ISSN: 1692-7257 - Volumen 1 – Número 43 - 2024

RCTA
Revista Colombiana de Tecnologías de Avanzada
UNIPAMPLONA

Digital Object Identifier: 10.24054/rcta.vXiXX.1234

Comparación de arquitecturas YOLO para la detección de ciclistas urbanos en un entorno de vehículos autónomos

YOLO architectures comparison for urban cyclist detection in an autonomous driving environment

Jorge David[1], Mateo Quintero[1], Paula Ortiz[1], Luís Gómez[1]
MSc. Mauricio Arias-Correa[1]

[1] ***Instituto Tecnológico Metropolitano,*** *Laboratorio de Visión Artificial y Fotónica, Facultad de Ingenierías, Medellín, Colombia.*

Correspondencia: *mauricioarias@itm.edu.co*

Recibido: *15 octubre 2023.* ***Aceptado:*** *17 diciembre 2023.* ***Publicado:*** *13 marzo 2024.*

Cómo citar: *JK Autor, "Nombre del artículo", RCTA, vol. x, n.°. x, pp. xxx-xxx, Abrev. Mes, año. Recuperado de https://*

Resumen: La OMS establece que más del 55% de las muertes en accidentes viales son de usuarios vulnerables, incluyendo un 3% de ciclistas. Aunque los vehículos autónomos pueden detectar objetos y personas en las carreteras, la detección de ciclistas y la predicción de sus movimientos siguen siendo desafíos significativos. Este artículo presenta resultados al comparar las arquitecturas YOLOv7, YOLOv8 y YOLO-NAS para detectar ciclistas urbanos. La metodología garantiza que los detectores se entrenaron bajo las mismas condiciones. Luego, se evaluaron con 111 imágenes de ciclistas utilizando métricas como IoU, precisión y recall. Los resultados destacan ventajas y desventajas en cada arquitectura, lo que sugiere priorizar el tiempo de inferencia o la calidad de la detección de ciclistas en futuros trabajos.

Palabras clave: Yolo, VRU, deep learning, detección de ciclistas, vehículo autónomo.

Abstract: The World Health Organization (WHO) states that over 55% of road traffic accident fatalities involve vulnerable road users, including 3% who are cyclists. While autonomous vehicles are capable of detecting objects and individuals on roadways, the detection of cyclists and the prediction of their movements continue to pose significant challenges. This paper presents results from the comparison of YOLOv7, YOLOv8, and YOLO-NAS architectures for urban cyclist detection. The methodology ensures that the detectors were trained under the same conditions. Subsequently, they were evaluated using 111 cyclist images with metrics such as IoU, precision, and recall. The results highlight advantages and disadvantages within each architecture, suggesting a priority for either inference time or the quality of cyclist detection in future work.

Keywords: Yolo, VRU, deep learning, cyclists' detection, autonomous vehicle.

1. INTRODUCCIÓN

De acuerdo con reporte de la Organización Mundial de la Salud (OMS) en 2018, las lesiones de tráfico son la octava causa principal de muerte para personas de todas las edades (1.35 millones en 2016), y más de la mitad de ellas son el grupo compuesto por peatones, ciclistas y motociclistas, denominados usuarios vulnerables de la vía. Un Usuario Vulnerable de la Vía (VRU por sus siglas en inglés), es un usuario de la vía con un riesgo aumentado de resultar herido o muerto en el tráfico debido a que no está rodeado por una cubierta protectora que reduciría significativamente la gravedad de un accidente [1]. Esta definición abarca a todos los tipos de peatones, ciclistas y motociclistas, personas con discapacidades o movilidad reducida. Identificar correctamente a los VRU es una de las tareas de percepción del entorno más desafiantes para los vehículos autónomos (AVs).

Debido al auge actual y futuro de los vehículos de conducción autónoma [2], es importante desarrollar sistemas de detección de VRU eficaces para los AVs. Varios trabajos han propuesto soluciones, principalmente para la detección de peatones [3], [4], [5], [6], [7], [8]. En la detección de ciclistas, sin embargo, no se ha hecho el mismo énfasis debido, probablemente, a que se ha presentado como una de las tareas de percepción más desafiantes a las que se enfrenta un AV [9], [10]. Dicha complejidad está relacionada con aspectos tales como la complejidad visual de los ciclistas, la variedad de posibles ángulos de orientación, inclinación y elevación, las diferentes relaciones de aspecto, la apariencia diversa, las oclusiones, reflejos, sombras y fondos que confunden a los detectores [11].

