INSTITUTO TECNOLÓGICO DE LA PAZ
DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN
MAESTRÍA EN SISTEMAS COMPUTACIONALES

DETECCIÓN DE ARMAS TIPO PISTOLA MEDIANTE EL USO DE REDES CONVOLUCIONALES CON UNA ARQUITECTURA TIPO YOLO Y ESTEREOSCOPÍA

QUE PARA OBTENER EL GRADO DE
MAESTRO EN SISTEMAS COMPUTACIONALES

PRESENTA:
AARON SCHCOLNIK ELIAS

DIRECTOR DE TESIS:
DR. SAÚL MARTÍNEZ DÍAZ

LA PAZ, BAJA CALIFORNIA SUR, MÉXICO, AGOSTO 2023.

La Paz, B.C.S., 07/ AGOSTO /2023

DEPI_MSC/053/2023

ASUNTO: Autorización de impresión

C. AARÓN SCHCOLNIK ELÍAS,
ESTUDIANTE DE LA MAESTRÍA EN SISTEMAS COMPUTACIONALES,
P R E S E N T E .

Con base en el dictamen de aprobación emitido por el Comité Tutorial de la Tesis denominada: **"DETECCIÓN DE ARMAS TIPO PISTOLA MEDIANTE EL USO DE REDES CONVOLUCIONALES CON UNA ARQUITECTURA TIPO YOLO Y ESTEROESCOPÍA"**, mediante la opción de tesis (Proyectos de Investigación), entregado por usted para su análisis, le informamos que se **AUTORIZA** la impresión.

A T E N T A M E N T E
Excelencia en Educación Tecnológica®

JUDITH GUADALUPE MARTÍNEZ TIRADO,
JEFA DE LA DIV. DE ESTUDIOS DE POSGRADO E INV.

EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA
TECNOLÓGICO NACIONAL DE MÉXICO
INSTITUTO TECNOLÓGICO DE LA PAZ
DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

c.c.p. Depto. de Servicios Escolares
c.c.p. Archivo.

*JGMT/icl**

DICTAMEN DEL COMITÉ TUTORIAL

La Paz, B.C.S., **04/AGOSTO/ 2023**

JUDITH GUADALUPE MARTÍNEZ TIRADO,
JEFA DE LA DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN,
P R E S E N T E.

Por medio del presente, enviamos a usted dictamen del Comité Tutorial de tesis para la obtención del grado de Maestro, con los siguientes datos generales:

No. de Control	Nombre
M21310004	AARÓN SCHCOLNIK ELÍAS
Maestría en:	SISTEMAS COMPUTACIONALES
Título de la tesis: DETECCIÓN DE ARMAS TIPO PISTOLA MEDIANTE EL USO DE REDES CONVOLUCIONALES CON UNA ARQUITECTURA TIPO YOLO Y ESTEROESCOPÍA	
DICTAMEN: Se autoriza el trabajo de investigación, en virtud de que realizó las correcciones correspondientes conforme a las observaciones planteadas por este Comité Tutorial.	

A t e n t a m e n t e.
El Comité Tutorial

MC. JORGE ENRIQUE LUNA TAYLOR

MSC. ILIANA CASTRO LIERA

DR. SAÚL MARTÍNEZ DÍAZ

c.c.p. Coordinador de la Maestría.
c.c.p. Departamento de Servicios Escolares.
c.c.p. Estudiante.

Boulevard Forjadores de B.C.S. #4720, Col. 8 de Octubre 1ra Sección, C.P. 23080, La Paz, B.C.S. Tel. (612) 12 10424 e-mail: depi_paz@tecnm.mx | lapaz.tecnm.mx

Dedicatoria

Me gustaría dedicar este trabajo a mi madre, Martha Lorena Elías Talamantes, por haberme apoyado en todo momento de mi vida y en toda decisión que he tomado, además de ayudarme a ser la persona que soy.

A mis abuelos, Aurelio Elías Lucero y Luisa Clementina Talamantes Taylor, por siempre estar para mí bajo cualquier circunstancia y siempre brindarme el apoyo que necesitara y también por ayudar a mi madre a hacerme la persona que soy.

