UNIVERSIDAD DE CONCEPCIÓN
FACULTAD DE INGENIERÍA
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

Detector de acciones de riesgo en video utilizando inteligencia artificial y transfer learning

POR

Camilo Pablo Andrés Domínguez Monroy

Memoria de Título presentada a la Facultad de Ingeniería de la Universidad de Concepción para optar al título profesional de Ingeniero Civil en Telecomunicaciones

Profesor Guía
Sebastián Godoy Medel

Enero 2025
Concepción (Chile)

A todos los estudiantes de Telecom...

Resumen

El presente proyecto aborda el desarrollo de un sistema inteligente para la detección automática de peleas en tiempo real mediante inteligencia artificial a través de cámaras de videovigilancia, utilizando el modelo de segmentación YOLOv8. A diferencia de enfoques previos centrados principalmente en la detección de objetos como armas u otros elementos físicos, esta propuesta se enfoca en identificar comportamientos violentos a partir de interacciones humanas, lo que representa un desafío técnico mayor debido a la sutileza y ambigüedad de muchos de estos eventos.

Para entrenar el modelo, se construyó un conjunto de datos compuesto por imágenes extraídas de videos reales, los cuales presentaban limitaciones como baja resolución, movimientos de cámara bruscos y condiciones de iluminación variables. El entrenamiento se complementó con reglas basadas en la distancia entre centroides de personas y un conteo mínimo de dos individuos para que una escena sea considerada como pelea. Esto permitió compensar parcialmente las limitaciones del dataset y mejorar la precisión del sistema.

La metodología aplicada prioriza la eficiencia computacional, utilizando una versión ligera del modelo capaz de operar en hardware de gama media.

Los resultados demuestran que el modelo es capaz de identificar situaciones de conflicto con una precisión aceptable, abriendo posibilidades para su implementación práctica en sistemas de seguridad ciudadana más inteligentes, y que, además su bajo requerimiento computacional lo hace compatible con dispositivos de bajo costo y proactivos.

Agradecimientos

A mi familia, por su constante apoyo y por ser una gran fuente de motivación durante todo este proceso. A mis amigos de la universidad, por los momentos compartidos que llenaron este camino de alegría, y a mis amigos del colegio, por demostrar que la verdadera amistad perdura a pesar de haber tomado rumbos distintos. Y, por último, pero no menos importante, a Mónica, por hacer mi tiempo universitario más ameno y por enseñarme que allí no solo hallé conocimientos, sino también el amor.

Glosario

R-CNN: Regions with Convolutional Neural Networks [6] es un modelo de detección de objetos propuesto por Ross Girshick en 2014. Representa uno de los primeros enfoques exitosos para integrar redes neuronales convolucionales en la detección de objetos, combinando la localización y clasificación de objetos en imágenes.

CSPDarknet53: Cross Stage Partial Darknet53 [20] es una arquitectura de red neuronal convolucional utilizada como backbone (esqueleto base para la extracción de características) en modelos avanzados de detección de objetos, como YOLOv4 donde este utiliza capas convolucionales profundas y conexiones residuales, lo que proporciona una buena capacidad de extracción de características.

auto-adversarial (SAT): Self-Adversarial Training [21] es una técnica de entrenamiento que hace que el modelo "luche contra sí mismo" durante una etapa previa al entrenamiento. Es un proceso en dos pasos primero está la Etapa de ataque adversarial donde el modelo altera las imágenes de entrada de manera que se hace más difícil para él mismo detectar correctamente los objetos, después de esto viene la etapa de entrenamiento normal donde, entrena sobre esas imágenes adversarialmente modificadas. Esto obliga al modelo a aprender características más robustas que le permitan detectar objetos incluso cuando están parcialmente ocultos.

Mosaic: Es una técnica de data augmentation [22] diseñada para mejorar la generalización del modelo y aumentar la robustez frente a condiciones visuales variables. Consiste en combinar cuatro imágenes diferentes en una sola, colocándolas en un mosaico dividido en cuatro cuadrantes.

CIoU: Complete Intersection over Union [23] es una métrica utilizada en tareas de detección de objetos para mejorar la precisión donde mide cuanto se solapan (intersección) dos cajas (la predicha y la verdadera) en relación con su unión. Es una evolución de otras métricas anteriores como IoU, GIoU y DIoU.

mAP: mean Average Precision [24] es una de las métricas más importantes para evaluar el rendimiento de un modelo de detección de objetos que se basa en dos conceptos clave que son, precisión qué significa la proporción de las detecciones que fueron correctas y recall qué, es la proporción de los objetos reales que fueron detectados.

Asignación dual de predicciones: es una técnica introducida en YOLOv10 [25] para mejorar la asociación entre las predicciones del modelo y los objetos reales durante el entrenamiento. Combina dos estrategias de asignación, una basada en la ubicación (proximidad al centro del objeto) y otra basada en similitud de características (embeddings), permitiendo una correspondencia más robusta y precisa, especialmente en escenas complejas o con objetos superpuestos.

Non-Maximum Suppression (NMS): es una técnica de detección de objetos [26], utilizada para eliminar detecciones redundantes o superpuestas, dejando solo la más precisa entre ellas. Es como un "filtro inteligente" que ayuda a decidir cuál caja de predicción conservar cuando varias se solapan demasiado detectando el mismo objeto.

SPP: Spatial Pyramid Pooling [5] es una técnica utilizada en redes neuronales convolucionales para permitir la extracción de características a múltiples escalas sin necesidad de redimensionar las imágenes de entrada. El módulo SPP divide la característica extraída de una imagen en regiones de diferentes tamaños, aplica operaciones de pooling en cada región, y luego concatena los resultados en una sola representación. Esto permite que la red sea más robusta a objetos de distintos tamaños y proporciones, mejorando el rendimiento en tareas de detección y clasificación sin perder la relación espacial de los objetos.

Pooling: Operación empleada en redes neuronales convolucionales (CNN) [27] para reducir la dimensión espacial de una imagen o mapa de características, conservando la información más relevante. Esta técnica no solo disminuye el costo computacional, sino que también mejora la robustez del modelo frente a variaciones, desplazamientos o distorsiones leves en los datos de entrada.

Data Augmentation (Aumento de Datos): Técnica utilizada en el entrenamiento de modelos de aprendizaje automático que consiste en generar nuevas muestras de datos a partir de las originales mediante transformaciones como rotaciones, recortes, volteos, cambios de brillo o escalado. Su objetivo es aumentar la diversidad del conjunto de datos, mejorar la generalización del modelo y reducir el sobreajuste (overfitting), especialmente cuando se dispone de pocos datos de entrenamiento.

Época: Una época representa un ciclo completo en el que el modelo analiza todo el conjunto de datos de entrenamiento, retroalimentándose con los errores cometidos para mejorar su precisión en la siguiente iteración.

Tabla de Contenidos

Lista de Figuras

1. Introducción

1.1. Introducción general

La seguridad en entornos urbanos es un tema de creciente preocupación, especialmente debido al aumento de incidentes violentos en espacios públicos. En este contexto, los sistemas de videovigilancia han evolucionado, integrando algoritmos de inteligencia artificial para la detección automatizada de situaciones de riesgo. Entre estos, YOLO se ha consolidado como uno de los enfoques más eficientes para la identificación de objetos y comportamientos en tiempo real, gracias a su capacidad de análisis rápido y preciso de imágenes [4].

En el contexto actual, los instrumentos de medición aplicados a la ciudadanía reflejan una alta percepción de inseguridad. Un ejemplo de ello es la Encuesta Nacional Urbana de Seguridad Ciudadana (ENUSC) 2023, cuyos resultados indican que más del 90,6 % de la población comparte esta percepción [1]. En este escenario, contar con tecnologías capaces de identificar y alertar sobre eventos violentos resulta crucial para prevenir situaciones que puedan derivar en delitos o agravar la sensación de inseguridad.

Este proyecto se enfoca en el desarrollo de un sistema que detecta situaciones de riesgo físico (peleas) a través de cámaras de videovigilancia, diferenciando entre los implicados y los que no involucrados, con el objetivo de mejorar la capacidad de respuesta ante estos incidentes.

Este proyecto se centra en el desarrollo de un sistema de detección de peleas en tiempo real utilizando el modelo YOLOv8. La idea se origina por la inviabilidad que supone el control humano de sistemas de videovigilancia, ya que los enfoques tradicionales que dependen de la supervisión constante de operadores humanos frente a las cámaras de seguridad. Esta propuesta busca automatizar la identificación de situaciones de conflicto, para mejorar la seguridad, ya que la implementación del modelo no solo mejora la capacidad de respuesta ante incidentes violentos en espacios vigilados, sino que también optimiza los recursos disponibles, permitiendo una reacción más rápida y eficiente. Además, reduce significativamente la necesidad de grandes equipos de monitoreo humano, lo que disminuye los costos operativos y minimiza el margen de error asociado a la fatiga o distracción de los vigilantes.

Para lograr este objetivo, se emplea un enfoque basado en el uso de datos extraídos de X (anteriormente Twitter) y del *dataset* especializado de Aktı et al. (2022) [2]. Con el propósito de mejorar la calidad del etiquetado, se reemplazó el uso de LabelMe por la plataforma Roboflow, lo que permite una segmentación semiautomática más precisa y eficiente. Además, se implementan técnicas

avanzadas, como se indicará más adelante, de preprocesamiento de datos y optimización de hiperparámetros para mejorar el desempeño del modelo de detección.

1.2. Introducción a YOLO

YOLO es un algoritmo de detección de objetos en tiempo real que revolucionó el campo de la visión por computador desde su primera aparición en 2016, el que fue desarrollado por Joseph Redmon [3]. Su principal innovación fue abordar la detección de objetos como un único problema de regresión, permitiendo localizar y clasificar objetos en una sola captura por la red neuronal. A diferencia de otros enfoques más lentos que utilizan propuestas de regiones, YOLO permite una detección rápida, ideal para aplicaciones en tiempo real como videovigilancia, conducción autónoma o monitoreo de seguridad.

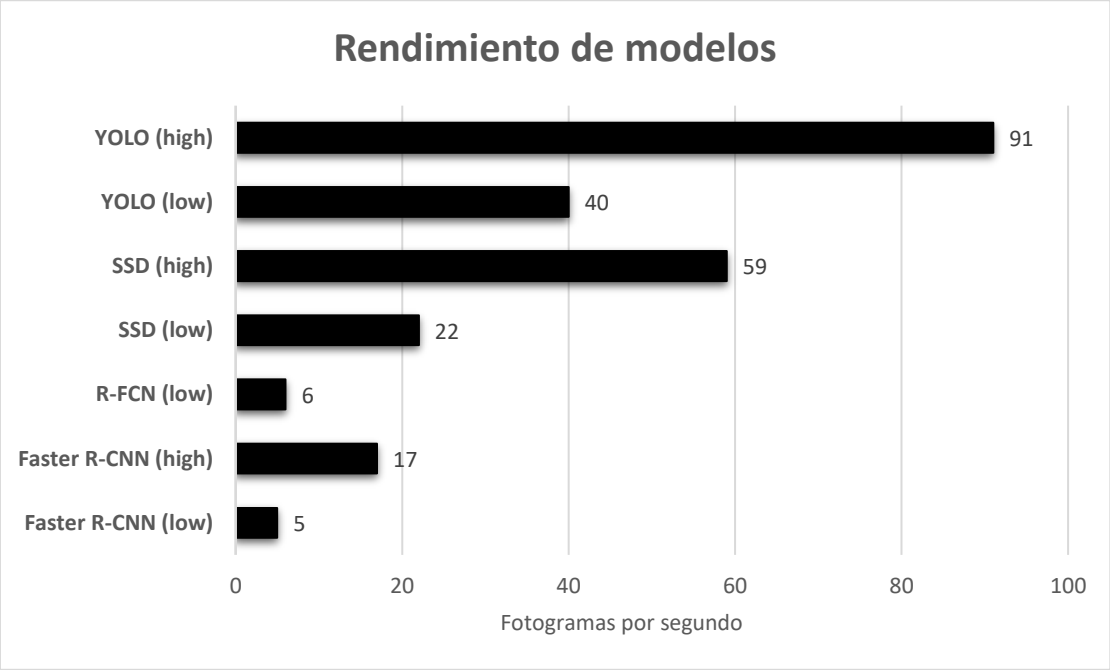

Figura 1.1 Gráfico rendimiento de modelos [4]

La figura presentada (1.1) compara el rendimiento de varios algoritmos de detección de objetos en términos de velocidad, medida en frames por segundo (FPS). Se observa que YOLO (high) alcanza el mayor rendimiento con 91 FPS, seguido por SSD (high) con 59 FPS, lo que implica una mayor capacidad para procesar imágenes en tiempo real. En contraste, algoritmos como Faster R-CNN (low) con 5 FPS y R-FCN (low) con 6 FPS muestran velocidades considerablemente menores, reflejando su enfoque hacia una detección más precisa, pero menos inmediata.

Esta diferencia resalta porqué YOLO se ha popularizado en aplicaciones que requieren alta eficiencia, como la videovigilancia o la detección de eventos en tiempo real, ofreciendo un equilibrio entre precisión y rapidez.

Durante el entrenamiento de una red como YOLO, no solo se ajustan los valores de las capas internas para reconocer patrones generales (como formas humanas o movimientos agresivos) [28], sino que también se entrena especialmente la capa final, que es la encargada de realizar las predicciones específicas. Esta última capa transforma toda la información aprendida a lo largo de la red en salidas concretas, como en este caso de si corresponde a una pelea o no.

1.3. Arquitectura y funcionamiento de YOLO

YOLO es una arquitectura de red neuronal convolucional, diseñada para realizar detección de objetos o acciones en tiempo real. A diferencia de otros enfoques tradicionales, YOLO unifica todo en un solo paso, en el cual predice simultáneamente las clases y las ubicaciones de los objetos en una única pasada por la red, lo que le permite ser rápido y eficiente en comparación con otros modelos de detección precitados.

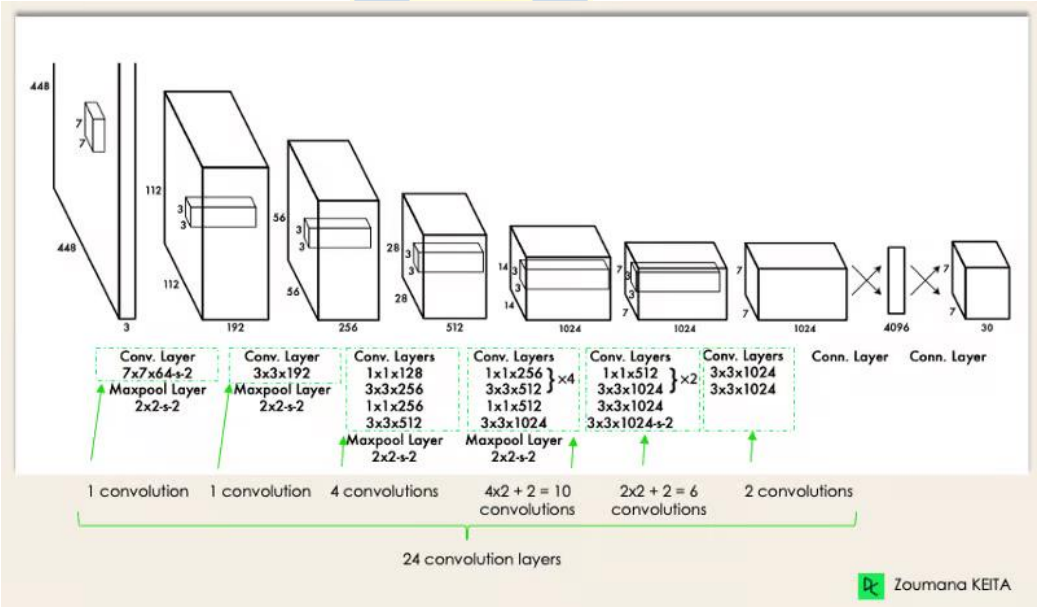

Figura 1.2 Gráfico de redes convolucionales [4]

Como se aprecia en la figura 1.2, se identifica el proceso de la imagen una vez procesada cuando capta YOLO el *frame*, atraviesa varias capas convolucionales que extraen características visuales relevantes, como bordes, formas y texturas. Durante este proceso, se aplican técnicas adicionales como la normalización por lotes, que estabiliza el entrenamiento, el dropout, que ayuda a prevenir el

sobreajuste y módulos como el Spatial Pyramid Pooling (SPP) [5], que extraen información a diferentes escalas espaciales.