Para que la integración en el tráfico de los AVs se incremente durante los próximos años, será necesario mejorar sus capacidades de detección de VRU, especialmente de ciclistas, pues en muchos países los ciclistas y los vehículos comparten la vía y no sería aceptable que las lesiones y muertes de ciclistas se incrementaran por cuenta de colisiones con AVs.

Todo lo anterior justifica el desarrollo de detectores de ciclistas urbanos, como usuarios vulnerables de la vía, para ser implementados en los vehículos autónomos [12].

Algunos autores han propuesto sistemas de detección de ciclistas basados en técnicas de machine learning [13], pero el avance de las arquitecturas de Deep Learning ha demostrado la efectividad de los detectores basados en arquitecturas YOLO, sobrepasando el desempeño de otras arquitecturas [14].

En este artículo se propone realizar una comparación de detectores de ciclistas entrenados con los modelos de las tres arquitecturas YOLO más recientes: YOLOv7, YOLOv8 y YOLO-NAS. La metodología propuesta incluye el levantamiento y curaduría de un conjunto de datos, el entrenamiento de las arquitecturas y el uso de métricas adecuadas para evaluar su desempeño. Los resultados y su análisis son presentados en detalle como insumo para futuros proyectos que pretenden optimizarlos y aplicarlos en el área de la conducción autónoma y en una de sus ramas más importantes: factores humanos en conducción autónoma.

2. METODOLOGÍA

En la figura 1 se puede apreciar el diagrama de flujo que representa la metodología aplicada para realizar la comparación de las arquitecturas YOLO. El proceso inició con la etapa de creación del conjunto de datos (dataset), los cuáles sirvieron para entrenar, en la siguiente etapa a cada una de las tres arquitecturas: YOLOv7, YOLOv8 y YOLO-NAS. Los resultados de las predicciones se utilizaron para calcular métricas de desempeño: IoU, Recall, Precision, y el tiempo de inferencia, correspondientes a cada arquitectura. Con esos resultados como salida del proceso, se genera una comparación objetiva.

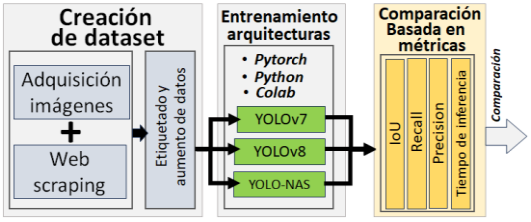

***Fig. 1**. Diagrama de flujo de la metodología desarrollada durante el proceso.*
***Fuente:** elaboración propia.*

Debido a que el entrenamiento de los detectores YOLO debe estar orientado a la detección de ciclistas urbanos, que eventualmente serían captados en la vía por un vehículo autónomo en movimiento (lo cual aportaría al mejoramiento de las estrategias de evitación de colisión entre dichos vehículos y los ciclistas como usuarios vulnerables de la vía); se decidió crear un dataset constituido por una combinación de imágenes adquiridas desde el parabrisas frontal de un vehículo en movimiento y

las imágenes públicas (sin restricciones de descarga y uso) de ciclistas urbanos extraídas desde la web a través de un software de web scraping denominado Bitzi, previamente desarrollado en el Instituto Tecnológico Metropolitano de Medellín, y registrado en la Dirección Nacional de Derechos de Autor en el año 2015 [15].

A partir del conjunto de datos inicialmente recopilado, se seleccionaron dos grandes grupos. El primer grupo constituido por imágenes de un solo ciclista, el segundo grupo constituido por imágenes de varios ciclistas. Posteriormente, se realizó un aumento de datos (data augmentation) sobre las imágenes RGB seleccionadas para diversificar el conjunto de datos y mejorar el rendimiento de los modelos que serán entrenados. Aunque es común utilizar el data augmentation para realizar transformaciones geométricas sobre las imágenes tales como rotación, escala y traslación, también se pueden realizar cambios en el brillo, contraste, saturación y tono, en enfoque, adición de ruido, entre otros [16], [17]. En nuestro caso el aumento de datos consistió en realizar cambios en el brillo y contraste de las imágenes para simular condiciones de iluminación diferente. Hacer cambios de saturación y tono para afectar la apariencia de los colores. Adicionar ruido aleatorio para simular condiciones climáticas adversas y filtros de desenfoque para modificar la apariencia de los ciclistas en las imágenes. Las transformaciones realizadas sobre el conjunto de datos se pueden apreciar en la figura 2.