Y a mi pareja, Samantha Daniela Vázquez Corona, ya que si no fuera por ella no hubiera empezado este proyecto en mi vida, gracias por tu constante apoyo.

Agradecimientos

Al Consejo Nacional de Ciencia y Tecnología (CONACYT), por el soporte académico brindado mediante la beca de estudios de posgrado.

Al Tecnológico Nacional de México (TecNM) campus La Paz, por la provisión de equipo de computo para las prácticas de este artículo.

Agradezco especialmente a mi director de tesis, el Dr. Saúl Martínez Díaz y a los miembros de mi comité tutorial, el M. en C. Jorge Enrique Luna Taylor y la M. en C. Iliana Castro Liera, por sus aportes y el gran apoyo que recibí de su parte en todo momento.

A mi madre, Martha Lorena Elías Talamantes, mi abuela, Luisa Clementina Talamantes Taylor y a mi padre Osías Schcolnik Corral, por apoyarme en todo mi camino como estudiante y motivarme a alcanzar este objetivo.

Y a mi pareja, Samantha Daniela Vázquez Corona, por ser la persona que me convenció a aceptar este reto y por ser mi mayor apoyo durante esta etapa de mi vida, gracias por todo.

Resumen

Las cámaras de seguridad y los sistemas de videovigilancia se han convertido en infraestructuras sumamente importantes para garantizar la seguridad de los ciudadanos. Sin embargo, el problema de inseguridad sigue en crecimiento debido en gran parte al fácil acceso a armas de fuego tipo pistola por lo que, la detección temprana de este tipo de armas es de suma importancia para ayudar a prevenir accidentes. El objetivo de este trabajo es implementar un sistema de visión estereoscópica que sea capaz de detectar con alta confianza y en tiempo real este tipo de objetos, además de lograr definir la distancia a la que se encuentra. Para dicha detección se implementó una arquitectura de redes neuronales convolucionales (CNN) con un algoritmo tipo YOLO, empleando aprendizaje transferido, junto con un algoritmo para la estimación estereoscópica de la distancia. El sistema presentado funciona con un valor de Intersección sobre unión (IoU) de 0.6, en el cual el se logró una precisión de 92.2 %, además, el error promedio detectado en la estimación de distancia, hasta un máximo de 3 metros, es de 9.3 centímetros.

Abstract

Security cameras and video surveillance systems have become extremely important infrastructures to guarantee the safety of citizens. However, the problem of insecurity continues to grow in large part due to easy access to pistol-type firearms, the early detection of these types of weapons is extremely important to help prevent accidents. The objective of this work is to implement a stereoscopic vision system that is capable of detecting this type of object in real time with high confidence, in addition to being able to define the distance at which said object is located. For this detection, a convolutional neural network (CNN) architecture was implemented with a YOLO-type algorithm using transfer learning together with an algorithm for stereoscopic distance estimation. The presented system works with an Intersection over Union (IoU) value of 0.6 in which an accuracy of 92.2 % was achieved. In addition, the average error detected in estimating distance up to a maximum of 3 meters is 9.3 centimeters.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Antecedentes

Uno de los principales problemas de muchas ciudades en el mundo es la inseguridad; en México en los últimos años, la violencia y la delincuencia han aumentado significativamente, lo que ha generado preocupación en la población. Según el Instituto Nacional de Estadística y Geografía (INEGI), en 2020 se registraron 36,579 homicidios en México, de los cuales 25,178 fueron llevados a cabo con armas de fuego [10]. Además, la Encuesta Nacional de Victimización y Percepción sobre Seguridad Pública (ENVIPE) de 2020 reveló que el 78.6 % de la población de 18 años y más considera que vivir en su entidad federativa es inseguro a consecuencia de la delincuencia [11]. Uno de los métodos más utilizados para combatir la inseguridad y disminuir el número de crímenes cometidos es el uso de sistemas de videovigilancia. Estos sistemas permiten la captura y el análisis de imágenes y videos en tiempo real, lo que puede ayudar a prevenir y detectar delitos, así como a mejorar la respuesta de las autoridades ante situaciones de emergencia. De acuerdo a [12] el uso de estos sistemas tiene como resultado una disminución en la cantidad de delitos y de arrestos, sin embargo, la detección de situaciones de alto riesgo a través de estos sistemas todavía se realiza de forma manual en muchas ciudades [13].