Además, YOLO utiliza un enfoque basado en dividir la imagen en una cuadrícula (grilla), donde cada celda es responsable de realizar predicciones sobre la presencia de información dentro de su área.

Por cada celda, el modelo genera múltiples predicciones que incluyen coordenadas de las cajas delimitadoras (bounding boxes). Un valor de confianza que indicaría las probabilidades de clase que estas determinan qué tipo de objeto es.

Posteriormente, el modelo combina todas las predicciones generadas en las distintas celdas, aplicando técnicas como Non-Maximum Suppression (NMS), que filtra las mejores detecciones evitando duplicados.

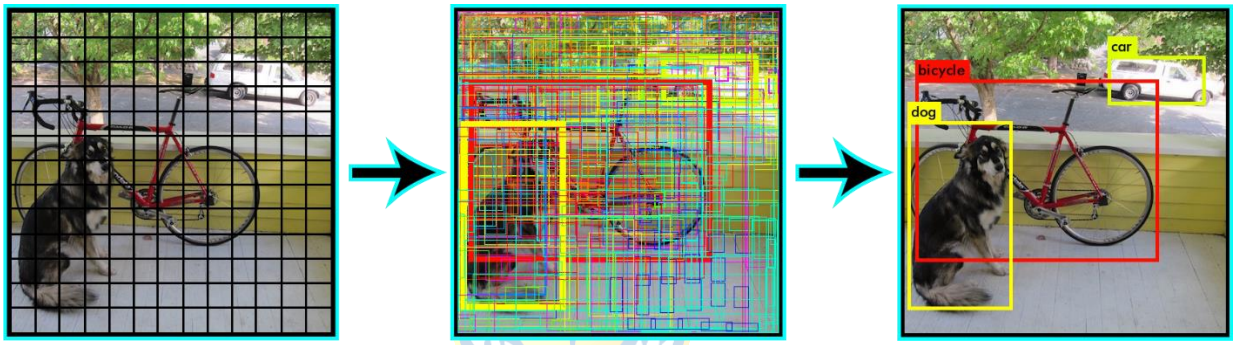

Figura 1.3 Gráfico explicativo grillado [28]

De este modo, el sistema no solo evalúa localmente cada zona de la imagen, sino que construye una visión global coherente, seleccionando las predicciones más precisas y relevantes, logrando una detección eficiente y en tiempo real.

1.4. Trabajos previos

Desde la introducción del modelo YOLO por Redmon et al. en 2016 [3], la detección de objetos experimentó un cambio de paradigma. A diferencia de modelos basados en regiones propuestas, como R-CNN o Fast R-CNN [6], YOLO plantea un enfoque unificado, en el que una única red neuronal convolucional realiza tanto la localización como la clasificación de los objetos en una sola pasada. Esto redujo considerablemente los tiempos de procesamiento, posibilitando aplicaciones en entornos de vigilancia en tiempo real.

La evolución de YOLO ha sido constante y significativa, prueba de ello es que hoy se cuenta con la versión 12. En 2020, YOLOv4 [7] marcó un hito importante al incorporar técnicas como CSPDarknet53 como backbone, entrenamiento auto-adversarial (SAT), aumento de datos con Mosaic y la función de pérdida CIoU. Estas mejoras permitieron alcanzar un equilibrio entre precisión y velocidad, incluso sin necesidad de hardware especializado. Las versiones siguientes, como YOLOv6 [8] y las variantes YOLOv7 a YOLOv9, continuaron esta línea evolutiva con optimizaciones arquitectónicas enfocadas en entornos industriales, mejorando su rendimiento (mAP) y robustez.

Recientemente, YOLOv10 [9] introdujo una innovación significativa al eliminar el proceso tradicional de Non-Maximum Suppression (NMS), reemplazándolo por una estrategia de asignación dual de predicciones. Esta técnica redujo la latencia y mejoró la consistencia temporal, reforzando el potencial del algoritmo en aplicaciones críticas de tiempo real.

Junto a estas versiones oficiales, han emergido múltiples variantes especializadas que adaptan YOLO a distintos dominios. Por ejemplo, YOLO-World [10] permite detectar objetos fuera del conjunto de clases entrenadas mediante un vocabulario abierto, gracias al uso de modelado visión-lenguaje. DC-SPP-YOLO [11], en tanto, optimiza la detección multiescala, una capacidad crucial en escenas heterogéneas. En el ámbito industrial, Nikam et al. [12] emplearon versiones YOLOv7 a v9-GELAN para detectar defectos en procesos de manufactura aditiva, alcanzando una precisión del 95.7%.

El potencial de YOLO también se ha proyectado hacia escenarios no convencionales. Dynamic YOLO [13], por ejemplo, fue diseñado para la detección de objetos en ambientes submarinos, incorporando convoluciones deformables y atención multiescala para abordar los desafíos del entorno marino. En contextos urbanos y aéreos, HF-YOLO [14] y LUD-YOLO [15] fueron optimizados para detectar peatones en zonas densamente pobladas o vehículos desde drones en movimiento, facilitando la vigilancia automatizada desde plataformas móviles.

Por su parte, HIC-YOLOv5 [16] mejoró la detección de objetos pequeños mediante la incorporación de una cabeza de predicción adicional y mecanismos de atención específicos. Estas contribuciones refuerzan la versatilidad del ecosistema YOLO, permitiendo su aplicación en una amplia gama de sectores, desde la seguridad urbana hasta la agricultura, el ámbito marítimo o la industria.

Más allá de sus cualidades técnicas, el impacto social de estas tecnologías se manifiesta en su capacidad concreta para aumentar la seguridad pública. Un ejemplo destacado es el trabajo de Schcolnik-Elias et al. [17], quienes desarrollaron un sistema de detección de armas de fuego en espacios públicos usando YOLO junto a visión estereoscópica, logrando una precisión del 92.2% y estimando distancias con un margen de error de apenas 9.3 cm. Este tipo de funcionalidad mejora la capacidad de respuesta de las autoridades y aporta información contextual clave en situaciones de emergencia.

Por otro lado, José Antonio Alcaide Recio [18] desarrolló un sistema de inteligencia artificial capaz de identificar situaciones adversas en entornos urbanos, el cual fue implementado sobre una arquitectura DevOps que facilita su despliegue rápido en redes de videovigilancia. Este enfoque no solo aporta agilidad operativa, sino que también permite escalar su implementación a nivel de ciudad inteligente.

Complementando estos desarrollos, Sagrario García-García y Raúl Pinto-Elías [19] utilizaron YOLO para detectar comportamientos humanos en tiempo real, enfocándose en la identificación de conductas anómalas tanto en espacios públicos como en ambientes laborales. Este tipo de análisis, más allá de la detección de objetos, permite comprender dinámicas sociales y prevenir eventos de riesgo antes de que escalen.

Incluso en áreas rurales, YOLO ha demostrado su capacidad de adaptación. Xiong et al. [20] utilizaron esta arquitectura para identificar plantas de marihuana en cultivos agrícolas, revelando que, más allá del contexto, la solidez del modelo permite la detección efectiva de patrones visuales complejos.

En cuanto a la detección de conflictos físicos, como peleas en espacios públicos, el trabajo de Aktı et al. [2] resulta fundamental. Su desarrollo de un conjunto de datos específico para entrenar modelos en la identificación de peleas ha contribuido a aumentar la sensibilidad de los sistemas frente a eventos que suelen ser breves y difíciles de captar para un observador humano.

La detección de peleas implica identificar patrones de comportamiento que no siempre son evidentes ni visualmente consistentes, en muchos casos, los movimientos pueden ser ambiguos, parcialmente ocultos o confundibles con interacciones no agresivas. Estudios recientes han comenzado a explorar este tipo de reconocimiento de acciones violentas mediante la incorporación de información temporal (videoclips en lugar de imágenes estáticas), el uso de redes recurrentes o modelos CNN [29], pero el enfoque aún es incipiente en comparación con la detección de objetos.

El presente proyecto plantea una contribución en esta línea emergente, utilizando un modelo YOLOv8 adaptado para la detección de peleas en tiempo real. A diferencia de los trabajos anteriores centrados en elementos visuales estáticos, se propone entrenar el modelo para identificar patrones asociados a conductas agresivas, como contacto físico repentino, agrupaciones irregulares de personas o posturas corporales violentas, en grabaciones de videovigilancia. Este cambio de enfoque representa un paso hacia sistemas de seguridad más inteligentes y contextualmente informados, capaces no sólo de ver qué hay en una imagen, sino también de interpretar qué está ocurriendo, lo que podría mejorar sustancialmente la capacidad de prevención y respuesta ante eventos violentos en espacios públicos o privados.

En conjunto, la evolución de YOLO no solo ha consolidado una base técnica sólida, sino que también ha abierto el camino hacia sistemas de videovigilancia verdaderamente inteligentes, capaces no solo de ver, sino de interpretar el entorno, actuar con rapidez y contribuir de forma efectiva a la seguridad pública y social.

1.5. Discusión

La revisión de la literatura confirma que la evolución del algoritmo, desde su introducción por Redmon et al. en 2016 [3], ha supuesto un antes y un después en el campo de la detección de objetos en tiempo real. A diferencia de los enfoques anteriores como R-CNN, que empleaban una arquitectura en dos etapas (primero la propuesta de regiones y luego la clasificación), YOLO consolidó una estructura *one-stage* que realiza la localización y clasificación simultáneamente en una sola captura por la red. Esta reformulación no solo mejoró la velocidad de procesamiento, sino que la hizo apta para aplicaciones en entornos dinámicos como la videovigilancia.

Las versiones posteriores, desde YOLOv4 hasta YOLOv10, han ido incorporando mejoras fundamentales tanto a nivel de arquitectura como de eficiencia computacional. Por ejemplo, YOLOv4 introdujo el backbone CSPDarknet53, una variante que divide los flujos de características para reducir el costo computacional sin pérdida de precisión [20]. También se añadieron innovaciones como Spatial Pyramid Pooling (SPP) [5] para capturar objetos a distintas escalas y el Self-Adversarial Training (SAT) [21], una técnica de entrenamiento en la que el propio modelo se desafía generando perturbaciones adversariales para fortalecer su robustez.

A nivel de aprendizaje, YOLO ha incorporado funciones de pérdida avanzadas como Complete IoU (CIoU), que no solo considera la superposición entre las cajas predicha y real, sino también su distancia euclidiana y proporciones, mejorando la calidad de la regresión [23]. En paralelo, técnicas de *data augmentation* como Mosaic han sido esenciales para mejorar la generalización, combinando cuatro imágenes aleatorias en una sola durante el entrenamiento, lo cual amplía la diversidad de escenarios sin necesidad de recolectar nuevos datos [22].

Con YOLOv10 [9], se alcanza un nuevo punto de madurez. Se elimina el tradicional paso de Non-Maximum Suppression (NMS) [26] una técnica usada para eliminar detecciones duplicadas gracias a una arquitectura basada en una asignación dual de predicciones. Esta estrategia permite asignar las predicciones simultáneamente en función de su posición y sus características, logrando una identificación más coherente de los objetos en escenas complejas.

Desde el punto de vista aplicado, el uso de YOLO en seguridad ciudadana ha sido amplio y exitoso. Por ejemplo, Schcolnik-Elias et al. [16] aplicaron YOLO con visión estereoscópica para la detección de armas, alcanzando una precisión del 92.2 % y la capacidad de estimar distancias con apenas 9.3 cm de error. Esto implica una mejora sustancial en la toma de decisiones por parte de agentes de seguridad. Paralelamente, trabajos como el de Alcaide Recio [17] muestran cómo estas tecnologías pueden integrarse en plataformas DevOps, reduciendo la latencia en despliegues urbanos.

Las peleas en espacios públicos suelen ser el primer eslabón en situaciones más graves, como robos, agresiones armadas o incluso delitos de alta connotación social. En ese sentido, contar con sistemas de detección temprana de conflictos representa una medida preventiva de gran valor para la seguridad ciudadana. Según el informe N.º 39 “Claves Ipsos” [30], elaborado junto a la Fundación Paz Ciudadana, un 66% de la población chilena considera la delincuencia como la principal prioridad del Gobierno, mientras que un 76% percibe que los problemas de seguridad han aumentado en el último año. Frente a este escenario, este trabajo propone una herramienta práctica y funcional que, mediante el uso de inteligencia artificial y visión por computador, permite detectar situaciones de pelea en tiempo real.

A nivel de comportamiento humano, investigaciones como la de Aktı et al. [2] demuestran que YOLO no se limita a la detección de objetos estáticos, sino que puede entrenarse para identificar patrones de movimiento asociados a violencia, como peleas, gracias a conjuntos de datos específicos. Esto representa un enfoque más proactivo para la videovigilancia, donde no solo se identifican amenazas físicas (como armas), sino también situaciones de riesgo basadas en la conducta.

Sin embargo, estos avances no están exentos de limitaciones. En primer lugar, muchos modelos basados en YOLO requieren hardware especializado (GPUs) para operar con la fluidez que prometen, en sistemas con recursos limitados, esto puede representar una barrera importante. Además, aunque técnicas como Mosaic mejoran la generalización, su efectividad puede disminuir en escenarios con alta variabilidad de resolución, iluminación o calidad de video, donde la dependencia de la cámara y el dataset se vuelve crítica. Por último, la integración eficiente de estos modelos en redes de videovigilancia reales requiere una adaptación cuidadosa para evitar cuellos de botella en el procesamiento y transmisión de datos, sobre todo en sistemas distribuidos o con restricciones de ancho de banda.

Otro punto relevante es la versatilidad que ha adquirido YOLO frente a otros retos no urbanos. Aplicaciones como la detección de marihuana en cultivos (Xiong et al. [19]) o la inspección submarina (Zhang et al. [11]) prueban que el algoritmo puede ser afinado para contextos muy diversos. Esta capacidad de adaptación también se ve reflejada en modelos como YOLO-World [10], que, mediante una arquitectura multimodal de visión-lenguaje, es capaz de detectar objetos que no están previamente definidos en el conjunto de entrenamiento, abriendo la puerta a sistemas más flexibles y menos dependientes de datos etiquetados.

En resumen, el ecosistema YOLO ha consolidado su papel no solo como una herramienta técnica poderosa, sino como un componente socialmente transformador. Su aplicación en

videovigilancia permite responder de forma temprana y automatizada a amenazas reales, desde la detección de armas hasta comportamientos violentos. A medida que sus limitaciones técnicas se reducen gracias a modelos más livianos, métodos de entrenamiento más robustos y arquitecturas más eficientes, se acerca cada vez más el objetivo de implementar sistemas de vigilancia realmente inteligentes, escalables.

1.6. Objetivos

1.6.1 Objetivo General

Desarrollar un sistema de detección de peleas en tiempo real utilizando el modelo YOLO, capaz de alcanzar una precisión aproximada del 70% mediante el análisis de imágenes captadas por cámaras de seguridad, que sea capaz de identificar peleas para mejorar la seguridad en espacios públicos y privados a través de cámaras de videovigilancia.