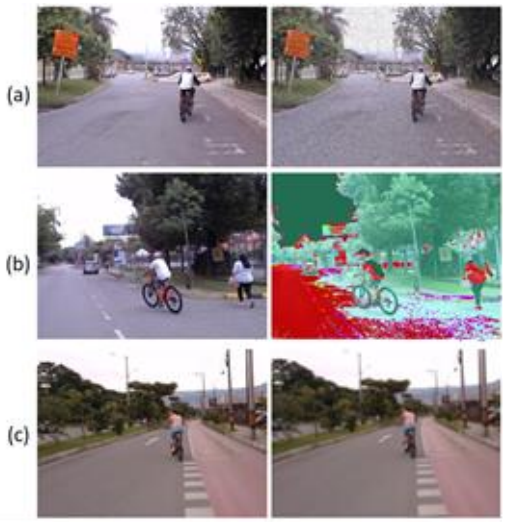

***Fig. 2**. Aumento de datos aplicado a las imágenes RGB del conjunto de datos etiquetado. En la columna de la izquierda se presentan las imágenes RGB originales, en la columna derecha las correspondientes transformaciones. En la fila (a), se aplicó un ruido sal y pimienta a la imagen, en (b) un filtro de espacio de color HSV, en (c), se aplicó un desenfoque.*
***Fuente:** elaboración propia.*

Las arquitecturas YOLO (“You Only Look Once”) han sido seleccionadas para entrenar los detectores, debido a su capacidad para detectar objetos en una sola pasada a través de una imagen en tiempo real, haciéndolos especialmente eficientes en términos de velocidad de procesamiento. El enfoque YOLO se basa en una única red neuronal convolucional (CNN) que toma una imagen como entrada y produce salidas en forma de cajas delimitadoras (bounding boxes) que identifican y localizan los objetos en la imagen. Esto permite a YOLO detectar simultáneamente a múltiples objetos en una sola imagen [18].

Las versiones más recientes: YOLOv7, YOLOv8 y YOLO-NAS, representan la vanguardia de la detección de objetos y su aplicación en la detección de ciclistas permitirá establecer cuáles son las ventajas de las arquitecturas con respecto a otras, que han sido reportadas en trabajos recientes para la detección de orientación de ciclistas tales como [13].

YOLOv7 [19], sigue la misma arquitectura del YOLO original con sus tres componentes fundamentales: la columna, el cuello y la cabeza. Cada uno de estos componentes juega un rol crucial en el proceso de reconocimiento de imágenes y detección de objetos. El propósito detrás del desarrollo del YOLOv7 fue diseñar una arquitectura capaz de predecir rectángulos delimitadores con una precisión superior a la de otros modelos, manteniendo la misma rapidez en la inferencia [20], [14]. Para lograr dicho propósito, YOLOv7 introdujo una mejora en la eficiencia de la columna vertebral de YOLO que toma en cuenta los requerimientos de memoria y la retro-propagación del descenso del gradiente para mejorar las capacidades de aprendizaje de la red. También incluye una planificación de re-parametrización lo cual permite la variación interna de los niveles de precisión y velocidades de inferencia en función de la aplicación.

En 2023 aparece YOLOv8 [21], con las mismas características de las arquitecturas YOLO precedentes, pero incorporando un modelo libre de ancla que le permite predecir directamente el centro de un objeto, mitigando los desafíos asociados con las regiones alrededor de las imágenes tales como la falta de generalización y dificultad en el manejo de irregularidades, lo cual a su reduce el número de rectángulos de predicción. Además, mejora la velocidad del proceso de generación de candidatos de detección, después del proceso de inferencia. YOLOv8, incorpora también una estrategia de

aumento de imágenes (Spatial Pyramid Pooling Feature), lo cual le permite al modelo aprender objetos en nuevas posiciones, oclusiones parciales y en contra de variaciones en los píxeles que rodean al objeto [22].

YOLO-NAS, es un modelo de detección de objetos innovador, que aprovecha al máximo los avances más recientes en la tecnología Deep Learning. Supera las limitaciones de las versiones previas de YOLO. El término "NAS", hace referencia a "Neural Architecture Search" (búsqueda de arquitectura neural), lo cual implica la optimización de algoritmos para automatizar el proceso de diseño de diseño de arquitecturas de redes neuronales [23] El principal objetivo de YOLO-NAS es lograr un balance óptimo entre la precisión del modelo, la complejidad computacional y el tamaño del modelo. Esta arquitectura representa la vanguardia de la detección de objetos en tiempo real [14].