Entonces, de acuerdo a la información anterior, las armas de fuego son uno de los objetos más importantes y peligrosos que se pueden encontrar dentro de una escena de crimen, por lo que si se identifica un arma de fuego dentro de una secuencia de video es posible que se esté llevando a cabo un acto criminal. Si bien algunos países permiten el porte abierto de armas de fuego, en tal caso, aún es aconsejable llamar la atención a los operadores del sistema de

videovigilancia para evaluar la situación. Una vez que se detecta el arma es importante que el sistema sea capaz de rastrear dicha arma dentro de las imágenes subsecuentes para así no perder de vista al posible criminal dentro de la escena. En particular, una solución innovadora a este problema es equipar a las cámaras de vigilancia o control con un sistema de alerta automático preciso de detección de armas de fuego mediante el uso de la visión artificial [14].

Los antecedentes de la visión artificial en el reconocimiento de armas de fuego se remontan a la década de 1990, cuando se comenzaron a utilizar técnicas de procesamiento de imágenes para la detección de armas en imágenes estáticas. Sin embargo, el reconocimiento de armas en videos de vigilancia en tiempo real representa un desafío mucho mayor debido a la complejidad y variabilidad de las escenas en movimiento.

En trabajos recientes como [15], los autores desarrollaron un software preciso para la detección de pistolas en imágenes RGB. Sin embargo, su método no puede detectar varias pistolas en la misma escena. La técnica utilizada consiste en, primero, eliminar objetos no relacionados con una pistola de la imagen segmentada utilizando el algoritmo de agrupamiento K-medias y luego, aplicar el método SURF (Speeded Up Robust Features) para detectar puntos de interés. En la última década el campo de visión artificial se vio afectado de manera dramática gracias al uso de aprendizaje profundo en general y las redes convolucionales (CNNs) donde estas aplicadas en particular, han logrado resultados superiores a todos los métodos clásicos de aprendizaje automático en clasificación, detección y segmentación de imágenes en varias aplicaciones [14]. En este documento se aborda la detección de armas de fuego tipo pistola mediante el uso de CNNs.

1.2. Objetivos

1.2.1. Objetivo general

Diseñar un algoritmo que sea capaz de utilizar un sistema de cámaras estereoscópico para detectar armas de fuego tipo pistola en tiempo real mediante el uso de redes convolucionales.

1.2.2. Objetivos específicos

- Analizar los algoritmos actualizados para la detección y clasificación de armas en imágenes.

- Evaluar y comparar diferentes arquitecturas de CNN y técnicas de preprocesamiento de imágenes para mejorar la precisión y la velocidad de detección.
- Seleccionar e implementar un método de detección y clasificación basado en redes neuronales para la detección de pistolas en tiempo real con alta precisión y velocidad.
- Implementar un algoritmo que sea capaz de estimar la distancia a la que se encuentra el arma de fuego una vez que esta ha sido detectada en la imagen.
- Evaluar el desempeño del método elegido.

1.3. Alcance y limitaciones

Alcance:

- El método de detección de armas tipo pistola fue probado y aplicado en los laboratorios de computación del Instituto Tecnologico de La Paz (ITLP) por lo que su funcionamiento y uso esta acotado a estas zonas en específico.

Limitaciones:

- No se llevaron a cabo pruebas en sistemas de bajo costo y/o baja capacidad de procesamiento, por lo que es difícil definir si este algoritmo es capaz de funcionar de manera óptima en estos mismos.
- La cantidad y la calidad de las imágenes. La obtención de imágenes de buena calidad es un proceso de suma importancia para la etapa de entrenamiento en la red, sin embargo, encontrar imágenes en las que se encuentre presente un arma tipo pistola en una situación real es difícil debido a su escasez.