1.6.2 Objetivos Específicos

- **Recopilación del *dataset***: Recopilar un conjunto de datos robusto para el entrenamiento del modelo, utilizando imágenes de redes sociales y bases de datos académicas.
- **Entrenamiento y ajuste del modelo:** Entrenar el sistema utilizando el conjunto de datos y ajustar los parámetros del algoritmo para mejorar la precisión en la identificación de peleas.
- **Pruebas del sistema:** Realizar pruebas del sistema en entornos simulados dentro de la universidad y con conjuntos de datos de videovigilancia, evaluando su rendimiento y precisión.
- **Correcciones y ajustes finales del sistema:** Incorporar las correcciones necesarias y realizar los ajustes finales para asegurar que el sistema cumple con los requisitos establecidos.

1.7. Metodología

Para desarrollar el modelo propuesto, la metodología se estructuró en cuatro etapas principales. Primero, se realizó la recopilación y preparación de un conjunto de datos representativo, compuesto por imágenes etiquetadas manualmente. Luego, se entrenó el modelo YOLOv8 utilizando Python, ajustando sus parámetros para mejorar su rendimiento. En la tercera etapa, se efectuaron pruebas en entornos simulados y reales para evaluar su precisión. Finalmente, se aplicaron ajustes finos al sistema en base a los resultados obtenidos, con el fin de asegurar una detección efectiva y en tiempo real de peleas.

OE1: Recopilación del dataset

1. **Revisión y búsqueda de datos reales**: Se realizó una investigación de fuentes públicas para encontrar videos que mostraran peleas reales. Se utilizaron principalmente redes sociales como *X*, donde se extrajeron manualmente escenas que presentaban situaciones de conflicto.
2. **Extracción de imágenes:** Los videos recopilados fueron divididos en *frames* relevantes, acumulando más de 3000 imágenes en un proceso que tomó aproximadamente 6 meses.
3. **Etiquetado manual y semiautomático:** Cada imagen fue etiquetada a mano utilizando dos clases: Pelea y No-Pelea. Posteriormente, se utilizó la herramienta Roboflow para facilitar el etiquetado semiautomático y acelerar el proceso sin perder precisión.
4. **Data Augmentation:** Se aplicaron técnicas de aumento de datos como rotaciones, espejado, inclinaciones y ajustes de brillo/contraste, lo cual mejoró la capacidad del modelo.

OE2: Entrenamiento y ajuste del modelo

1. **Selección del modelo YOLOv8:** Se eligió YOLOv8 por su buen balance entre precisión y velocidad, además de ser adecuado para las limitaciones del hardware utilizado.
2. **Configuración del entrenamiento:** El modelo fue entrenado durante 300 épocas, con *Early Stopping* para detener el entrenamiento si no se observaban mejoras tras 20 iteraciones consecutivas.
3. **Evaluación durante el entrenamiento:** Durante el proceso, se validaron los resultados usando imágenes que mostraban gráficamente la comprensión del modelo. Se utilizaron indicadores "1" y "0" en las esquinas de las imágenes para visualizar si el modelo detectaba correctamente una pelea o no.
4. **Centroides:** Se calculó el punto medio de las *bounding box* para estimar la cercanía entre individuos. Si la distancia entre dos centroides era menor a 10 píxeles, se interpretaba como posible interacción física.

5. **Conteo de personas**: Se programó el sistema para detectar una pelea solo si se identificaban al menos dos personas en el cuadro. Esto ayudó a evitar falsos positivos.

OE3: Pruebas del sistema

1. **Simulación de escenarios reales:** Se realizaron pruebas en las dependencias de la universidad con peleas actuadas, evaluando el rendimiento en condiciones reales de cámaras de vigilancia.
2. **Evaluación en distintos entornos:** Se procesaron videos de peleas reales y simuladas, observando el rendimiento tanto en entornos controlados como en grabaciones espontáneas. Se obtuvo una precisión superior al 70% en detecciones en tiempo real.
3. **Detección en vivo con cámaras del DTI:** Se capturaron escenas en tiempo real desde cámaras universitarias para validar el sistema, confirmando su viabilidad operativa.

OE4: Ajustes finales y optimización del sistema

1. **Corrección de errores y reentrenamiento:** Se analizaron casos de falsos positivos y negativos, corrigiendo etiquetas mal clasificadas y entrenando nuevamente el modelo con las mejoras.
2. **Ajustes en umbrales de confianza:** Se ajustaron los parámetros del modelo para aumentar su precisión en casos ambiguos y reducir las detecciones erróneas.
3. **Evaluación final:** El sistema fue validado realizando una deteccion en tiempo real dentro de las dependencias de la universidad el cual se profundizará más adelante.

1.7.1 Recopilación del *dataset* y su impacto en el desempeño del modelo

La calidad y diversidad del conjunto de datos es uno de los factores más determinantes en el rendimiento de un modelo de inteligencia artificial. En tareas complejas como la detección de peleas en entornos reales, contar con información representativa y etiquetada correctamente es fundamental para que el modelo pueda aprender a distinguir comportamientos agresivos de actividades normales.

Una de las principales dificultades radica en la escasa disponibilidad de bases de datos o la calidad de los mismos. Las peleas, al ser eventos esporádicos y, muchas veces, de corta duración, no suelen estar ampliamente documentadas, frente a esta limitación, la estrategia de búsqueda de imágenes y clips en redes sociales resultó ser una herramienta valiosa. A través de esta plataforma fue posible recopilar videos reales de enfrentamientos, publicados por testigos o medios de comunicación, lo que permitió reunir ejemplos auténticos y variados de peleas en espacios públicos.

Además, se incorporó el *dataset* propuesto por Aktı et al. (2022), Este *dataset* contiene secuencias de video especializada en este tipo de situaciones, por lo que la combinación de material de redes sociales con fuentes académicas enriqueció el conjunto de entrenamiento, mejorando la capacidad del sistema para generalizar ante nuevas escenas.

Dado que los modelos como YOLOv8 ya han sido preentrenados en grandes conjuntos de datos generales (como COCO), fue posible aplicar la técnica de *transfer learning*. Esto permitió reutilizar los conocimientos previos del modelo en tareas básicas como la detección de personas, acelerando el proceso de entrenamiento y requiriendo menos datos etiquetados desde cero. Gracias al *transfer learning*, el modelo pudo enfocarse en aprender las características específicas que diferencian una pelea de una interacción común, reduciendo significativamente el tiempo de entrenamiento y mejorando su precisión.

En conjunto, la combinación de fuentes diversas de datos y el uso estratégico del *transfer learning* resultaron fundamentales para lograr una precisión del 70 % en la detección de peleas, una cifra considerable debido a la complejidad visual y contextual de este tipo de eventos.

Figura 1.4 Ejemplo de imagen obtenida desde *dataset* académico de Aktı [2]

Figura 1.5 Ejemplo de imagen obtenida desde *X*

1.7.2 Entrenamiento y ajuste del modelo

El entrenamiento de un modelo de inteligencia artificial, como YOLOv8, es una de las fases más importantes en el desarrollo de sistemas de detección automática. En esta etapa, el modelo ajusta sus parámetros internos a partir de ejemplos etiquetados para poder identificar patrones visuales, como una pelea en este caso.

Durante este proceso, el modelo aprende gradualmente a diferenciar entre lo que constituye una situación de conflicto (como una pelea) y lo que no, basándose en las características visuales extraídas de las imágenes o secuencias de video. Inicialmente, sus predicciones pueden ser equivocadas, marcando objetos incorrectos o ignorando por completo los eventos relevantes. Sin embargo, con cada época de entrenamiento va ajustando su comportamiento para mejorar su precisión.

Para ilustrar esta evolución, se pueden mostrar imágenes con indicadores visuales: un "1" en la esquina para representar cuando el modelo ha identificado correctamente una pelea (acierto), y un "0" cuando ha fallado como se logra apreciar en las figuras 4.1 y 4.2. Estas permiten visualizar claramente la transición del modelo desde un estado inicial de confusión hasta un punto en el que logra reconocer patrones complejos de forma confiable.

1.8. Alcances y Limitaciones

1.8.1 Alcances

El proyecto logró implementar un sistema de detección de peleas en tiempo real basado en el modelo YOLOv8, con una precisión aproximada del 70%. Gracias al uso de técnicas de *transfer learning* y a la recolección cuidadosa de datos, se consiguió entrenar un modelo funcional capaz de reconocer eventos de violencia en escenarios urbanos. Además, demostró ser una herramienta de apoyo que permite reducir la necesidad de supervisión constante de las cámaras.

1.8.2 Limitaciones

Limitaciones en el conjunto de datos:

La recopilación del *dataset* representó uno de los principales desafíos. Muchos de los videos obtenidos de redes sociales (por ejemplo, *X/Twitter*) presentaban problemas de baja calidad, incluyendo resolución limitada, ruido visual significativo y movimientos bruscos de cámara. Estos factores afectaron la calidad de los datos de entrenamiento, dificultando la capacidad del modelo para generalizar correctamente en escenarios diversos.

Detección de eventos sutiles:

Otro desafío importante fue la dificultad inherente a la detección de eventos sutiles. En muchas ocasiones, las peleas no eran plenamente evidentes, ya sea por la rapidez de los movimientos, la parcial visibilidad del evento o gestos ambiguos como empujones breves. Además, las condiciones de baja iluminación o los ángulos desfavorables de cámara redujeron la capacidad del modelo para detectar estos eventos de forma precisa.

Restricciones de hardware:

Las limitaciones de hardware también influyeron en el proyecto. Se trabajó utilizando una GPU NVIDIA GTX 1650 y un procesador Intel(R) Core(TM) i5-10300H, lo cual restringió la capacidad de procesamiento de modelos más pesados y de alta precisión, como YOLO-X. Por esta razón, se optó por implementar una versión más ligera del modelo (YOLOv8n), sacrificando algo de precisión a cambio de un rendimiento aceptable y una mayor eficiencia computacional, necesaria para garantizar la viabilidad de un sistema en tiempo real en equipos de características limitadas.

2. Marco Teórico

2.1. Introducción

La videovigilancia tradicional depende en gran medida de operadores humanos que monitorean cámaras para detectar situaciones de riesgo. Este método, aunque extendido, presenta múltiples desventajas como la supervisión constante de decenas de pantallas que puede generar fatiga, distracción y errores en la interpretación de escenas, especialmente en contextos dinámicos como espacios públicos o campus universitarios.

Con el avance de la inteligencia artificial y la visión por computador, ha sido posible automatizar la detección de objetos mediante algoritmos como YOLO, ampliamente utilizado por su capacidad para reconocer elementos visuales con alta precisión y velocidad, sin embargo, la mayoría de los estudios y aplicaciones previas han centrado el uso de YOLO en la detección de objetos físicos como armas, cuchillos o pistolas sin considerar la complejidad de identificar comportamientos violentos o interacciones humanas ambiguas [16].

Este proyecto plantea una aproximación distinta aplicar, YOLOv8 no para reconocer objetos estáticos, sino para identificar conductas agresivas a partir de patrones visuales en imágenes. Para ello, se entrena un modelo segmentador con un conjunto de datos especialmente etiquetados con escenas de peleas y no peleas, lo que permite desarrollar un sistema de videovigilancia inteligente. Este enfoque transforma la seguridad desde un modelo reactivo y manual hacia uno proactivo y automatizado.

2.2. Conceptos Fundamentales

Visión por computador: Rama de la inteligencia artificial que permite a las máquinas interpretar y procesar imágenes del mundo real. Es la base sobre la cual se construyen los sistemas de detección automatizada en este proyecto.

Detección de objetos: Técnica que identifica y localiza instancias de objetos dentro de una imagen. Se utiliza el modelo YOLO que predice simultáneamente múltiples cuadros delimitadores y clases.

Clases "Pelea" y "No-Pelea": Representan las categorías que el sistema debe reconocer. Cada imagen se etiqueta como "Pelea" o "No-Pelea" lo que guía el aprendizaje supervisado del modelo.

2.3. Teorías y principios relevantes

El funcionamiento del modelo YOLO, en sus distintas versiones, se basa en una arquitectura de detección rápida y eficiente. Uno de los principios más relevantes en su operación es el modelo de grillado o cuadrícula, que divide cada imagen de entrada en S × S celdas iguales. Cada celda es responsable de predecir la presencia de un objeto si su centro se encuentra dentro de dicha celda.

Cada una de estas celdas genera múltiples predicciones, las cuales incluyen:

1. Las coordenadas de las bounding boxes (cajas delimitadoras).
2. Un valor de confianza, que representa la probabilidad de que exista un objeto en la caja y la precisión con la que está localizada.
3. Las probabilidades de clase, indicando a qué categoría pertenece el objeto, en este caso, "Pelea" o "No pelea".

La red neuronal evalúa toda la imagen en una captura, lo que permite detectar múltiples objetos o eventos simultáneamente y en tiempo real. Este enfoque integral contrasta con modelos más tradicionales como R-CNN, que realizan detección por etapas.

Una parte clave del funcionamiento es la predicción unificada: si múltiples celdas detectan una misma acción o región de interés, el sistema utiliza algoritmos como Non-Maximum Suppression (NMS) para consolidar las predicciones redundantes y mantener sólo las más relevantes, mejorando la precisión final del modelo.

Además, durante el entrenamiento, la red aprende a asociar patrones visuales específicos con las clases de interés, ajustando los pesos de sus capas internas mediante el algoritmo de backpropagation. Esto permite que YOLO no solo detecte la presencia de una pelea, sino también que ignore comportamientos irrelevantes o ambiguos. A diferencia de la detección de objetos estáticos como armas, la identificación de una pelea requiere captar dinámicas visuales: posturas agresivas, contacto físico, agrupamiento de personas, entre otros.

En resumen, el modelo YOLO transforma imágenes en predicciones estructuradas mediante un sistema de cuadrículas, utilizando principios de redes convolucionales, regresión directa y optimización por retro propagación, lo cual permite una detección veloz y suficientemente precisa para tareas en entornos reales, como sistemas de videovigilancia inteligentes.

2.4. Justificación del enfoque

El enfoque adoptado en este proyecto se justifica por la necesidad de automatizar procesos de vigilancia en entornos donde los recursos humanos son limitados y la atención constante a múltiples cámaras resulta poco eficiente. A diferencia de los sistemas tradicionales, que requieren supervisión manual continua, este proyecto se apoya en modelos de detección en tiempo real para identificar eventos potencialmente violentos.

La elección de la arquitectura YOLOv8-seg responde a su capacidad de equilibrio entre velocidad y precisión, lo cual es crucial en aplicaciones de vigilancia. Su naturaleza de segmentación permite obtener una mayor precisión espacial sobre los objetos detectados, lo que facilita el análisis del comportamiento en la escena. Esta característica se alinea con la meta de no solo reconocer la presencia de personas, sino también evaluar su interacción.

Además, se incorporaron métricas espaciales simples, como la distancia entre centroides, y el conteo de personas, para inferir comportamientos sin requerir modelos complejos de reconocimiento de acciones. Esta decisión técnica se fundamenta en la necesidad de operar con recursos computacionales limitados, utilizando hardware accesible (GPU GTX 1650), sin comprometer la viabilidad del sistema en entornos reales.

Por tanto, el enfoque adoptado equilibra la eficiencia computacional, la precisión en la detección, y la capacidad de generalización, permitiendo que el sistema pueda ser integrado en múltiples contextos como universidades, condominios o espacios públicos, adaptándose a distintas configuraciones de cámara y condiciones ambientales.

