MDPI AG*. doi: 10.3390/app10217834.

[50] R. Padilla, W. L. Passos, T. L. B. Dias, S. L. Netto, and E. A. B. Da Silva, “A comparative analysis of object detection metrics with a companion open-source toolkit,” *Electronics (Switzerland)*, vol. 10, no. 3, pp. 1–28, Feb. 2021, doi: 10.3390/electronics10030279.

[51] C. Shorten and T. M. Khoshgoftaar, “A survey on Image Data Augmentation for Deep Learning,” *J Big Data*, vol. 6, no. 1, Dec. 2019, doi: 10.1186/s40537-019-0197-0.

[52] S. Yang, W. Xiao, M. Zhang, S. Guo, J. Zhao, and F. Shen, “Image Data Augmentation for Deep Learning: A Survey,” Nov. 2023, [Online]. Available: http://arxiv.org/abs/2204.08610

[53] Ultralytics, “Data Augmentation using Ultralytics YOLO,” https://docs.ultralytics.com/guides/yolo-data-augmentation/.

[54] R. Andonie, “Hyperparameter optimization in learning systems,” *Journal of Membrane Computing*, vol. 1, no. 4, pp. 279–291, Dec. 2019, doi: 10.1007/s41965-019-00023-0.

[55] Google, “Regresión lineal: Hiperparámetros,” https://developers.google.com/machine-learning/crash-course/linear-regression/hyperparameters?hl=es-419#:~:text=Los%20hiperpar%C3%A1metros%20son%20variables%20que,Tama%C3%B1o%20del%20lote.

[56] C. V. Nguyen, M. Milford, and R. Mahony, “3D tracking of water hazards with polarized stereo cameras,” Feb. 2017, [Online]. Available: http://arxiv.org/abs/1701.04175

[57] CVAT, “CVAT,” https://www.cvat.ai/.

[58] N. Bilous, V. Malko, M. Frohme, and A. Nechyporenko, “Comparison of CNN-Based Architectures for Detection of Different Object Classes,” *AI*, vol. 5, no. 4, pp. 2300–2320, Dec. 2024, doi: 10.3390/ai5040113.

[59] Google, “Colab,” https://colab.research.google.com/.

[60] Google, “Datasets: Dividing the original dataset,” https://developers.google.com/machine-learning/crash-course/overfitting/dividing-datasets.

[61] B. Vrigazova, “The Proportion for Splitting Data into Training and Test Set for the Bootstrap in Classification Problems,” *Business Systems Research*, vol. 12, no. 1, pp. 228–242, May 2021, doi: 10.2478/bsrj-2021-0015.

# ANEXOS

## Registro de protocolo

UNIVERSIDAD AUTÓNOMA DE QUERÉTARO
FACULTAD DE INGENIERÍA
DIVISIÓN DE INVESTIGACIÓN Y POSGRADO

REGISTRO DEL PROTOCOLO DE INVESTIGACIÓN
DEL ESTUDIANTE DE POSGRADO

| | | |
|---|---|---|
| Los 2 Espacios obscuros exclusivos para la Dirección | No. Registro de Proyecto*: | 15227 |
| | Fecha de Registro*: | 25-FEBRERO-2025 |
| | Fecha de inicio de proyecto: | 15/01/2024 |
| | Fecha de término de proyecto: | 15/12/2025 |

| 1. DATOS DEL SOLICITANTE | | |
|---|---|---|
| No. de expediente: | 328939 | |
| Apellido Paterno | Apellido Materno | Nombre(s) |
| TRUJILLO | LOPEZ | MIGUEL |
| Dirección: | | |
| Calle y número | Colonia | C.P. |
| Zodiaco #108 | Bolaños | 76149 |
| Estado | Teléfono (incluir lada) | Correo Electrónico |
| 22 | 9513260629 | miguetrulo07@gmail.com |

| 2. DATOS DEL PROYECTO | | |
|---|---|---|
| Facultad: | INGENIERÍA | |
| Programa: | MAESTRIA EN CIENCIAS EN INTELIGENCIA ARTIFICIAL | |
| Tema específico del proyecto: | Identificación de concentraciones de agua en caminos usando técnicas de inteligencia artificial | |
| RAMOS ARREGUIN JUAN MANUEL<br>Director de tesis | TOVAR ARRIAGA SAUL<br>Coordinador de programa | TRUJILLO LOPEZ MIGUEL<br>Alumno |
| Dr. Juan Carlos Jáuregui Correa<br>Jefe de División de Investigación y Posgrado de la Fac. de Ing. | Dra. María de la Luz Pérez Rea<br>Directora de Fac. Ing. | Dr. Manuel Toledano Ayala<br>Director de Investigación y Posgrado UAQ |