1.3.1. Justificación

Los sistemas de videovigilancia deben de trabajar de manera constante y en tiempo real, sin embargo, en la mayoría de los casos, la revisión de estos sistemas sigue siendo llevada a cabo por personal de seguridad, lo cual implica una inversión bastante alta debido a los horarios que deben ser cubiertos. Además, puede ser difícil para el personal de seguridad mantenerse

completamente alerta al revisar manualmente los sistemas, esto debido a varias razones como la monotonía del trabajo, distracciones o incluso por simple error humano.

1.3.2. Hipótesis

Se podrá desarrollar un algoritmo basado en redes convolucionales (CNN) para la detección de pistolas que utilice técnicas como el aumento de datos para mejorar el rendimiento, también el uso de aprendizaje transferido mejorará la precisión y velocidad del algoritmo. Además, el algoritmo podría ser diseñado para detectar no solo la presencia de una pistola, sino también su orientación, posición en la imagen y la distancia real a la que se encuentra la pistola, lo que podría ser útil para aplicaciones de seguimiento y vigilancia.

1.3.3. Estado del arte

Como se presentó con anterioridad en los antecedentes, los métodos clásicos utilizados para la detección y clasificación de armas se basaban en la detección de esquinas en el objeto de importancia [15]. Estos modelos logran funcionar de manera rápida, pero algunos de sus principales problemas son su baja precisión, lo susceptible que son a los cambios de iluminación en la imagen y el hecho de que no logran detectar múltiples armas en la misma imagen. En años más recientes, se ha cambiado por completo el enfoque. El campo es básicamente dominado por el uso de CNNs. Existen múltiples arquitecturas diferentes para llevar a cabo la detección y clasificación de objetos de interés. Uno de los primeros artículos que enfocaron el uso de CNNs a la detección y clasificación de armas de fuego fue [14], en donde utilizaron una arquitectura VGG-16 y Faster R-CNN. En este trabajo, se logró una precisión promedio del 84.21 % y fue probado utilizando videos pregrabados. Se logró que el modelo presentara una alarma de presencia de armas en un lapso de tiempo menor a 0.2 segundos. Pero su aporte más importante fue presentar una base de datos pública de imágenes de armas de fuego tipo pistola, la cual consta de 3000 imágenes de diferentes modelos de este tipo. En [16] se aplicó un algoritmo llamado "Single Shot Multibox Detector (SSD)" en conjunto con una CNN conocida como MobileNet para detectar cinco diferentes armas de fuego: rifle automático, rifle francotirador, escopeta, arma tipo pistola y ametralladora. Las ventajas de utilizar esta arquitectura en conjunto con el algoritmo SSD es el hecho de que su velocidad es extremadamente rápida debido al poco peso de

la arquitectura, sin embargo, el proyecto se vio afectado por una baja precisión, logrando como máximo una precisión promedio de 85.24 %. Como trabajo más reciente en [17] se presentan tres modelos diferentes basados en CNNs: YOLOv3, RetinaNet y Faster R-CNN, para la detección automática de armas de fuego tipo pistola. Dichos modelos fueron entrenados de dos maneras diferentes, la primera a través de imágenes en donde únicamente se anotó la presencia de una pistola, y el segundo entrenamiento se llevó a cabo con la anotación de la pistola y presentándole la pose de la persona que cuenta con la pistola en su mano. El mejor resultado fue presentado por el primer método de entrenamiento donde su precisión promedio fue de 97.23 % y se obtuvo con el modelo RetinaNet. Igualmente es importante destacar que el uso de la información de la pose causó que el modelo basado en YOLOv3 mejorara su precisión mientras que causó que el modelo RetinaNet se viera afectado negativamente. A pesar de la gran precisión alcanzada por sus modelos el artículo no presenta ejemplos, ni hace mención, sobre la capacidad de utilizarlos en tiempo real. En [18], presentan una comparativa en el uso de las arquitecturas tipo YOLO para la detección de armas en tiempo real. Las arquitecturas presentadas son YOLOv3 y YOLOv4 y la métrica que se está intentando comparar es su precisión tras ser entrenadas con la misma base de datos. Los resultados que obtuvieron es que YOLOv4 cuenta con una mayor precisión que YOLOv3. La diferencia entre la precisión no fue muy grande, se presenta que YOLOv4 cuenta con una precisión del 85 %, mientras que YOLOv3 presenta una precisión del 84 %. Sin embargo, donde podemos ver el mayor cambio es en la métrica "Mean Average Precision (mAP)" (precisión promedio en cada clase). En este caso, los resultados fueron más prominentes. YOLOv3 presentó un mAP igual a 77.3 %, mientras que YOLOv4 presentó un mAP de 84.85 %. Lo anterior presenta el cómo cada nueva versión del modelo YOLO mejora tanto la velocidad de procesamiento como la precisión de la detección.