3. Resultados experimentales

3.1. Etiquetado manual en Roboflow

El etiquetado es el proceso de asignar etiquetas o anotaciones a datos (como imágenes o videos) para identificar objetos, acciones o características relevantes. En el contexto del aprendizaje automático, es la base para entrenar modelos de inteligencia artificial, ya que los algoritmos aprenden a reconocer patrones a partir de ejemplos etiquetados. Sin un etiquetado preciso y consistente, los modelos carecen de la información necesaria para generalizar y realizar predicciones confiables.

El desarrollo del modelo presentado se sustenta en un conjunto de datos compuesto por más de 3.000 imágenes etiquetadas manualmente. Este proceso fue realizado a lo largo de un período aproximado de seis meses, utilizando herramientas de etiquetado semiautomático proporcionadas por Roboflow, lo que permitió acelerar parcialmente la tarea sin comprometer la calidad de las anotaciones. Cada imagen fue revisada y segmentada para distinguir entre escenas con presencia de peleas y aquellas que no, permitiendo al modelo aprender patrones visuales complejos y contextualmente relevantes. Esta labor fue fundamental para alcanzar los niveles de confiabilidad en la detección en tiempo real, ya que un dataset bien estructurado y diverso es clave para lograr un desempeño robusto en condiciones variables.

3.2. Plataforma Roboflow: Etiquetado asistido por IA

Roboflow es una herramienta para la gestión de *datasets* en visión con inteligencia artificial. Entre sus funciones clave destacan:

3.2.1 Etiquetado semiautomático:

- **Utiliza modelos preentrenados:** Utiliza por ejemplo YOLO para sugerir *bounding boxes* o clasificaciones automáticamente.
- **Segmentación de regiones de interés**: Delimitación de *bounding boxes* alrededor de personas permitiendo quedar a criterio del autor las personas implicadas en comportamientos agresivos y también las que no, ya que, YOLO funciona mejor como detector binario donde mejora la precisión si ponemos un evento como cuando ocurre y cuando no es así.

3.2.2 Flujo de trabajo optimizado:

- **Colaboración**: Permite equipos trabajar simultáneamente en el mismo proyecto.
- **Validación**: Herramientas para revisar y corregir etiquetas inconsistentes.
- **Integración**: Exporta *datasets* en formatos compatibles con TensorFlow, PyTorch, etc.

3.2.3 Data augmentation:

- **Transformaciones a las imágenes**: estas son por ejemplo (rotación, cambio de brillo, recorte, efecto espejo) para aumentar la diversidad del dataset

Clasificación de eventos: Etiquetas como "pelea" y "no-pelea".

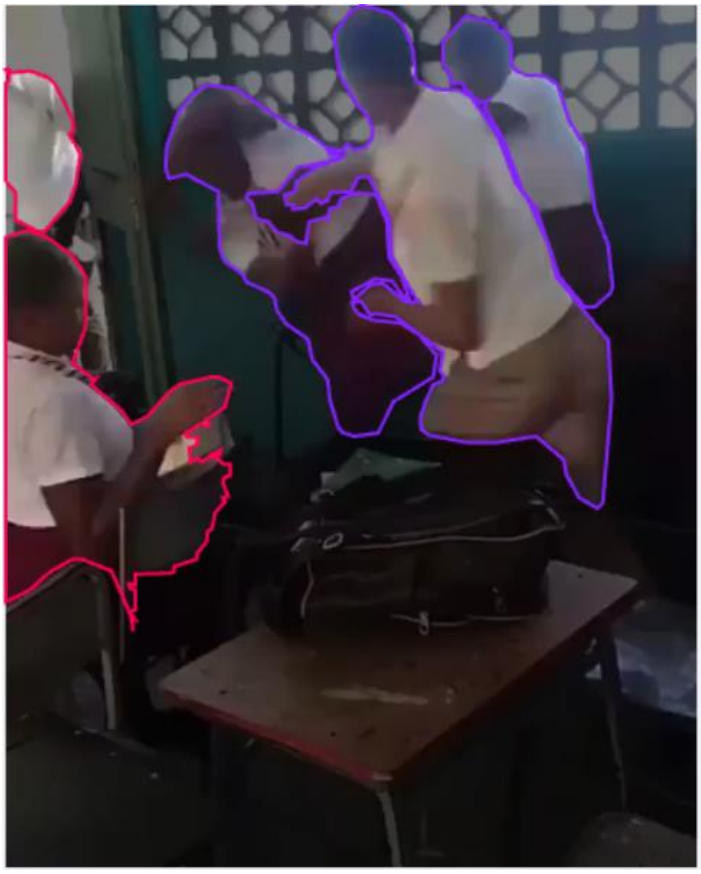

Figura 3.1 Ejemplo de etiquetado manual en Roboflow. Las regiones moradas consisten en personas ‘peleando’, mientras que las rojas corresponden a los no involucrados.

Figura 3.2 Segundo ejemplo de etiquetado manual en Roboflow.

Figura 3.3 Tercer ejemplo de etiquetado manual en Roboflow.

3.3. Diagrama de flujo de detección

Un diagrama de flujo es una representación gráfica que describe de forma secuencial y lógica los pasos o decisiones dentro de un proceso. En el contexto del presente trabajo, el diagrama de flujo permite visualizar las etapas principales del sistema de detección de peleas, desde la captura de la imagen hasta la clasificación final de la situación observada. Cada bloque representa una acción o una decisión clave, facilitando la comprensión del funcionamiento del modelo y el razonamiento detrás de su toma de decisiones.

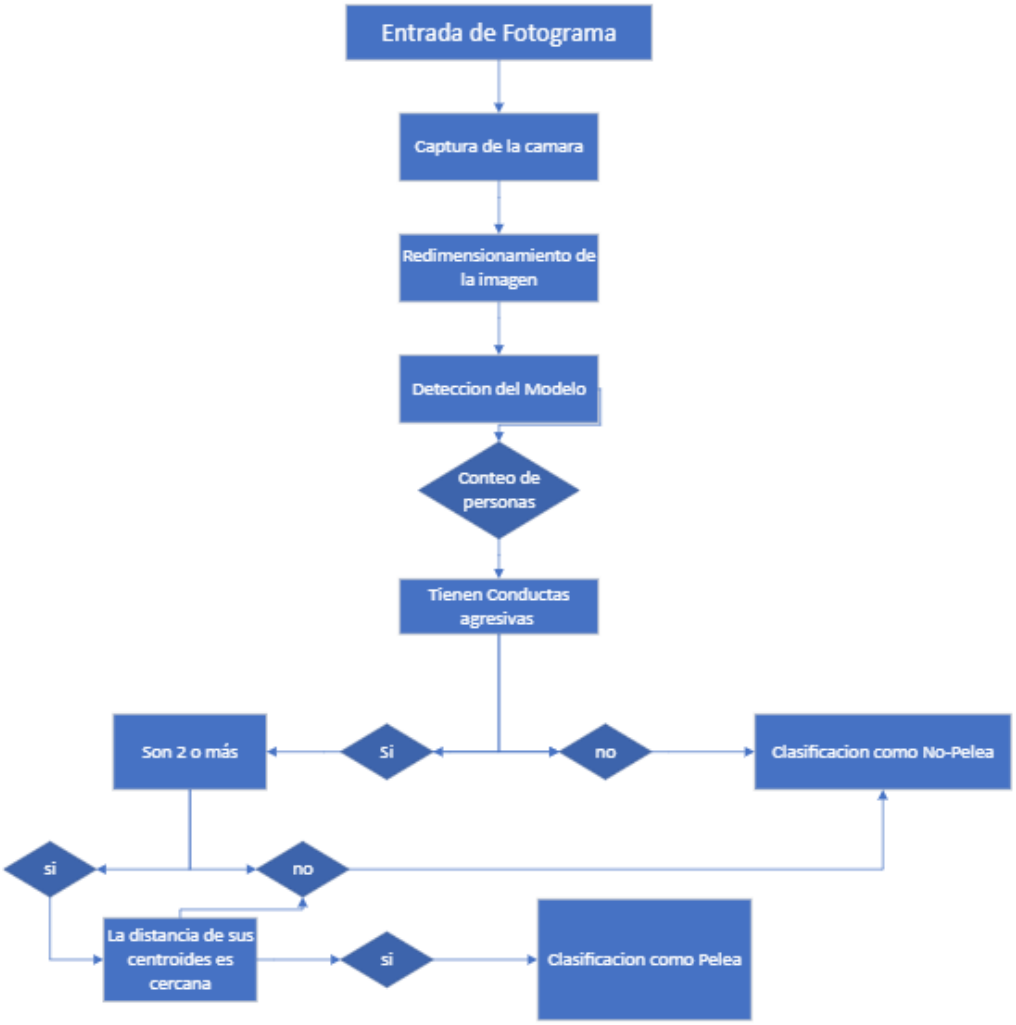

Figura 3.4 Diagrama de flujo de detección del modelo

El diagrama de flujo representa las etapas clave del sistema propuesto para la detección de peleas en tiempo real. Este proceso comienza con la entrada de un fotograma, que puede provenir de una cámara de vigilancia. A continuación, se realiza la captura de imagen desde la cámara, la cual es redimensionada para ajustarse a los requerimientos del modelo YOLO, optimizando así la detección.

Luego, la imagen redimensionada es procesada por el modelo de detección, que se encarga de identificar objetos de interés, en este caso personas. Tras ello, se ejecuta un conteo de personas dentro del encuadre, lo que permite determinar si hay más de un individuo presente, requisito mínimo para que una situación sea potencialmente considerada una pelea.

Posteriormente, el sistema analiza si existen conductas agresivas a partir de las características aprendidas durante el entrenamiento del modelo, por ejemplo, un empujón, empuñar la mano algún movimiento brusco entre otros comportamientos no comunes dentro de un transeúnte. Si no se detectan tales conductas, la escena se clasifica directamente como "No-Pelea". En cambio, si se identifica un comportamiento agresivo, se continúa con un filtro adicional que evalúa si las personas involucradas son dos o más.

En caso de cumplirse esta condición, se verifica si la distancia entre los centroides (puntos medios de los cuerpos detectados) es suficientemente cercana, lo que es indicativo de una interacción física posible. Si esta distancia es adecuada, se realiza la clasificación como "Pelea" y ,en cambio, si no se cumple con este criterio, se descarta como pelea y se clasifica la escena como "No-Pelea".

Este enfoque secuencial y condicional permite reducir falsos positivos al incorporar tanto la cantidad de personas como su proximidad y tipo de conducta, mejorando la precisión y robustez del sistema en escenarios del mundo real.

3.4. Gráficos resultantes de entrenamiento train/val/metrics

Estos gráficos muestran el proceso de entrenamiento y validación de un modelo YOLOv8, utilizado para la detección y segmentación de objetos en tiempo real. Se divide en dos tipos de métricas principales: las pérdidas (losses), que indican el error porcentual tiene el modelo en cada etapa del aprendizaje, y las métricas de desempeño, que reflejan qué tan bien está detectando los objetos esperados en este caso ‘peleas’.

Las pérdidas incluyen errores en la predicción de cajas delimitadoras (box loss), segmentación (seg loss), clasificación (cls loss) y refinamiento de bordes (dfl loss). Estas deben disminuir con el entrenamiento, lo que indica que el modelo está aprendiendo a hacer mejores predicciones.

Por otro lado, las métricas de desempeño incluyen precisión (precision), que mide cuántas detecciones fueron correctas, y recall, que evalúa cuántos objetos relevantes fueron detectados. Además, se muestra el mAP (mean average precision), un indicador clave de la precisión general del modelo.

Las figuras muestran la evolución de distintas funciones de pérdida durante el proceso de entrenamiento del modelo para la tarea de segmentación, clasificación y detección de peleas. El eje X representa el número de épocas (iteraciones completas sobre el conjunto de datos de entrenamiento), mientras que el eje Y indica el valor de pérdida (loss) correspondiente a cada métrica.

Box_loss está relacionada con la precisión en la predicción de las cajas delimitadoras que encierran a los objetos (en este caso, personas peleando). Se observa una disminución constante, lo que indica que el modelo aprende progresivamente a ubicar correctamente a los sujetos en conflicto.

Seg_loss Corresponde a la pérdida de segmentación, es decir, la calidad con la que el modelo delimita los contornos precisos de los objetos dentro de las cajas. El descenso notable sugiere que el modelo mejora su capacidad para diferenciar claramente las figuras de interés.

Cl_loss representa la pérdida de clasificación, es decir, que tan acertadamente el modelo asigna a cada detección una clase correcta (ya sea "Pelea" o "No Pelea"). Una caída rápida al inicio seguida de una estabilización indica una mejora sólida en la diferenciación entre clases.

Dfl_loss (Distribution Focal Loss) Esta pérdida se usa para mejorar la precisión en la predicción de la ubicación de los bordes de los objetos dentro de la caja delimitadora. Su reducción también valida que el modelo ajusta de forma precisa los bordes durante el entrenamiento.

Estas figuras representan el comportamiento de la pérdida del modelo durante la validación, es decir, cuando se evalúa el rendimiento del modelo sobre datos que no ha visto previamente. Esto es esencial para comprobar que el aprendizaje es generalizable y no se limita al conjunto de imágenes de entrenamiento.

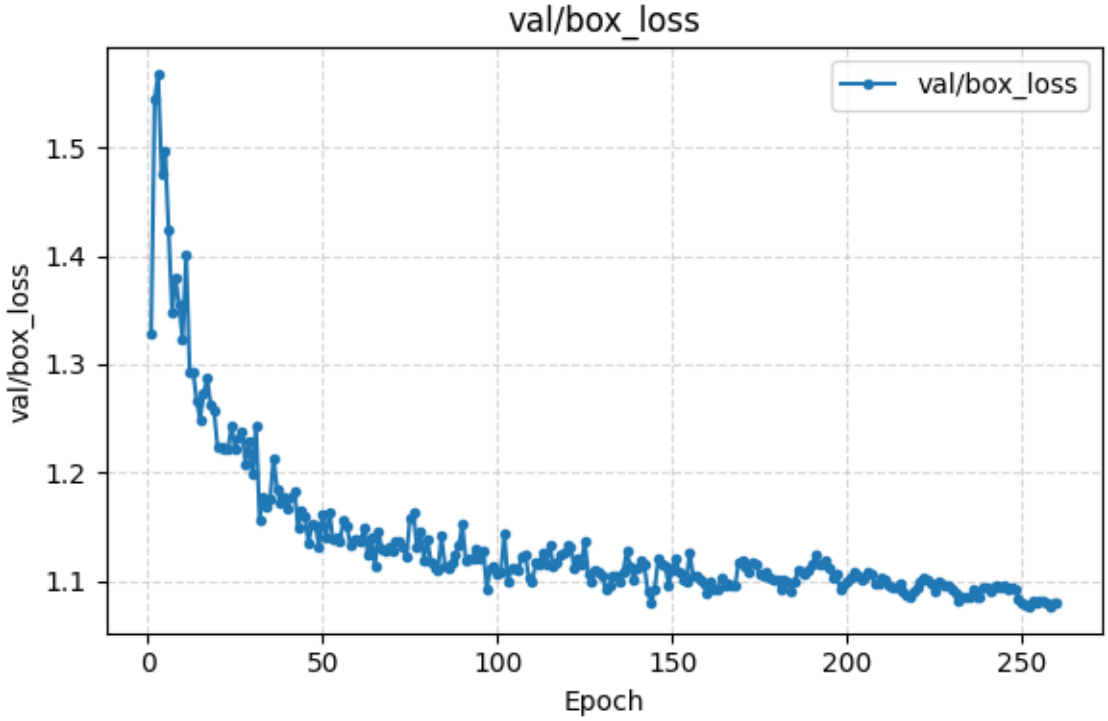

Figura 3.5 Gráfico validación de perdida de *bounding box*

Box_loss Mide la capacidad del modelo para detectar la ubicación de las personas en conflicto mediante cajas delimitadoras. Se observa una disminución constante en las primeras 100 épocas, seguida de una leve estabilización, lo que indica un desempeño consistente.