Anexo 3 Manual de procedimientos Administrativos de Posgrado de la Facultad de Ingeniería

## Dictamen ético

FACULTAD DE INGENIERÍA
COMITÉ DE ÉTICA

C. U., 09 de diciembre de 2024

**Miguel Trujillo Lopez**
**Estudiante de Maestría en Ciencias en Inteligencia Artificial**
**Expediente 328939**

**Presente**

El Comité de Ética Aplicada a la Investigación de la Facultad de Ingeniería ha revisado el protocolo del trabajo de tesis:

CEAIFI-195-2024-TP

**Identificación de concentraciones de agua en caminos usando técnicas de inteligencia artificial**

Con apego a los lineamientos éticos de beneficencia, no maleficencia, justicia y autonomía, este comité ha dado el siguiente dictamen:

**Exento de dictamen ético**

El presente dictamen tiene vigencia de un año a partir de su fecha de emisión.

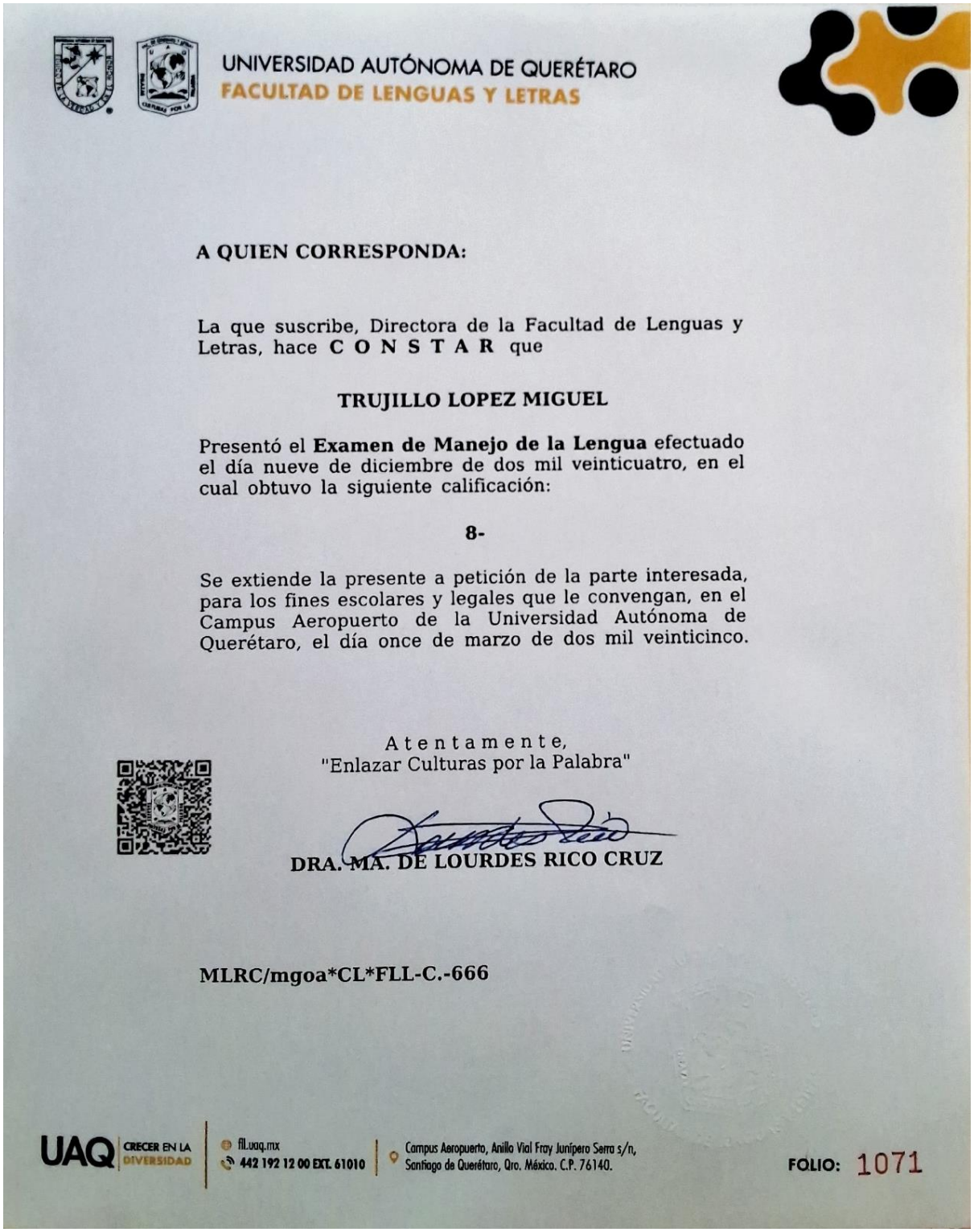
Sirva esta carta para los fines académicos que al interesado convengan.