El entrenamiento de cada una de las tres arquitecturas YOLO se llevó a cabo en una máquina de cómputo de Colab Pro asignada por Google Colab. El dataset se distribuyó en tres proporciones distintas para el entrenamiento: el 72.5% de las imágenes (1689 imágenes) se destinaron al conjunto de entrenamiento, el 20% (465 imágenes) al conjunto de validación y otro 7.5% (176 imágenes) al conjunto de prueba.

Las características de la máquina se presentan en la tabla 1.

Tabla 1: *Características de la máquina de cómputo asignada - Colab Pro-*

Dispositivo	Característica
CPU	Intel Xeon E5-2686 v4 (2.4GHz, 12 núcleos)
GPU	NVIDIA Tesla T4 CUDA cores: 72 Memoria: 16 GB GDDR6 Ancho de banda de memoria: 320 GB/s
Memoria	32 GB
Almacenamiento	500GB

Las características del entrenamiento también fueron las mismas y expresadas como hiper-parámetros se pueden apreciar en la tabla 2.

Tabla 2: *Hiperparámetros asignados para entrenamiento de las arquitecturas YOLO*

Hiper-parámetro	Valor
warmup_initial_lr	1e-6
lr_warmup_epochs	2.0
initial_lr	0.001
max_epochs	30
batch size	16

Todas las implementaciones de código fueron desarrolladas utilizando el lenguaje de programación Python sobre Colab, con el framework de PyTorch y cargando modelos pre-entrenados (transfer learning) para las tres arquitecturas de YOLO. En la tabla 3, se presentan las fuentes de los modelos.

Tabla 3: *Origen de los modelos pre-entrenados de YOLO*

Modelo	Repositorio
YOLOv7	https://github.com/WongKinYiu/yolov7
YOLOv8	https://github.com/ultralytics/ultralytics
YOLO-NAS	https://github.com/Deci-AI/supergradients/blob/master/YOLONAS.md

Comparar arquitecturas de redes, en particular de clasificadores y detectores de objetos en imágenes, requiere de una evaluación por medio de métricas de desempeño. Las métricas se calculas para un grupo de imágenes, a partir de las predicciones que arrojan los detectores (de ciclistas en este caso). Esas predicciones pueden estar equivocadas (Falsos positivos FP, Falsos negativos FN) o ser correctas (Verdaderos positivos TP, verdaderos negativos TN) y las razones de sus valores sirven para calcular las métricas denominadas: Error, Accuracy, Precision, Recall, F1-score y mAP, según sea necesario, se utilizan algunas de ellas o todas. En este trabajo, se utilizarán las métricas IoU, precision y recall, así como la relación de estas dos últimas, para evaluar el desempeño de las arquitecturas entrenadas.

Precision es una métrica utilizada en tareas de detección de objetos para evaluar la precisión de las predicciones positivas del modelo. Ayuda a determinar la confiabilidad del modelo al identificar instancias positivas, minimizando los falsos positivos. Una precisión más alta, indica una menor tasa de instancias positivas falsamente predichas [24].

$$Precision = \frac{TP}{TP + FP} \quad (1)$$

Precision se calcula por medio de la ecuación 1, donde: TP representa el número de instancias positivas predichas correctamente y FP representa el número de instancias falsamente predichas como positivas.

Recall mide la proporción de instancias positivas reales correctamente identificadas por el modelo. El recall cuantifica la habilidad del modelo para detectar correctamente y capturar instancias de objetos. Un mayor recall indica una menor rata de detecciones perdidas [24].

$$Recall = TPR = \frac{TP}{P} = \frac{TP}{FN + TP} \quad (2)$$

Se calcula por medio de la ecuación 2, donde: TP representa el número de instancias predichas correctamente como positivas y FN representa el número de instancias falsamente predichas como negativas.

En la siguiente sección se presentarán los resultados obtenidos incluyendo los detalles de la aplicación de otras métricas y la obtención de las curvas, relevantes para el proceso de comparación.

3. RESULTADOS

En la tabla 4, se puede apreciar los tiempos de entrenamiento para cada arquitectura YOLO evaluada. Es importante tener en cuenta que las características de los entrenamientos y las máquinas de procesamiento utilizadas fueron las mismas, según se especificó en la sección de metodología.

Tabla 4: *Tiempo de entrenamiento*

Modelo	Tiempo de entrenamiento (h)
YOLOv7	3.5
YOLOv8	3.83
YOLO-NAS	2.33

Bajo las mismas condiciones de entrenamiento, la arquitectura YOLOv8 requirió de mayor tiempo (3.83 horas), mientras que la arquitectura YOLO-NAS fue entrenada en apenas el 60% del mismo tiempo.