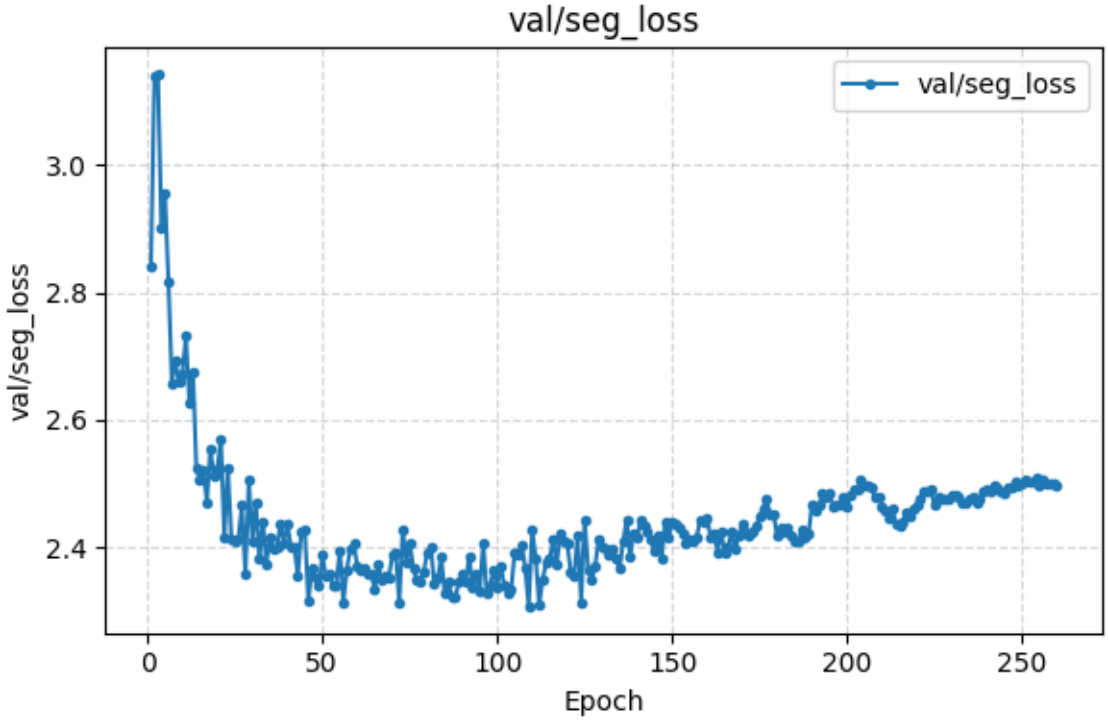

Figura 3.6 Gráfico validación de perdida de segmentación

Seg_loss Evalúa la precisión con la que el modelo realiza la segmentación de los sujetos en la imagen. Se observa una curva en forma de "U", con un mínimo alcanzado alrededor de la época 100, tras la cual comienza un ligero incremento. Este comportamiento puede ser indicio de inicio de sobreajuste, donde el modelo empieza a ajustarse demasiado al conjunto de entrenamiento.

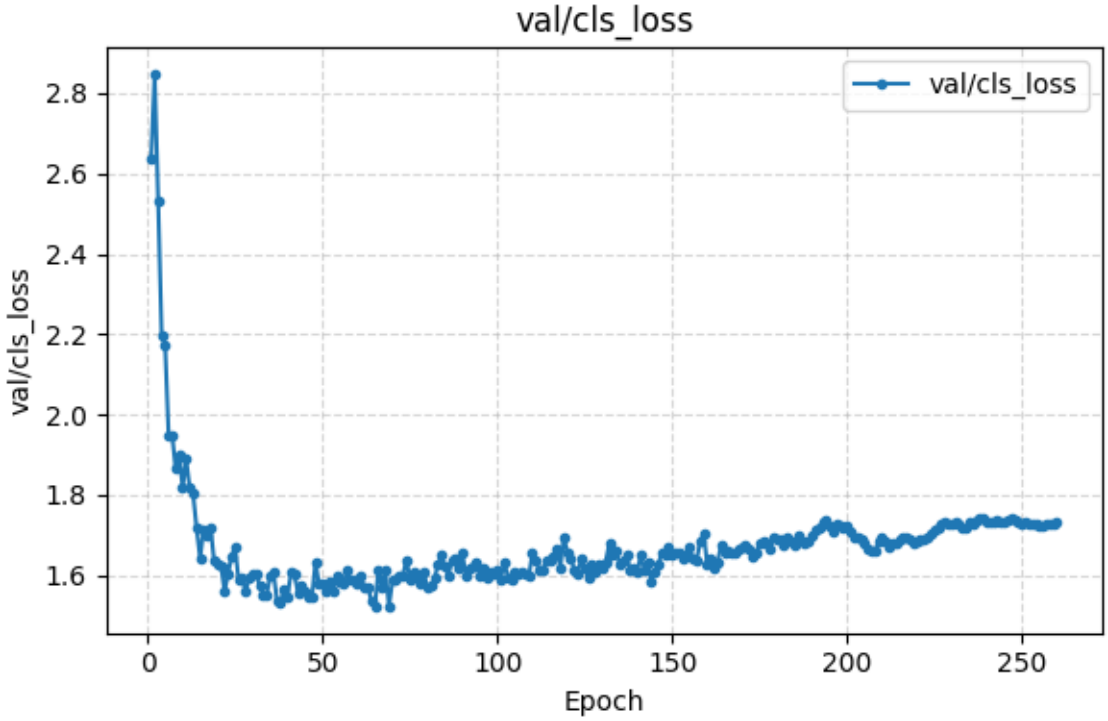

Figura 3.7 Gráfico validación de perdida de clasificación

Cls_loss Corresponde a la pérdida de clasificación de las clases "Pelea" y "No Pelea". El patrón es similar al anterior: una caída inicial y posterior aumento leve, lo que también sugiere una posible saturación en la capacidad de generalización.

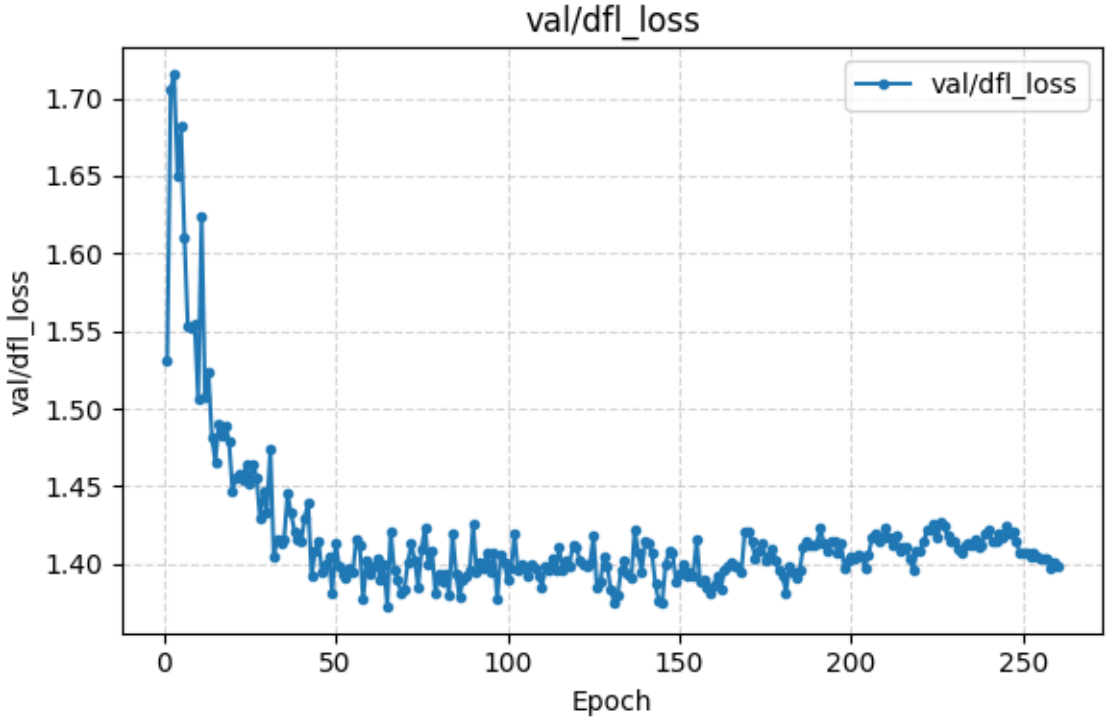

Figura 3.8 Gráfico validación de perdida de posición de bordes *bounding box*

Dfi_loss Evalúa la precisión en la predicción fina de la posición de los bordes de las cajas. Tiene un comportamiento muy parecido al de cls_loss, lo cual refuerza la hipótesis de sobreajuste leve a partir de cierta cantidad de épocas.

Los gráficos muestran cómo evoluciona algunas métricas a lo largo del tiempo, diferenciando entre dos tipos de predicciones:

(B): Bounding Box: desempeño en detección de cajas.

(M): Mask: desempeño en segmentación.

3.4.1 Metrics/precision(B) y metrics/precision(M)

Miden qué tan precisas son las predicciones del modelo: es decir, cuántas de las predicciones positivas realmente son verdaderas. Ambas métricas muestran una tendencia ascendente y estable hacia valores cercanos a 0.65 – 0.7, lo que indica un buen nivel de precisión tanto en detección como en segmentación.

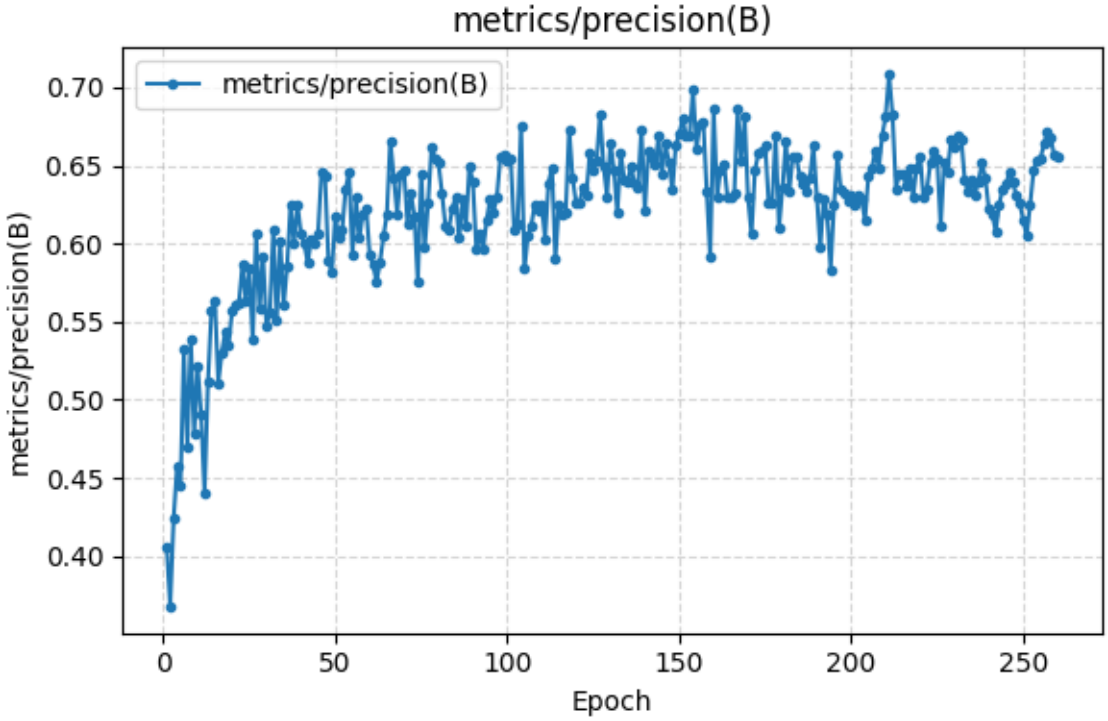

Figura 3.9 Gráfico precision métrica *bounding box*

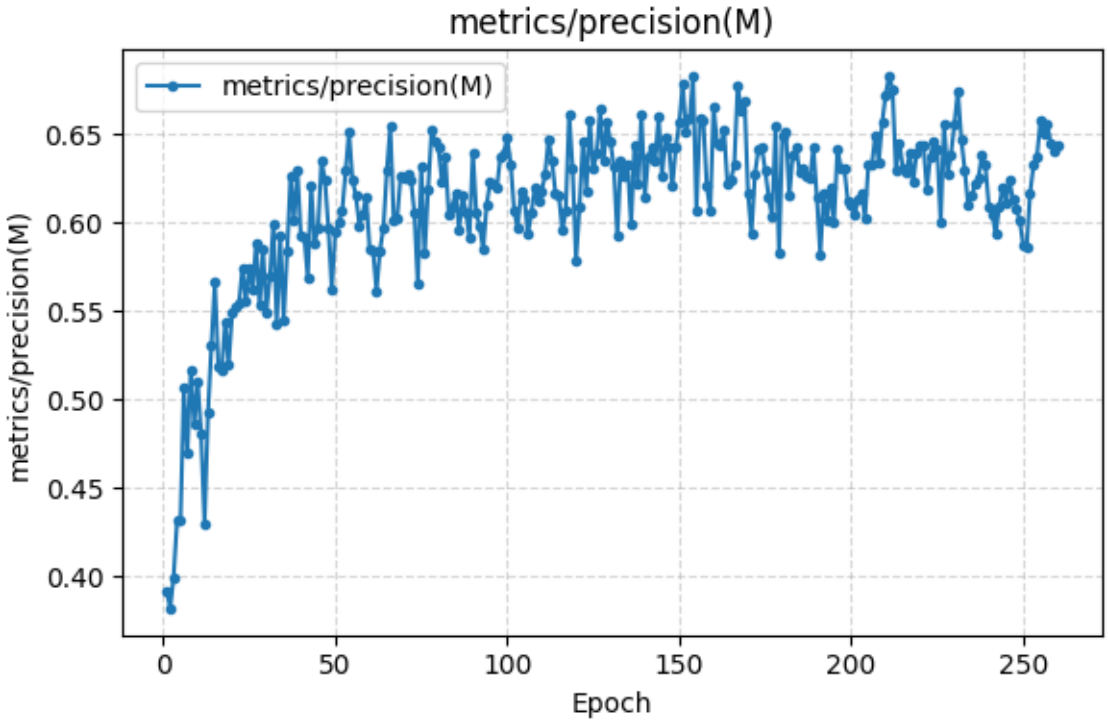

Figura 3.10 Gráfico precision métrica segmentación

3.4.2 Metrics/recall(B) y metrics/recall(M)

Miden qué tantas de las verdaderas clases fueron correctamente identificadas. Aunque hay algo de variabilidad, se estabilizan alrededor de 0.5 – 0.55, lo que indica que el modelo tiene un desempeño equilibrado.