Capítulo 2

Marco teórico

2.1. Visión por computadora

La visión por computadora es un campo de la inteligencia artificial y la informática que se enfoca en permitir que las computadoras interpreten y comprendan la información visual del mundo que las rodea. La visión por computadora implica el desarrollo de algoritmos y técnicas que permiten a las computadoras analizar, interpretar y reconocer imágenes y videos. El objetivo de la visión por computadora es crear máquinas que puedan percibir y comprender el mundo visual de la misma manera que lo hacen los humanos. Esto implica el desarrollo de algoritmos que puedan reconocer objetos, detectar patrones y comprender el contexto de la información visual.

2.1.1. OpenCV

OpenCV (Open Source Computer Vision) es una biblioteca de software libre y código abierto de visión por computadora y aprendizaje automático. Fue introducida en el año 2000 por Intel como una biblioteca de funciones de procesamiento de imágenes en tiempo real para mejorar la eficiencia en aplicaciones de visión por computadora. Antes de OpenCV, las bibliotecas de visión por computadora eran costosas y no estaban disponibles para todos los desarrolladores. OpenCV ofrece una amplia variedad de algoritmos y herramientas para procesamiento de imágenes y visión por computadora, incluyendo detección de objetos, seguimiento de objetos, reconocimiento de patrones, segmentación de imágenes, calibración de cámaras, entre otros. Además, OpenCV

es compatible con múltiples lenguajes de programación, incluyendo C++, Python y Java, lo que la hace una herramienta versátil y accesible. En la actualidad se ha convertido en una de las bibliotecas de visión por computadora más populares.

2.1.2. Etapas de un sistema de visión para detección de objetos

La visión computacional se puede resumir como el conjunto de procesos de adquisición, caracterización y análisis de la información de las imágenes obtenidas del entorno 3D [19] (figura 2.1) La adquisición de imagen simplemente corresponde a la acción de capturar las imágenes