Atentamente
**"El ingenio para crear, no para destruir"**

Dra. Aurora Femat Díaz
Presidente del CEAIFI
afemat@uaq.mx

UNIVERSIDAD AUTÓNOMA DE QUERÉTARO
CERRO DE LAS CAMPANAS S/N, COL. LAS CAMPANAS
FACULTAD DE INGENIERÍA
C.P. 76010 QUERÉTARO, QRO. TEL. 192 12 00 EXT. 6023

## Constancia de Ingles



UNIVERSIDAD AUTÓNOMA DE QUERÉTARO
FACULTAD DE LENGUAS Y LETRAS

**A QUIEN CORRESPONDA:**

La que suscribe, Directora de la Facultad de Lenguas y Letras, hace **C O N S T A R** que

**TRUJILLO LOPEZ MIGUEL**

Presentó el **Examen de Manejo de la Lengua** efectuado el día nueve de diciembre de dos mil veinticuatro, en el cual obtuvo la siguiente calificación:

**8-**

Se extiende la presente a petición de la parte interesada, para los fines escolares y legales que le convengan, en el Campus Aeropuerto de la Universidad Autónoma de Querétaro, el día once de marzo de dos mil veinticinco.

A t e n t a m e n t e,
"Enlazar Culturas por la Palabra"