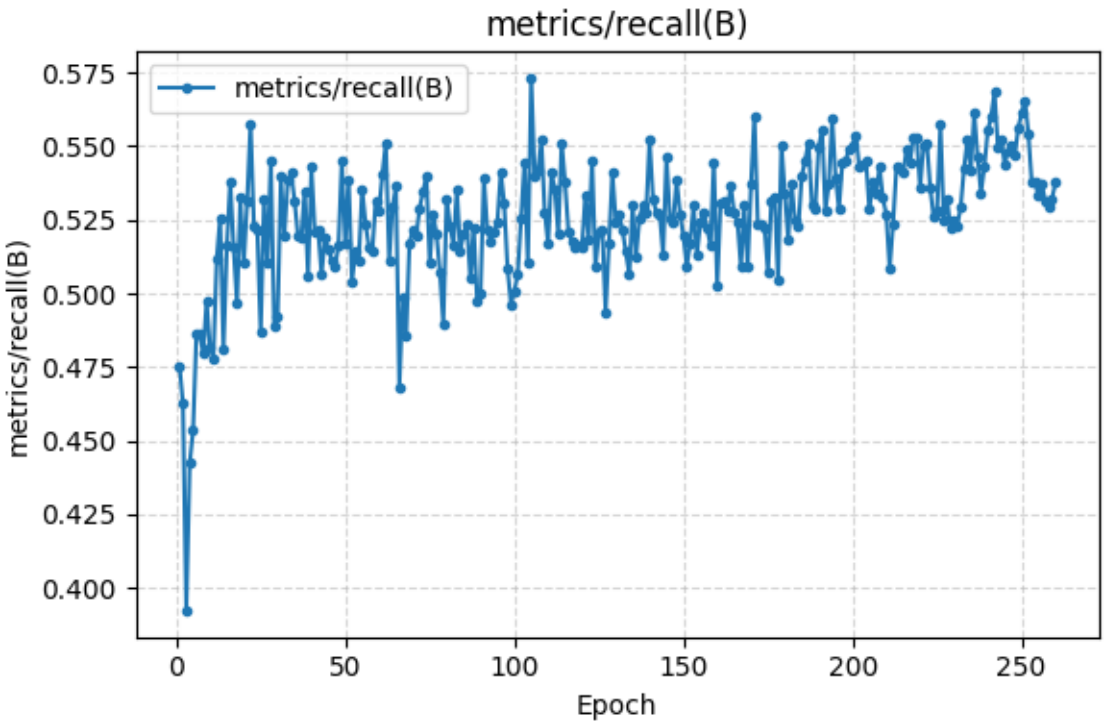

Figura 3.11 Gráfico métrica recall *bounding box*

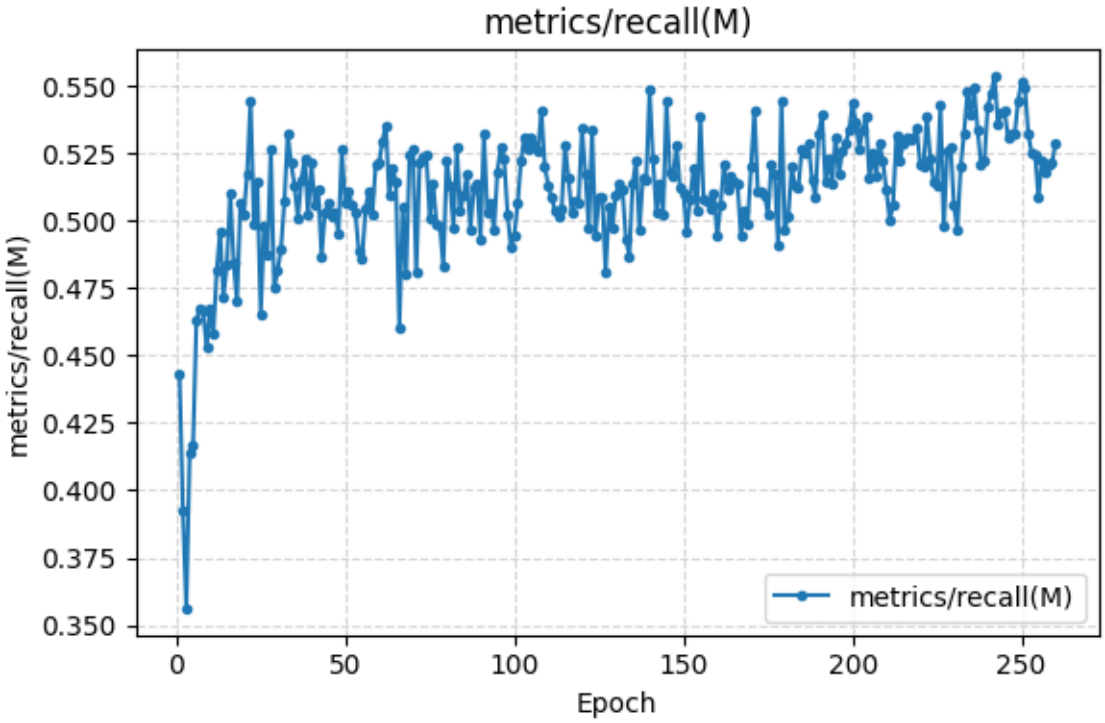

Figura 3.12 Gráfico métrica recall segmentación

3.4.3 Metrics/mAP50(B) y metrics/mAP50(M)

El mAP0.5 indica qué tan bien el modelo localiza y clasifica objetos cuando el umbral de coincidencia es del 50%. Ambas curvas muestran una mejora rápida al inicio y luego se estabilizan en torno a 0.50 – 0.55, lo cual es un resultado bastante sólido.

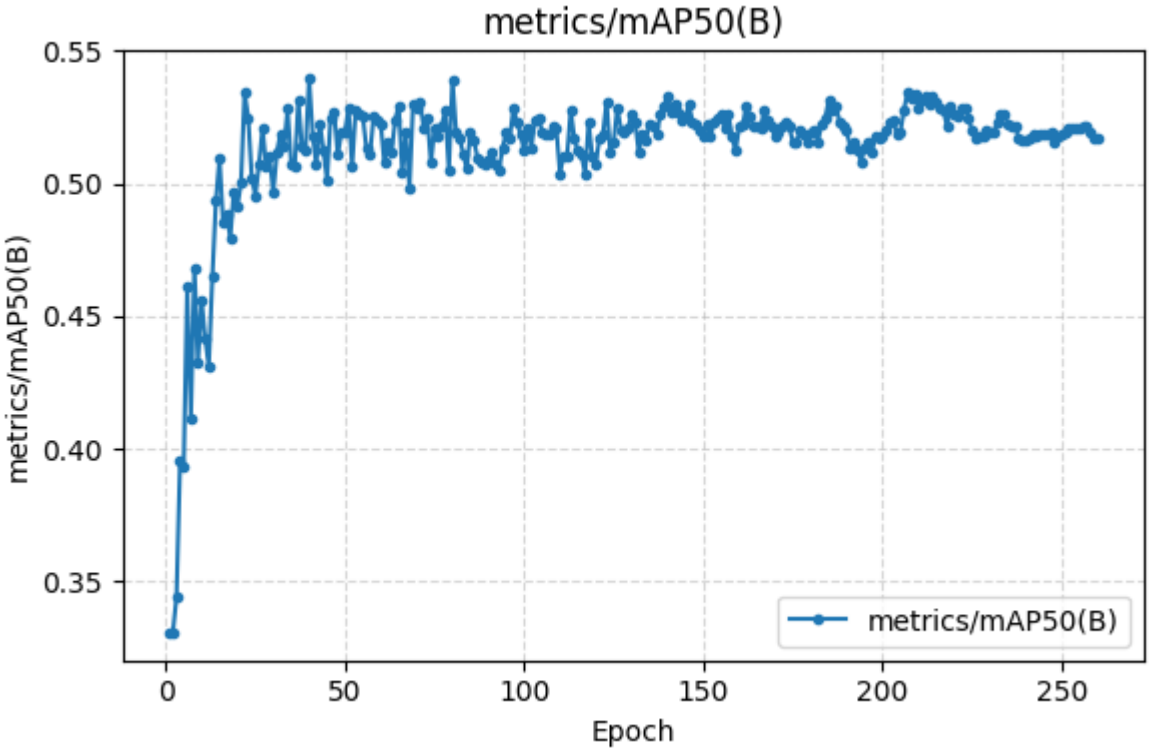

Figura 3.13 Gráfico métrica precision mAP0.5 *bounding box*

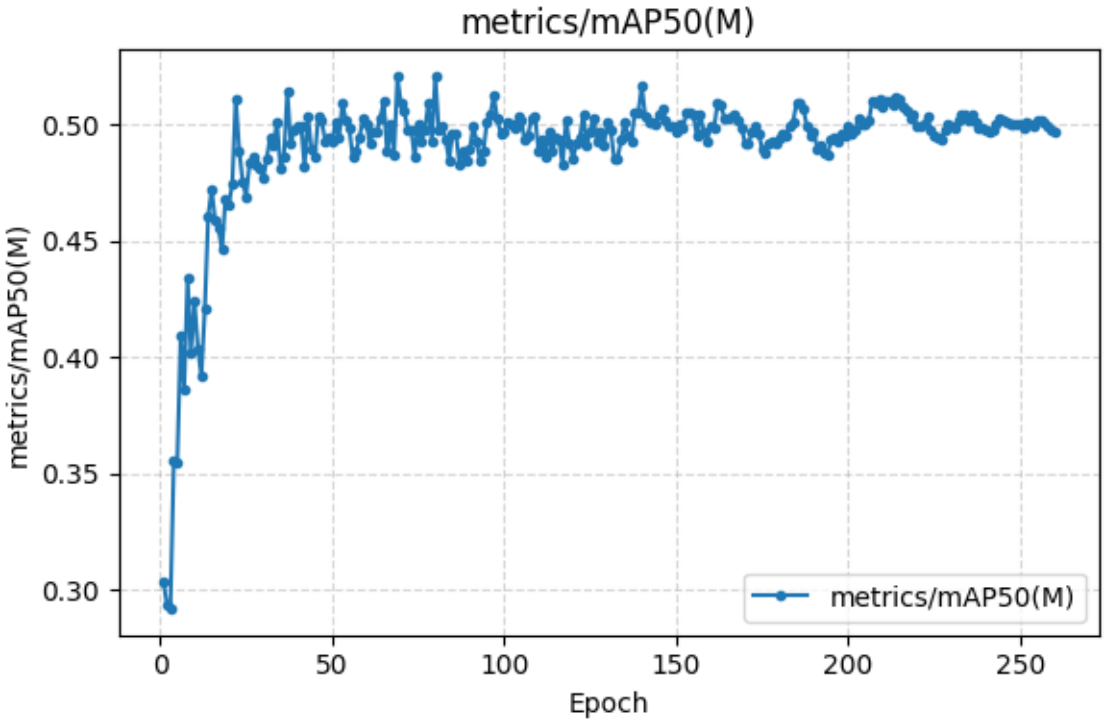

Figura 3.14 Gráfico métrica precision mAP0.5 segmentación

3.4.4 Metrics/mAP50-95(B) y metrics/mAP50-95(M)

Esta es una métrica más exigente, ya que promedia la precisión en varios umbrales (de 0.5 a 0.95 en pasos de 0.05). Aunque empieza más baja (0.15–0.20), sube progresivamente y se estabiliza cerca de 0.35–0.40 para *Bounding Boxes* y 0.30 para máscaras, lo cual es esperable y evidencia un modelo que generaliza bien en distintas condiciones.

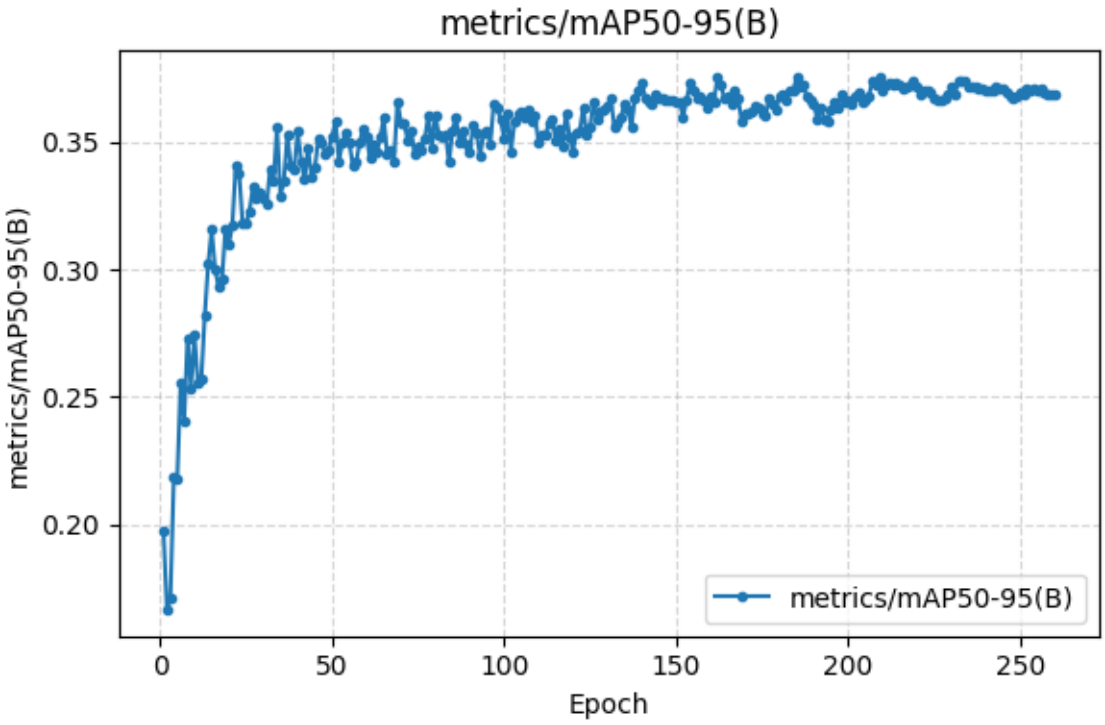

Figura 3.15 Gráfico métrica mAP50-95 *bounding box*

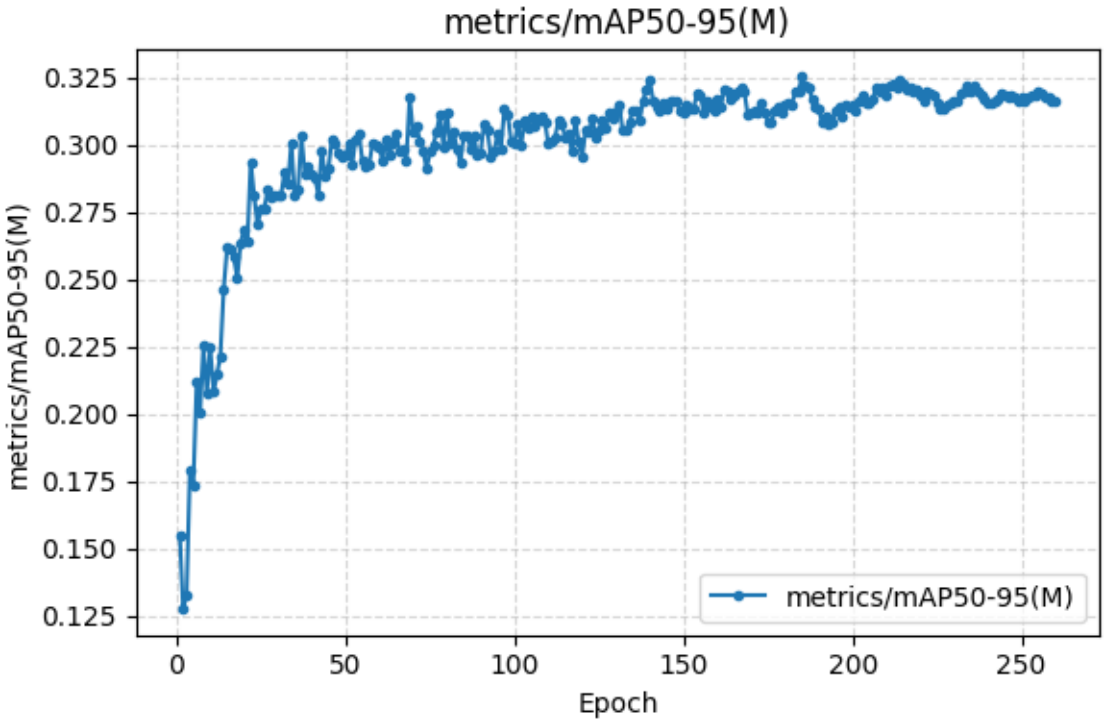

Figura 3.16 Gráfico métrica mAP50-95 segmentación

Los gráficos muestran una reducción progresiva en las pérdidas, lo que indica que el modelo está aprendiendo adecuadamente, sin embargo, en la validación, algunas pérdidas presentan discontinuidad, lo que podría sugerir sobreajuste o desafíos en la generalización a nuevos datos.