Fig. 3. *Imágenes evaluadas. En (a) y (b) imágenes de un ciclista. En (c) y (d) imágenes de varios ciclistas.*
Fuente: *elaboración propia.*

Para un total de 111 imágenes de ciclistas (52 imágenes de un ciclista y 58 de varios ciclistas) que incluyeron desde imágenes con una instancia perfectamente definida (un ciclista perfectamente definido y sin oclusiones), hasta imágenes con múltiples instancias con oclusiones según se aprecia en la figura 3, se obtuvieron los valores de las predicciones que afectan directamente el cálculo de las métricas precision y recall. Dichos valores son: verdaderos positivos (TP), falsos positivos (FP) y falsos negativos (FN), y los resultados se pueden apreciar en la tabla 5. En la misma tabla se incluyó el índice de superposición IoU (intersection over unión) con un valor umbral del 50% de superposición. Esto último indica que una detección se considera efectiva si el índice de Jaccard (IoU) entre la detección y el ground truth es superior a 0.5 [25], lo cual se ilustra en la Figura 4.

Tabla 5*: Valores de las predicciones relevantes para cada modelo YOLO*

Arquitectura	TP	FP	FN	IoU
YOLOv7	182	100	91	0.4043
YOLOv8	240	42	49	0.6083
YOLO-NAS	244	38	26	0.7002

Fig. 4. *Obtención del índice de superposición (IoU) para una imagen del grupo de imágenes de evaluación. El cuadro verde indica la región del ground truth, el cuadro rojo indica la región de detección*
Fuente: *elaboración propia.*

De la tabla 5, se desprende que YOLO-NAS genera una región (bounding box), más cercana a la región del ground truth del grupo de imágenes evaluada. Las predicciones TP, FP y FN de la tabla 5 permitieron calcular las métricas de desempeño recall (proporción de verdaderos positivos con respecto a todos los ciclistas presentes en el ground truth) y precision (proporción de verdaderos positivos con respecto a todas las detecciones realizadas). Los resultados se pueden apreciar en la tabla 6.

Tabla 6: *Resultados de evaluación de desempeño con precision y recall.*

Arquitectura	Precision	Recall
YOLOv7	0.6612	0.6695
YOLOv8	0.8854	0.8439
YOLO-NAS	0.8825	0.8836

En la tabla 6 se resaltan los valores de desempeño superior para cada métrica. En términos de precisión, YOLOv8 y YOLO-NAS superan a YOLOv7, lo cual indica que las versiones v8 y NAS entrenadas están generando detecciones con menos falsos positivos y por lo tanto más precisas en la detección de imágenes en las cuáles exista mayor probabilidad de confundir a ciclistas con otros actores viales (motociclistas, por ejemplo). En el caso del recall, es la arquitectura YOLO-NAS la que supera a las demás, por lo cual se puede concluir que este detector es más efectivo en encontrar a todos los ciclistas presentes en la imagen. Esta métrica es la de mayor relevancia para nuestro trabajo debido a que en un entorno donde los vehículos autónomos son los actores viales predominantes en la vía, el mejor detector será aquel que detecte a la mayoría de los ciclistas y reaccionando en consecuencia.

En la figura 5 se presenta la curva P-R (Precision-Recall) para los tres detectores YOLO entrenados, utilizando un umbral IoU de 0.5. Se observa claramente que el algoritmo de detección que alcanza el mejor equilibrio entre recall (0.8836) y precisión (0.8825) es YOLO NAS.

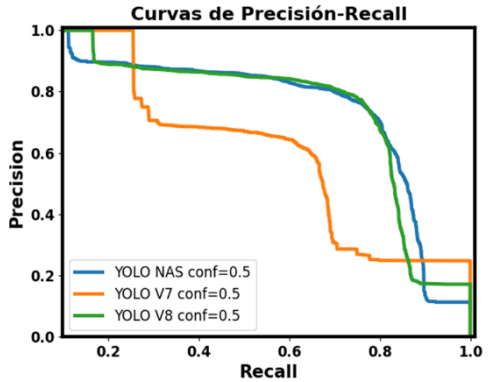

Fig. 5. *Curva Precision-Recall para los detectores YOLO evaluados.*
Fuente: *elaboración propia.*

Es clara la superioridad de YOLO NAS sobre los otros detectores. Por otro lado, YOLOv7 genera una gran cantidad de falsos positivos en comparación con las otras dos arquitecturas.