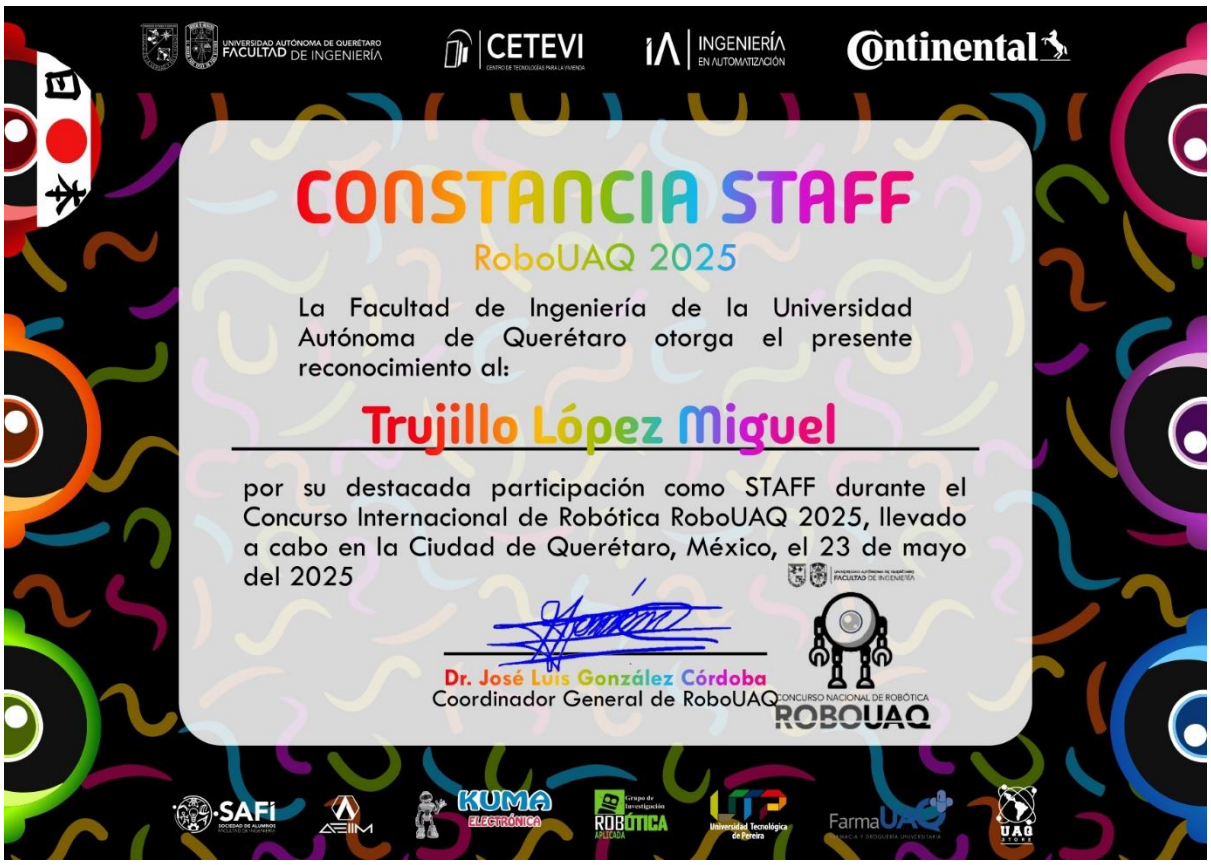
**DRA. MA. DE LOURDES RICO CRUZ**

**MLRC/mgoa*CL*FLL-C.-666**

UAQ CRECER EN LA DIVERSIDAD | fll.uaq.mx | 442 192 12 00 EXT. 61010 | Campus Aeropuerto, Anillo Vial Fray Junípero Serra s/n, Santiago de Querétaro, Qro. México. C.P. 76140.

FOLIO: 1071

## Retribución Social



FACULTAD DE INGENIERÍA
DIRECCIÓN DE INVESTIGACIÓN Y POSGRADO

# Constancia de actividades de retribución social

Santiago de Querétaro a 13 de mayo de 2026

A quien corresponda:
P r e s e n t e.

En cumplimiento a lo establecido en el *Artículo 20, Capítulo VII. De la Conclusión de la Beca o Apoyo*, del *Reglamento de Becas de la Secretaría de Ciencia, Humanidades Tecnología e Innovación*, y en el marco de la Convocatoria **Becas nacionales para estudios de posgrado 2024-1**, hago constar que el **C. Miguel Trujillo Lopez** con número de **CVU 1347351** , **beneficiado** con una beca para obtener el grado de **Maestría** en el programa **005351 – Maestría en Ciencias en Inteligencia Artificial**, que se imparte en la **Universidad Autónoma de Querétaro - Campus Aeropuerto**, realizó las actividades de retribución social durante el periodo de vigencia de la beca en el tiempo en el que fue **alumno** regular de esta Institución.

Asimismo, hago constar que, conforme a lo establecido en la Ley General de Archivos, la coordinación del posgrado organiza y conserva la evidencia documental de dichas actividades en caso de que la SECIHTI o cualquier otra instancia la requiera.

Sin más por el momento, le envío un cordial saludo.

Dr. Saúl Tovar Arriaga

Coordinador del programa Maestría en Ciencias en Inteligencia Artificial

**UNIVERSIDAD AUTÓNOMA DE QUERÉTARO**
CERRO DE LAS CAMPANAS S/N, COL. LAS CAMPANAS
FACULTAD DE INGENIERÍA, EDIFICIO G.
C.P. 76010 QUERÉTARO, QRO. TEL. 192 12 00 EXT. 6023

**Publicación de articulo**


Estimados Autores del artículo con título

***"Detección De Concentraciones de Agua en Caminos Usando YOLO"***

enviado para su participación en el XIX Coloquio de Posgrado de la Facultad de Ingeniería ha sido:

ACEPTADO:

Para su participación en el Coloquio a desarrollarse del 19 al 21 de Noviembre del 2025 en la Ciudad de Querétaro, Querétaro

*Noviembre de 2025*

*Facultad de Ingeniería*

# Detección De Concentraciones de Agua en Caminos Usando YOLO

Water-puddle Detection on Roads Using YOLO

Trujillo Lopez, Miguel; Ramos Arreguin, Juan Manuel; Pedraza Ortega, Jesús Carlos; Tovar Arriaga, Saul; Aceves Fernández, Marco Antonio; Ramírez Arriaga, Karen Andrea.