Las métricas de precisión y mAP mejoran con el tiempo, lo que confirma que el modelo está aprendiendo a detectar objetos con mayor exactitud. Sin embargo, las variaciones en el recall y la precisión indican que aún hay margen de mejora, posiblemente optimizando hiperparámetros o utilizando más datos diversos para el entrenamiento.

3.5. Gráficos resultantes de entrenamiento Precision-Confidence

El presente gráfico representa la curva de precisión vs. confianza para el modelo de detección de peleas. En el eje X se muestra la confianza de la predicción (qué tan seguro está el modelo de su decisión), mientras que en el eje Y se encuentra la precisión (qué porcentaje de las detecciones fueron correctas).

Las líneas representan diferentes clases:

- Naranja (Pelea): Muestra el desempeño del modelo al detectar peleas.
- Celeste (No-Pelea): Indica la precisión para identificar personas no implicadas en la 'Pelea'.
- Azul (Promedio de todas las clases): Refleja el comportamiento global del modelo.

En general, a medida que aumenta la confianza, la precisión también mejora, lo que indica que el modelo es más confiable cuando hace predicciones con alta seguridad. Sin embargo, en los valores cercanos a 1, hay alteración, lo que sugiere posibles falsos positivos o problemas de sobreajuste.

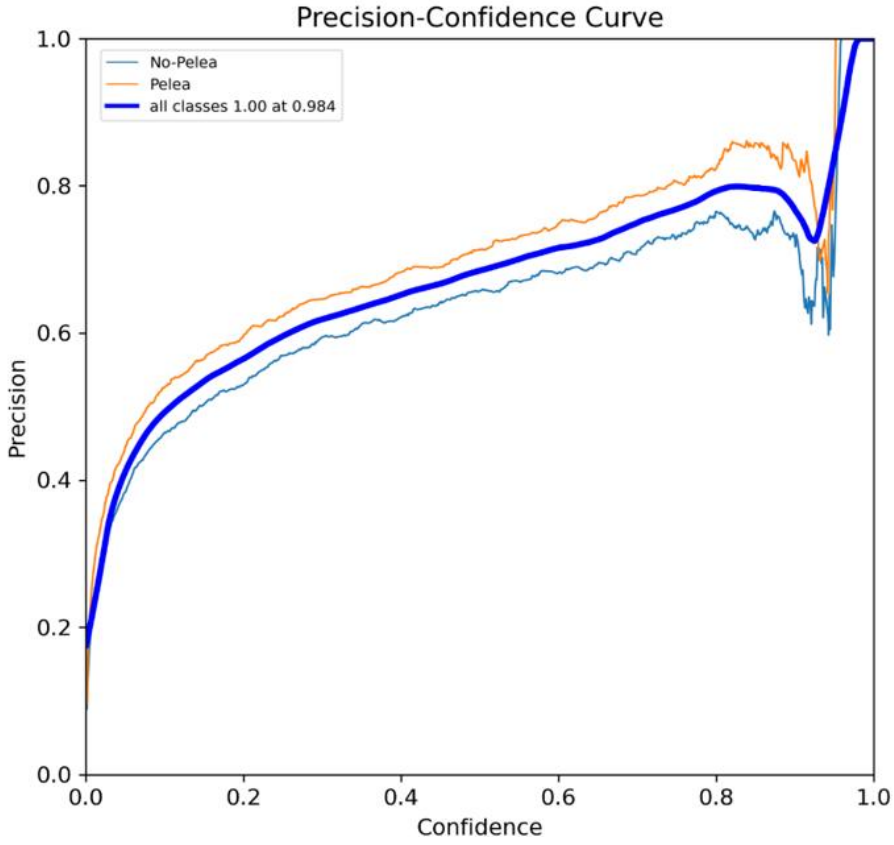

Figura 3.17 Gráfico Precision-Confidence después de entrenamiento del modelo

El gráfico muestra que la precisión del modelo mejora a medida que aumenta la confianza en las predicciones. La clase "Pelea" (línea naranja) tiene una precisión más alta que la clase "No-Pelea" (línea azul claro), lo que indica que el modelo es más efectivo identificando peleas que descartándolas.

3.6. Mejoras y estrategias del modelo dentro del proyecto

Para mejorar la precisión del modelo en la detección de peleas, se implementaron diversas estrategias relacionadas tanto con el entrenamiento como con la lógica de detección. Una de ellas fue el cálculo de la distancia entre centroides, definidos como los puntos centrales de los *bounding boxes* que rodean a las personas detectadas. Estos centroides se obtienen promediando las coordenadas de las esquinas superior izquierda (X_1, Y_1) e inferior derecha (X_2, Y_2), siguiendo la fórmula X = $(X_1 + X_2)/2$ Y= $(Y_1 + Y_2)/2$. Cuando la distancia entre dos centroides es menor a un umbral (10 píxeles en este caso), se interpreta como una posible interacción cercana, útil para identificar conflictos físicos. Este umbral, sin embargo, no es generalizable, ya que depende de variables como la posición de la cámara o la escala de la escena. Además, se incorporó un conteo de personas por imagen en que el sistema solo evalúa la clase "pelea" si hay dos o más personas en el fotograma, reduciendo así los falsos positivos en escenas con individuos aislados. Durante el entrenamiento del modelo, se emplearon 300 épocas como límite, aunque el proceso se detuvo automáticamente en la época 270 gracias a la técnica de *Early Stopping*, que finaliza el entrenamiento cuando no se detectan mejoras en la pérdida de validación tras 20 iteraciones consecutivas. Este mecanismo, integrado por defecto en Ultralytics, permite evitar el sobreajuste y asegura un uso eficiente de los recursos computacionales durante el ajuste de parámetros.

4. Resultados de Detección

4.1. Valores de entrenamiento

En la figura se observa un caso de entrenamiento fallido, evidenciado por la etiqueta "0" en la esquina de la imagen. Este valor indica que el modelo no fue capaz de detectar correctamente una pelea cuando está realmente ocurría o, en otros casos, que tampoco logró entender adecuadamente que no había una pelea presente. Esta representación resulta útil para evidenciar cómo, en las primeras etapas del entrenamiento o frente a imágenes ambiguas, el sistema puede fallar tanto en identificar peleas o en la no existencia de esta.

Figura 4.1 Ejemplo de entrenamiento mayoritariamente fallido

En esta figura se presenta un ejemplo de entrenamiento mayoritariamente exitoso. La etiqueta "1", ubicada en la esquina de la imagen, indica que el modelo ha logrado identificar correctamente las clases de pelea y no-pelea. Esto evidencia que el modelo ha aprendido a distinguir patrones visuales relevantes, tanto para comportamientos agresivos como para escenas normales, validando su capacidad de interpretar correctamente las clases durante el proceso de entrenamiento.

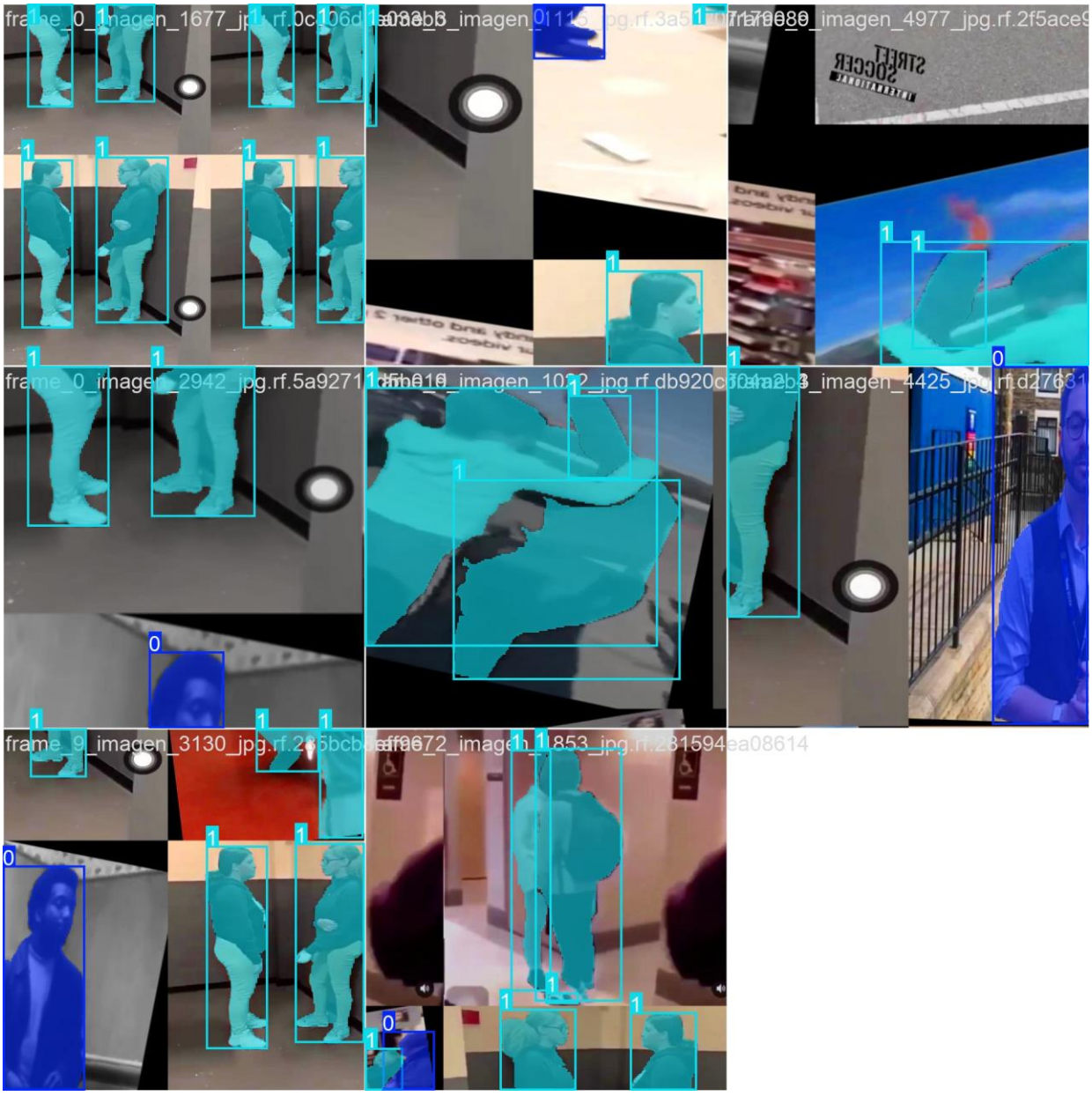

Figura 4.2 Ejemplo de entrenamiento mayoritariamente acertado

4.2. Detección de Imagen obtenida desde X

Se evaluó el modelo con contenido de grabaciones públicas de peleas obtenido desde **X (Twitter)**. Un ejemplo destacado de detección de una pelea callejera:

Figura 4.3 Ejemplo de detección de pelea callejera

En la figura 4.3 se observa una escena urbana donde el modelo ha segmentado y clasificado correctamente a múltiples individuos. Las figuras coloreadas en morado corresponden a situaciones que el sistema ha etiquetado como “Pelea”, mientras que el color rojo identifica la clase “No-Pelea”. Cada etiqueta se acompaña de un porcentaje que representa el nivel de confianza del modelo respecto a su predicción. Por ejemplo, valores como 96% o 97% indican una alta certeza en la identificación de las clases.

4.3. Prueba de campo

Para evaluar el rendimiento del modelo en condiciones realistas, se realizó una prueba controlada en las dependencias de la universidad específicamente en el DTI. El objetivo fue simular una pelea entre compañeros, replicando escenarios de violencia para validar la capacidad del sistema.

Figura 4.4 Ejemplo de detección en universidad con sus respectivos porcentajes de confiabilidad

En la Figura 4.4 se presenta una escena simulada donde el modelo ha segmentado y clasificado correctamente a distintos individuos, en este caso compañeros de la universidad. Las *bounding boxes* indican las clases asignadas por el sistema: “Pelea” y “No-Pelea”. Cada predicción incluye un porcentaje que refleja el nivel de confianza del modelo respecto a su clasificación. Se puede observar que las personas involucradas en el evento de agresión han sido etiquetadas como “Pelea” con valores de confianza superiores al 80%, mientras que los individuos no participantes fueron correctamente identificados como “No-Pelea”, con niveles de confianza similares. Esto sugiere que el modelo logra discriminar de manera efectiva entre comportamientos agresivos y no agresivos en escenarios controlados.

4.4. Prueba de campo en tiempo real

Como resultado práctico, se realizó una prueba de detección en tiempo real ante una situación simulada de pelea. En la Figura 4.5 se muestra el cuadro original capturado por la cámara de videovigilancia ubicada en las dependencias de la universidad específicamente en el DTI, en la Figura 5.4 el resultado tras el procesamiento con el modelo YOLOv8, donde se identifican y segmentan los implicados en el conflicto.

Figura 4.5 Imagen en tiempo real guardada en sistema automáticamente

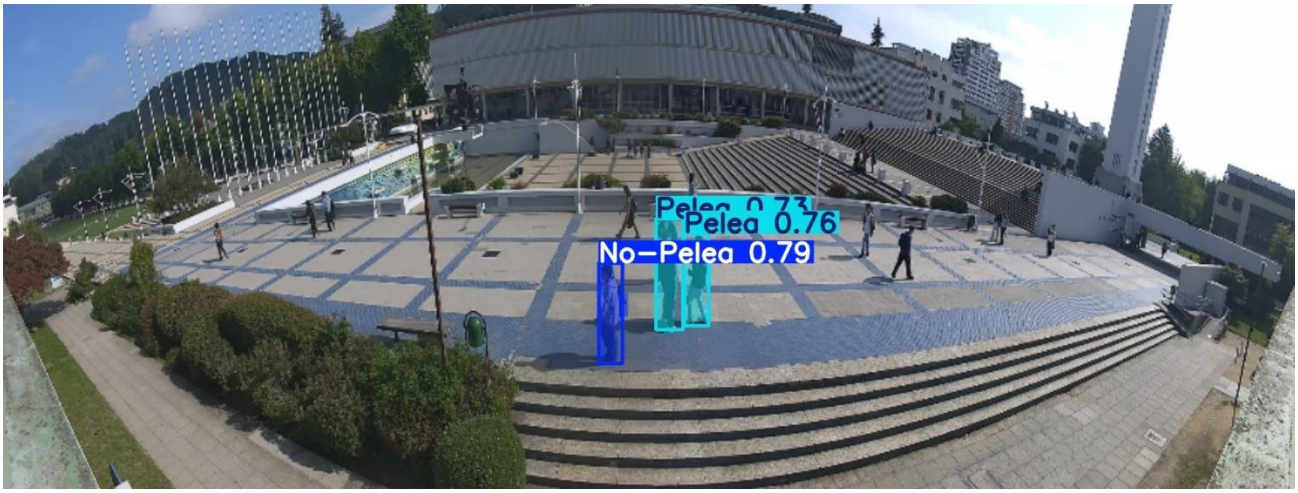

Figura 4.6 Imagen procesada después de la detección

Desde las dependencias universitarias, específicamente en el DTI, se aprecia que el sistema logra detectar interacciones clasificadas como “Pelea” con niveles de confianza superiores al 70%, demostrando su capacidad para operar en tiempo real. En paralelo, se observa una clasificación precisa como “No-Pelea” en el caso de un estudiante no involucrado, lo que evidencia que el modelo es capaz de discriminar adecuadamente entre comportamientos conflictivos y no conflictivos. No obstante, también se identifican limitaciones relacionadas con la generalización donde el modelo reduce su precisión al analizar individuos más alejados o parcialmente visibles, lo que sugiere que aún presenta dificultades para interpretar escenas con condiciones visuales complejas o desfavorables.