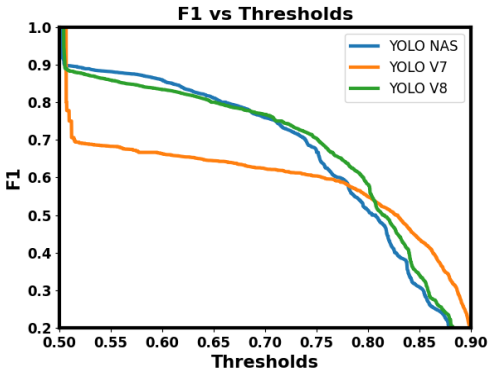

Fig. 6. *Comportamiento de los detectores cuando se varía el umbral (curva F1 vs Umbrales)*
Fuente: *elaboración propia.*

A pesar del buen resultado en la métrica precision para el detector con arquitectura YOLOv8, la cantidad de ciclistas detectados está por debajo de la cantidad detectada por YOLO-NAS.

El comportamiento del sistema cuando se varía el umbral para determinar si una detección es del todo correcta o no, se puede apreciar en la figura 6.

La curva F1 vs. Umbrales, muestra cómo el valor de la métrica F1 varía a medida que se ajusta el umbral de decisión del modelo. A medida que el umbral aumenta la presión de los modelos disminuye, y se puede apreciar que la arquitectura de YOLOv7 tiene un desempeño mucho menor a las otras arquitecturas. El umbral del IoU en 0.5 se presenta como el mejor para decidir si una detección es correcta o no.

Los tiempos de inferencia para cada arquitectura según la cantidad de instancias de ciclistas en las imágenes, se puede apreciar en la tabla 7. Debido a la complejidad de los modelos de las arquitecturas YOLOv8 y YOLO-NAS en comparación con YOLOv7, el tiempo que requiere ésta, para detectar a los ciclistas es menor. El tiempo que requiere YOLO-NAS para detectar varios ciclistas en una imagen es hasta 5 veces mayor que el tiempo requerido por YOLOv7.

Tabla 7: *Tiempo de inferencia (Ti) para cada arquitectura.*

Arquitectura	Ti para 1 ciclista (ms)	Ti varios ciclistas (ms)
YOLOv7	8.3	15.5
YOLOv8	9.4	84.3
YOLO-NAS	29.2412	79.1275

4. CONCLUSIONES Y TRABAJOS FUTUROS

En este artículo, se ha llevado a cabo una evaluación de desempeño de las versiones de la arquitectura YOLO denominadas YOLOv7, YOLOv8 y YOLO-NAS aplicadas a la detección de ciclistas urbanos. Nuestro objetivo ha sido evaluar su efectividad en abordar los desafíos asociados con la detección de ciclistas en vías urbanas en un entorno en el que -eventualmente- los vehículos autónomos serían los actores viales de mayor presencia. Para lograrlo, entrenamos las arquitecturas YOLO con un conjunto de datos combinado (imágenes propias e imágenes obtenidas a partir de webscrapping), etiquetadas manualmente.

Implementamos una estrategia de entrenamiento y pruebas sólido. La estrategia incluyó un entrenamiento con características comunes, incluyendo la base de datos, la cantidad de épocas, los hiper-parámetros y las imágenes de prueba.

En la evaluación de las arquitecturas YOLO se utilizaron las métricas Recall y Precision, a partir de los datos TP, FP y FN, con un IoU del 50% en la detección de ciclistas para 111 imágenes con diferentes características e instancias de ciclistas.

Esta metodología nos permitió establecer una base sólida para comparar el desempeño de las arquitecturas YOLO en cuestión aplicadas a la detección de ciclistas, y obtener conclusiones significativas para una siguiente etapa en el desarrollo de un proyecto de mayor envergadura que incluya tanto a ciclistas como a vehículos autónomos.

Los experimentos realizados, permitieron establecer que la arquitectura YOLO-NAS presentó un mejor desempeño en el recall y en IoU para la región de detección de los ciclistas en las imágenes. YOLOv8 presentó un mejor resultado para la métrica precision, pero estuvo por debajo de YOLO-NAS en el recall. Por otro lado, la arquitectura YOLOv7 estuvo por debajo de las otras dos arquitecturas en ambas métricas, en IoU, en los valores de las predicciones e inclusive en el tiempo de entrenamiento.