*Facultad de Ingeniería*
*Universidad Autónoma De Querétaro*
*Querétaro, México*
*miguetrulo@gmail.com; ORCID ([https://orcid.org/0009-0003-3807-4136])*
*juan.ramos@uaq.edu.mx; ORCID ([https://orcid.org/0000-0002-2604-9692])*
*caryoko@yahoo.com; ORCID ([https://orcid.org/0000-0001-5125-8907])*
*saul.tovar@uaq.mx; ORCID ([https://orcid.org/0000-0002-2695-1934])*
*marco.aceves@uaq.edu.mx; ORCID ([https://orcid.org/0000-0002-5455-0329])*
*andrea.r.arriaga@gmail.com ORCID ([https://orcid.org/ 0000-0003-4910-6298])*

**Resumen:** La presencia de cúmulos de agua en caminos o carreteras puede ser una situación peligrosa para los automóviles, estos pueden ocultar de la vista del conductor ciertos obstáculos los cuales pueden causar un accidente. Tan solo en México el número de accidentes de tráfico aumenta considerablemente en temporada de lluvias. El desarrollo de tecnologías que permitan la detección de este tipo de fenómeno puede ayudar a la asistencia en la conducción. En este trabajo se presenta la implementación de un modelo de inteligencia artificial basado en el método YOLO el cual es capaz de detectar las concentraciones de agua con una alta precisión (78%) bajo diversas condiciones climatológicas, de luz, en distintos tipos de condiciones de camino, estructurado (carreteras o caminos pavimentado) y no estructurado (caminos sin pavimento) y en una topología de México (Querétaro-Guanajuato).

**Palabras clave:** Detección de cúmulos de agua; Visión por computadora; YOLO; Aprendizaje profundo; Detección de objetos.

**Abstract:** The presence of water-puddle on roads or highways can be a dangerous situation for cars. These can obscure certain obstacles from the driver's view, potentially causing an accident. In Mexico alone, the number of traffic accidents increases considerably during the rainy season. The development of technologies that allow the detection of this type of phenomenon can aid in driver assistance. This paper presents the implementation of an artificial intelligence model based on the YOLO method, which is capable of detecting water concentrations with high precision (78%) under various weather and light conditions, in different types of road conditions, structured (roads or paved roads) and unstructured (unpaved roads) and in a topology of Mexico (Querétaro-Guanajuato).

**Keywords:** Water-puddle detection; Computer Vision; YOLO; Deep Learning; Object detection.

## 1. Introducción

La conducción se ha convertido en una de las actividades principales y más frecuentes dentro de nuestra sociedad. Millones de personas viajan o conducen un vehículo automotor, por lo que el desarrollo de tecnologías que mejoren o den asistencia al conductor son de beneficio para sus usuarios. A través de los años los vehículos han sido construidos con un mayor número de herramientas que ayuden a la conducción y seguridad del mismo.

La lluvia es un evento climatológico el cual afecta muchas veces a varios sectores de la ciudad, de entre los cuales se encuentra la conducción y el tráfico. Las afectaciones causadas por la lluvia son diversas, estas afectan la infraestructura de las calles cuando se llenan de agua, llegando así a provocar diversos percances a los conductores. Debido a que el pavimento mojado puede causar un mal funcionamiento del vehículo o provocar concentraciones de agua (charcos) las cuales ocultan a la vista del piloto posibles obstáculos como hoyos u objetos (ramas, topes,) que afecten al vehículo y sus tripulantes.

Solamente en México entre el año 2021 y 2022 los accidentes vehiculares en temporada de lluvias llegaron a aumentar hasta un 13%, mientras que en los últimos 5 años alrededor del 25% de los accidentes registrados se dan en condiciones de lluvia [1]. En Estados Unidos se reporta que alrededor de 3 billones de dólares se gasta de forma anual debido a daños causados por baches [2]. Manejar en medio de cúmulos de agua o zonas inundadas puede causar

UNIVERSIDAD AUTÓNOMA DE QUERÉTARO | FACULTAD DE INGENIERÍA | DIPFI POSGRADO INGENIERÍA

COLOQUIO 19° POSGRADO Fi

Se otorga la presente

# CONSTANCIA a:

Trujillo Lopez, Miguel; Ramos Arreguin, Juan Manuel; Pedraza Ortega, Jesús Carlos; Tovar Arriaga, Saul; Aceves Fernández, Marco Antonio; Ramírez Arriaga, Karen Andrea.

Por su participación en el
XIX Coloquio de Posgrado de la Facultad de Ingeniería
de la Universidad Autónoma de Querétaro con el artículo:

*"Detección De Concentraciones de Agua en Caminos Usando YOLO"*

*Noviembre de 2025*

*Facultad de Ingeniería*

Dra. María de la Luz Pérez Rea
Directora
Facultad de Ingeniería

Dr. Juan Carlos Jauregui Correa
Jefe de la División de Investigación y Posgrado
Facultad de Ingeniería

Msc. Luis Angel Iturralde Carrera
Presidente del Coloquio de Posgrados de la
Facultad de Ingeniería