5. Conclusiones

5.1. Sumario

El proyecto abordó el desarrollo de un sistema de detección de peleas en tiempo real utilizando el algoritmo YOLOv8 y cámaras de videovigilancia. Se recopiló y procesó un conjunto de datos combinando imágenes extraídas de redes sociales, un *dataset* académico (Aktı et al., 2022) utilizando la segmentación automatizada de Roboflow. A través del entrenamiento y evaluación del modelo, se logró obtener una precisión significativa en la clasificación de escenas de pelea y no pelea, validando su funcionamiento en entornos controlados.

5.1.1 Conclusiones

El trabajo ha demostrado la viabilidad técnica y práctica de implementar un sistema de detección de peleas en tiempo real utilizando el modelo YOLOv8, alcanzando una precisión aproximada del 70% en escenarios simulados y registros de videovigilancia reales. Esta investigación se enmarca en un contexto de creciente necesidad por reforzar la seguridad ciudadana a través de herramientas tecnológicas inteligentes, capaces de asistir en la vigilancia y prevención de incidentes en espacios públicos y privados.

Mediante el uso de técnicas de Deep learning y transfer learning, fue posible entrenar un modelo robusto partiendo de un conjunto de datos alimentado con imágenes recolectadas desde redes sociales y el dataset especializado propuesto por Aktı et al., enfocado en situaciones de pelea. Esta combinación permitió facilitar el proceso de aprendizaje y mejorar el rendimiento general del sistema, aun en presencia de variabilidad de resolución, iluminación y movimiento.

La implementación del modelo YOLOv8 logró un equilibrio eficiente entre velocidad y precisión, permitiendo identificar con éxito a los individuos involucrados en situaciones de conflicto y diferenciarlos de los no implicados. Además, se incorporó una visualización binaria en tiempo real, donde indica que el modelo reconoce una pelea y también señala su ausencia. Esta forma de representación facilita la interpretación de los resultados, evidenciando de manera clara cuándo el sistema comprende correctamente la escena.

Este proyecto no solo reafirma el potencial de los modelos de detección de objetos en tareas de seguridad, sino que también abre nuevas oportunidades para el desarrollo de sistemas de vigilancia automatizados, escalables y adaptables a distintos entornos. Se concluye que el uso de inteligencia artificial aplicada a la videovigilancia no solo es posible, sino necesario, como parte de una estrategia

integral para enfrentar los desafíos actuales en seguridad pública. Futuras líneas de investigación podrían considerar la integración de audio, reconocimiento facial o predicción de comportamiento, para elevar aún más la capacidad del sistema en contextos urbanos complejos.

5.1.2 Avances

- Se implementó con éxito un modelo de detección basado en YOLOv8, capaz de identificar situaciones de pelea con una precisión de alrededor del 70 %.

- Las métricas de entrenamiento mostraron una disminución sostenida de la pérdida, reflejando un adecuado aprendizaje del modelo.

- Como parte del proceso de validación del sistema, se realizó una prueba en tiempo real utilizando una de las cámaras de vigilancia de la universidad, específicamente la ubicada en el Departamento de Tecnologías de la Información (DTI). En esta, se simularon situaciones de conflicto con el objetivo de evaluar la capacidad del modelo entrenado para detectar peleas. Donde se ve que los resultados fueron satisfactorios, ya que el sistema logró identificar correctamente los eventos, alcanzando una confiabilidad superior al 70 % en la detección.

5.2. Trabajo futuro

Este proyecto sienta una base sólida para la implementación de sistemas de vigilancia inteligente capaces de detectar comportamientos violentos, como peleas, en tiempo real. No obstante, se identifican diversas líneas de desarrollo que podrían mejorar significativamente el desempeño y la aplicabilidad del sistema en escenarios reales.

Una de las principales tareas pendientes es el desarrollo y ampliación del conjunto de datos (dataset)**.** El modelo fue entrenado con un número limitado de videos, muchos de los cuales presentan baja resolución, ángulos desfavorables o condiciones lumínicas deficientes. Para aumentar la robustez del modelo, se requiere construir un dataset más amplio, etiquetado con precisión y representativo de distintas condiciones de iluminación, tipos de cámaras, y contextos sociales (colegios, plazas, estaciones, etc.). Esto permitiría una generalización más efectiva del modelo y una disminución de falsos positivos o negativos.

Asimismo, trabajar con un modelo de cámara estandarizado es clave. Actualmente, se utilizaron grabaciones diversas, muchas tomadas con dispositivos móviles o sin control sobre las condiciones de captura. En futuros trabajos, emplear cámaras con parámetros técnicos conocidos (como resolución fija, tasa de cuadros estable y posición fija) permitiría al sistema ajustar mejor su detección y facilitar el entrenamiento supervisado, así como facilitar su integración en sistemas reales de vigilancia urbana.

Otra área crítica de mejora es la capacidad computacional del sistema**.** Durante este proyecto se empleó una GPU NVIDIA GTX 1650 y un procesador Intel Core i5-10300H, lo cual obligó a utilizar versiones ligeras del modelo (como YOLOv8-N). En trabajos futuros, sería recomendable utilizar una tarjeta gráfica más potente (como una NVIDIA RTX 3080 o superior), junto con un procesador de alto rendimiento (por ejemplo, Intel Core i9 o AMD Ryzen 9), lo que permitiría ejecutar modelos más complejos y precisos, como YOLOv8-L o incluso YOLOv8-X, sin sacrificar la velocidad de inferencia.

Finalmente, este trabajo podría ser expandido para detectar no solo peleas, sino también otros comportamientos de riesgo (como acoso, vandalismo o emergencias médicas), desarrollando nuevas clases dentro del modelo. Esto convertiría al sistema en una herramienta integral de seguridad ciudadana y prevención, con aplicaciones tanto en el ámbito público (municipalidades, colegios, transporte), como en entornos privados (condominios, empresas, universidades).

5.3. Posibles aplicaciones

5.3.1 Conexión directa con personal de seguridad de la Universidad

En el contexto universitario, se plantea la conexión directa del sistema con el equipo de guardias internos. Ante la detección de una pelea, se podría enviar automáticamente una alerta al dispositivo móvil o a la central de operaciones del personal de seguridad, permitiendo una intervención inmediata para prevenir eventos de violencia.

5.3.2 Integración con Carabineros y protocolos de emergencia

Un trabajo futuro sería enlazar el sistema a protocolos de respuesta de Carabineros de Chile. La generación de alertas automáticas, junto con la transmisión de la ubicación y un breve registro visual del incidente, permitiría reducir significativamente los tiempos de respuesta ante emergencias reales, mejorando la seguridad pública a gran escala.

5.3.3 Aplicación en condominios, edificios y barrios residenciales

El sistema también podría ser adaptado para su uso en condominios, edificios de departamentos y barrios residenciales, mediante una integración con sistemas de vigilancia internos, el programa podría alertar a las administraciones y vecinos de situaciones de violencia o comportamiento anómalos en áreas comunes, fortaleciendo la prevención y aumentando la percepción de seguridad ciudadana.

5.3.4 Escalabilidad y expansión funcional

Finalmente, se contempla la posibilidad de extender el sistema a la detección de otros comportamientos de riesgo, como vandalismo, robos o portación de armas. Esto convertiría la plataforma en un sistema de videovigilancia inteligente multifunción, adaptable a distintas necesidades de seguridad en tiempo real.

Referencias

[1] R. Gutter, “Percepción de inseguridad en Chile alcanza nivel histórico máximo según encuesta Enusc,” *Revista Seguridad & Defensa*, 24 de noviembre de 2023. [En línea]. Disponible en: https://revistaseguridad.cl/2023/11/24/percepcion-de-inseguridad-en-chile-alcanza-nivel-historico-maximo-segun-encuesta-enusc/

[2] Aktı, Ş., Ofli, F., Imran, M., & Ekenel, H. K. (2022). Fight detection from still images in the wild. In Proceedings of the IEEE/CVF winter conference on applications of computer vision (pp. 550-559).

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

[4] Z. Keita, "YOLO Object Detection Explained", DataCamp Blog, Sep. 14, 2022. [Online]. Available: https://www.datacamp.com/es/blog/yolo-object-detection-explained. [Accessed: Jan. 12, 2025].

[5] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 9, pp. 1904-1916, Sep. 2015. doi: 10.1109/TPAMI.2015.2389824.

[6] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Region-Based Convolutional Networks for Accurate Object Detection and Segmentation," in *Proceedings of the IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 3, pp. 429-437, Mar. 2015, doi: 10.1109/TPAMI.2015.2389824.

[7] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," *arXiv preprint arXiv:2004.10934*, Apr. 2020. [Online]. Available: https://arxiv.org/abs/2004.10934.

[8] C. Li et al., "YOLOv6: A Single-Stage Object Detection Framework for Industrial Applications," *arXiv preprint arXiv:2209.02976*, Sep. 2022. [Online]. Available: https://arxiv.org/abs/2209.02976.

[9] A. Wang, H. Chen, L. Liu, K. Chen, Z. Lin, J. Han, and G. Ding, "YOLOv10: Real-Time End-to-End Object Detection," *arXiv preprint arXiv:2405.14458*, May 2024. [Online]. Available: https://arxiv.org/abs/2405.14458.

[10] T. Cheng, L. Song, Y. Ge, W. Liu, X. Wang, and Y. Shan, "YOLO-World: Real-Time Open-Vocabulary Object Detection," *arXiv preprint arXiv:2401.17270*, Feb. 2024. [Online]. Available: https://arxiv.org/abs/2401.17270.

[11] Z. Huang, J. Wang, X. Fu, T. Yu, Y. Guo, and R. Wang, "DC-SPP-YOLO: Dense Connection and Spatial Pyramid Pooling Based YOLO for Object Detection," *arXiv preprint arXiv:1903.08589*, Sep. 2022. [Online]. Available: https://arxiv.org/abs/1903.08589.

[12] G. R. Satsangee, H. Al-Musaibeli, and R. Ahmad, "A Defect Detection Method Based on YOLOv7 for Automated Remanufacturing," *Applied Sciences*, vol. 14, no. 13, p. 5503, Jun. 2024. doi: 10.3390/app14135503.

[13] J. Chen and M. J. Er, "Dynamic YOLO for Small Underwater Object Detection," *Artificial Intelligence Review*, vol. 57, article no. 165, Jun. 2024. doi: 10.1007/s10462-024-10788-1.

[14] C. Li et al., "YOLOv6: A Single-Stage Object Detection Framework for Industrial Applications," *arXiv preprint arXiv:2209.02976*, Sep. 2022. [Online]. Available: https://arxiv.org/abs/2209.02976.

[15] Q. Fan, Y. Li, M. Deveci, K. Zhong, and S. Kadry, "LUD-YOLO: A novel lightweight object detection network for unmanned aerial vehicle," Information Sciences, vol. 686, p. 121366, Jan. 2025. doi: 10.1016/j.ins.2024.121366.

[16] A. Schcolnik-Elias, S. Martínez-Díaz, J. E. Luna-Taylor, y I. Castro-Liera, “Detección de armas tipo pistola mediante el uso de redes convolucionales con una arquitectura tipo YOLO y

estereoscopía", Pädi Boletín Científico de Ciencias Básicas e Ingenierías del ICBI, vol. 11, no Especial2, pp. 196–204, sep. 2023, doi: 10.29057/icbi.v11iespecial2.10727.

[17] J. A. Alcaide Recio, "Sistema de reconocimiento de situaciones adversas mediante visión artificial sobre un entorno DevOps," Proyecto Fin de Máster, Universidad Politécnica de Madrid, 2020.

[18] S. Garcia-Garcia y R. Pinto-Elías, "Human Activity Recognition implenting the Yolo models", en *Proceedings - 2022 International Conference on Mechatronics, Electronics and Automotive Engineering, ICMEAE 2022*, Institute of Electrical and Electronics Engineers Inc., 2022, pp. 127–132. doi: 10.1109/ICMEAE58636.2022.00029.

[19] K. Xiong, Q. Li, Y. Meng, y Q. Li, "A Study on Weed Detection Based on Improved Yolo v5", en *2023 4th International Conference on Information Science and Education, ICISE-IE 2023*, Institute of Electrical and Electronics Engineers Inc., 2023, pp. 23–26. doi: 10.1109/ICISE-IE60962.2023.10456396.

[20] B. Gong, D. Ergu, C. Ying, and B. Ma, "A Method for Wheat Head Detection Based on Yolov4," *Preprint*, Sep. 2020. [Online]. Available: https://www.researchgate.net/publication/346141981.

[21] G. Tang, T. Jiang, W. Zhou, C. Li, W. Yao, and Y. Zhao, "Adversarial Patch Attacks Against Aerial Imagery Object Detectors," *Neurocomputing*, vol. 537, pp. 128-140, Jun. 2023. doi: 10.1016/j.neucom.2023.03.050.

[22] N. Vishwakarma, "Understanding Mosaic Data Augmentation," *Analytics Vidhya*, Mar. 2024. [Online]. Available: https://www.analyticsvidhya.com/blog/2023/12/mosaic-data-augmentation/.

[23] P. Chandra, "IoU Loss Functions for Faster & More Accurate Object Detection," *LearnOpenCV*, Jun. 2023. [Online]. Available: https://learnopencv.com/iou-loss-functions-object-detection/.

[24] D. Shah, "Mean Average Precision (mAP) Explained: Everything You Need to Know," *V7 Labs Blog*, Mar. 2022. [Online]. Available: https://www.v7labs.com/blog/mean-average-

precision.

[25] S. Li, C. He, R. Li, and L. Zhang, "A Dual Weighting Label Assignment Scheme for Object Detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 9387-9393. [Online]. Available: https://openaccess.thecvf.com/content/CVPR2022/papers/Li_A_Dual_Weighting_Label_Assignment_Scheme_for_Object_Detection_CVPR_2022_paper.pdf.

[26] C. Bhalerao, "A Deep Dive Into Non-Maximum Suppression (NMS)," *Built In*, Nov. 2023. [Online]. Available: https://builtin.com/machine-learning/non-maximum-suppression.

[27] GeeksforGeeks, "CNN | Introduction to Pooling Layer," *GeeksforGeeks*, Apr. 2025. [Online]. Available: https://www.geeksforgeeks.org/cnn-introduction-to-pooling-layer/.

[28] E. A., "Detección de objetos con YOLO: implementaciones y cómo usarlas," *Medium*, 12 de mayo de 2018. [En línea]. Disponible en: https://medium.com/@enriqueav/detección-de-objetos-con-yolo-implementaciones-y-como-usarlas-c73ca2489246.

[29] M. S. Aliakbarian et al., "A survey on violence detection from video," *Computer Vision and Image Understanding*, vol. 195, 2020.

[30] A. Ojeda, "Encuesta IPSOS y Fundación Paz Ciudadana," Paz Ciudadana, Santiago, Chile, mayo 2025. [En línea]. Disponible en: https://pazciudadana.cl/noticias/encuesta-ipsos-y-fundacion-paz-ciudadana/.