En cuanto a los resultados visuales (netamente cualitativos), las arquitecturas presentaron un rendimiento limitado cuando las escenas eran lejanas o tenían malas condiciones de iluminación. Además, en algunos casos, se dejaron de detectar ciclistas cuando estaban parcialmente ocluidos por objetos diferentes o por otros ciclistas. En escenas con condiciones de iluminación favorables, con ciclistas no ocluidos (o con baja oclusión), así como una relación de aspecto común entre ciclista y bicicleta, las arquitecturas YOLOv8 y YOLO-NAS presentaron mejor desempeño que la arquitectura YOLOv7, detectando todas las instancias de los ciclistas.

En un entorno en el cual los vehículos autónomos son el actor vial de mayor presencia en las vías urbanas, un detector de ciclistas debería cumplir con los siguientes requisitos básicos: detectar a todos los ciclistas y hacerlo en el menor tiempo posible. En ese orden de ideas, YOLO-NAS es un detector de excelente desempeño con respecto a las métricas, pero el tiempo que requiere para detectar a un ciclista (o a varios) en una imagen es hasta 5 veces mayor que el detector más rápido (YOLOv7). Esto nos deja en este trabajo, bajo las condiciones de entrenamiento y evaluación llevadas a cabo, que la arquitectura YOLOv8 es la arquitectura con mejor desempeño para detectar ciclistas en vías urbanas desde un vehículo autónomo.

En resumen, cuando se evalúan diferentes arquitecturas para la detección de ciclistas en la vía, se concluye que no hay un mejor modelo en todos los aspectos. La elección del modelo dependerá de las necesidades específicas de la aplicación. Para futuros trabajos, sería beneficioso extender el análisis con otros conjuntos de datos de mayor diversidad y cantidad de imágenes, bajo condiciones de entrenamiento más completas y con máquinas de procesamiento más rápidas.

Los trabajos futuros permitirán hacer un ajuste fino de las arquitecturas presentadas por medio de la optimización de los hiper-parámetros.

En general, este estudio sirve como base para futuros esfuerzos de investigación destinados a mejorar la eficacia de los sistemas de detección de ciclistas desde vehículos autónomos y minimizar los indicadores de ciclistas lesionados o muertos en la vía a causa de colisiones con vehículos motorizados sean o no autónomos.

REFERENCIAS

[1] Flohr, F. B. (2018). Vulnerable Road User Detection and Orientation Estimation for Context-Aware Automated Driving.

[2] Thrun, S. (2010). Toward robotic cars. Communications of the ACM, 53(4), 99–106. https://doi.org/10.1145/1721654.1721679

[3] Alhajyaseen, W. K. M., Asano, M., & Nakamura, H. (2012). Estimation of left-turning vehicle maneuvers for the assessment of pedestrian safety at intersections. IATSS Research, 36(1), 66–74. https://doi.org/10.1016/j.iatssr.2012.03.002

[4] Brohm, T., Haupt, K., & Thiel, R. (2019). Pedestrian Intention and Gesture Classification Using Neural Networks. ATZ Worldwide, 121(4), 26–31. https://doi.org/10.1007/s38311-019-0006-6

[5] Chen, Y. Y., Jhong, S. Y., Li, G. Y., & Chen, P. H. (2019). Thermal-Based Pedestrian Detection Using Faster R-CNN and Region Decomposition Branch. Proceedings - 2019 International Symposium on Intelligent Signal Processing and Communication Systems, ISPACS 2019, 2019–2020. https://doi.org/10.1109/ISPACS48206.2019.8986298

[6] Heo, D., Nam, J. Y., & Ko, B. C. (2019). Estimation of pedestrian pose orientation using soft target training based on teacher-student framework. Sensors (Switzerland), 19(5). https://doi.org/10.3390/s19051147

[7] Lan, W., Dang, J., Wang, Y., & Wang, S. (2018). Pedestrian detection based on yolo network model. Proceedings of 2018 IEEE International Conference on Mechatronics and Automation, ICMA 2018, 1547–1551. https://doi.org/10.1109/ICMA.2018.8484698

[8] Murphey, Y. L., Liu, C., Tayyab, M., & Narayan, D. (2018). Accurate pedestrian path prediction using neural networks. 2017 IEEE Symposium Series on Computational Intelligence, SSCI 2017 - Proceedings, 2018-Janua, 1–7. https://doi.org/10.1109/SSCI.2017.8285398

[9] Fairley, P. (2017). Self-driving cars have a bicycle problem [News]. IEEE Spectrum, 54(3), 12–13.

[10] Mannion, P. (2019). Vulnerable road user detection: state-of-the-art and open challenges. 1–5. http://arxiv.org/abs/1902.03601

[11] Li, X., Flohr, F., Yang, Y., Xiong, H., Braun, M., Pan, S., Li, K., & Gavrila, D. M. (2016). A new benchmark for vision-based cyclist detection. IEEE Intelligent Vehicles Symposium, Proceedings, 2016-Augus(Iv), 1028–1033. https://doi.org/10.1109/IVS.2016.7535515

[12] Kress, V., Jung, J., Zernetsch, S., Doll, K., & Sick, B. (2019). Pose Based Start Intention Detection of Cyclists. 2019 IEEE Intelligent Transportation Systems Conference, ITSC 2019, 2381–2386. https://doi.org/10.1109/ITSC.2019.8917215

[13] Garcia-Venegas, M., Mercado-Ravell, D. A., Pinedo-Sanchez, L. A., & Carballo-Monsivais, C. A. (2021). On the safety of vulnerable road users by cyclist detection and tracking. Machine Vision and Applications, 32(5), 109.

[14] Casas, E., Ramos, L., Bendek, E., & Rivas-Echeverría, F. (2023). Assessing the Effectiveness of YOLO Architectures for Smoke and Wildfire Detection. IEEE Access.

[15] BITZI, software image scraping. (2017). Instituto Tecnológico Metropolitano. Medellín. https://doi.org/10.1109/mspec.2017.7864743

[16] Shijie, J., Ping, W., Peiyi, J., & Siping, H. (2017, October). Research on data augmentation for image classification based on convolution neural networks. In 2017 Chinese automation congress (CAC) (pp. 4165-4170). IEEE.

[17] Wickramanayake, S., Hsu, W., & Lee, M. L. (2021). Explanation-based data augmentation for image classification. Advances in Neural Information Processing Systems, 34, 20929-20940.

[18] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 779-788).Clymer, J. R. (1992). "Discrete Event Fuzzy Airport Control". IEEE Trans. On Systems, Man, and Cybernetics, Vol. 22, No. 2.

[19] Wang, C. Y., Bochkovskiy, A., & Liao, H. Y. M. (2023). YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 7464-7475).

[20] Yasir, M., Zhan, L., Liu, S., Wan, J., Hossain, M. S., Isiacik Colak, A. T., ... & Yang, Q. (2023). Instance segmentation ship detection based on improved Yolov7 using complex background SAR images. Frontiers in Marine Science, 10, 1113669.

[21] Jocher, G., Chaurasia, A., & Qiu, J. (2023). YOLO by Ultralytics. URL: https://github.com/ultralytics/ultralytics.

[22] Xia, K., Lv, Z., Zhou, C., Gu, G., Zhao, Z., Liu, K., & Li, Z. (2023). Mixed Receptive Fields Augmented YOLO with Multi-Path Spatial Pyramid Pooling for Steel Surface Defect Detection. Sensors, 23(11), 5114.

[23] Liu, Y., Sun, Y., Xue, B., Zhang, M., Yen, G. G., & Tan, K. C. (2021). A survey on evolutionary neural architecture search. IEEE transactions on neural networks and learning systems.

[24] Padilla, R., Netto, S. L., & Da Silva, E. A. (2020, July). A survey on performance metrics for object-detection algorithms. In 2020 international conference on systems, signals and image processing (IWSSIP) (pp. 237-242). IEEE.

[25] Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. Advances in neural information processing systems, 28.

[26] Li, X., Li, L., Flohr, F., Wang, J., Xiong, H., Bernhard, M., Pan, S., Gavrila, D. M., & Li, K. (2017). A unified framework for concurrent pedestrian and cyclist detection. IEEE Transactions on Intelligent Transportation Systems, 18(2), 269–281. https://doi.org/10.1109/TITS.2016.2567418

[27] Lin, Y., Wang, P., & Ma, M. (2017). Intelligent Transportation System(ITS): Concept, Challenge and Opportunity. Proceedings - 3rd IEEE International Conference on Big Data Security on Cloud, BigDataSecurity 2017, 3rd IEEE International Conference on High Performance and Smart Computing, HPSC 2017 and 2nd IEEE International Conference on Intelligent Data and Securit, 167–172. https://doi.org/10.1109/BigDataSecurity.2017.50

[28] World Health Organization. (2018). Global status report on road safety 2018. In Global status report on road safety 2018: Summary (No. WHO/NMH/NVI/18.20).