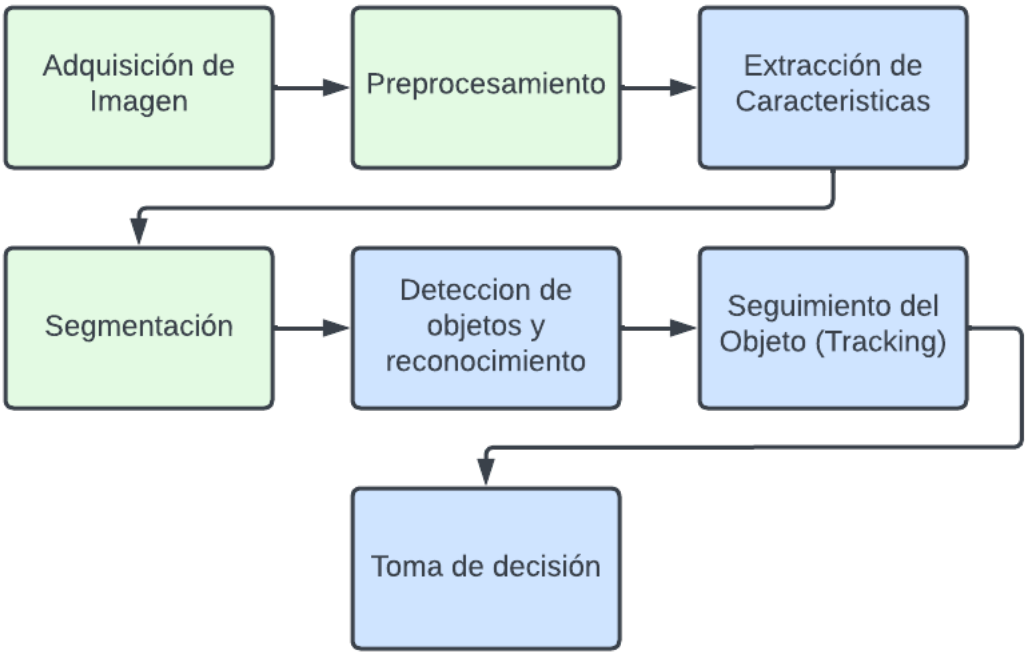


Figura 2.1: Etapas de un sistema de visión artificial para detección de objetos.

mediante el uso de cualquier dispositivo que cuente con una cámara, escáner, etc.

2.1.3. Preprocesamiento

El preprocesamiento de imágenes es un conjunto de técnicas y operaciones que se aplican a una imagen digital antes de ser procesada por un algoritmo de visión por computadora. El objetivo del preprocesamiento es mejorar la calidad de la imagen, reducir el ruido y mejorar la

capacidad del algoritmo para detectar características y objetos de interés en la imagen. Algunas de las técnicas de preprocesamiento de imágenes incluyen:

1. Redimensionamiento
2. Normalización
3. Corrección de color
4. Reducción de color
5. Contraste

En general, el preprocesamiento de imágenes es una etapa importante en la visión por computadora, ya que puede mejorar significativamente la calidad y la precisión de los algoritmos de procesamiento de imágenes y detección de objetos.

2.1.4. Extracción de características

La extracción de características en un sistema de visión por computadora es el proceso de identificar y extraer las características relevantes de una imagen que son útiles para la tarea de análisis o reconocimiento de objetos. Estas características pueden ser formas, texturas, colores, bordes u otros patrones visuales que permiten distinguir entre diferentes objetos o clases de objetos. Esta etapa permite reducir la cantidad de información en la imagen y enfocarse en las características más relevantes para la tarea específica. Además, la extracción de características puede mejorar la precisión y la eficiencia de los algoritmos de análisis y reconocimiento de objetos.

2.1.5. Segmentación

La segmentación de imágenes en un sistema de visión computacional es un proceso que se utiliza para dividir una imagen en diferentes regiones o segmentos con características similares. El objetivo de la segmentación es identificar y separar las regiones de interés de la imagen del resto de la imagen, para facilitar la detección, reconocimiento y análisis de objetos. Podemos dividir la segmentación de imágenes en tres tipos diferentes:

1. Segmentación de instancias

2. Segmentación panóptica

3. Segmentación semántica

La segmentación de instancias es una técnica de visión por computadora que se utiliza para identificar y separar objetos individuales en una imagen. A diferencia de la segmentación semántica, que asigna una etiqueta o categoría a cada píxel de la imagen, la segmentación de instancias identifica y separa cada objeto individual en la imagen, incluso si los objetos pertenecen a la misma categoría [1]. Se realiza mediante la asignación de una etiqueta única a cada objeto en la imagen. Cada objeto se identifica por su posición y forma, y se separa del resto de la imagen. Esta técnica se utiliza en muchas aplicaciones de visión por computadora, como la detección de objetos en tiempo real, la robótica y la conducción autónoma, la figura 2.2 muestra la estructura general de estas técnicas. El enfoque popular para la segmentación de instancias implica la detección de objetos mediante una caja seguida de la segmentación de la caja de objeto [20], [21], [22].

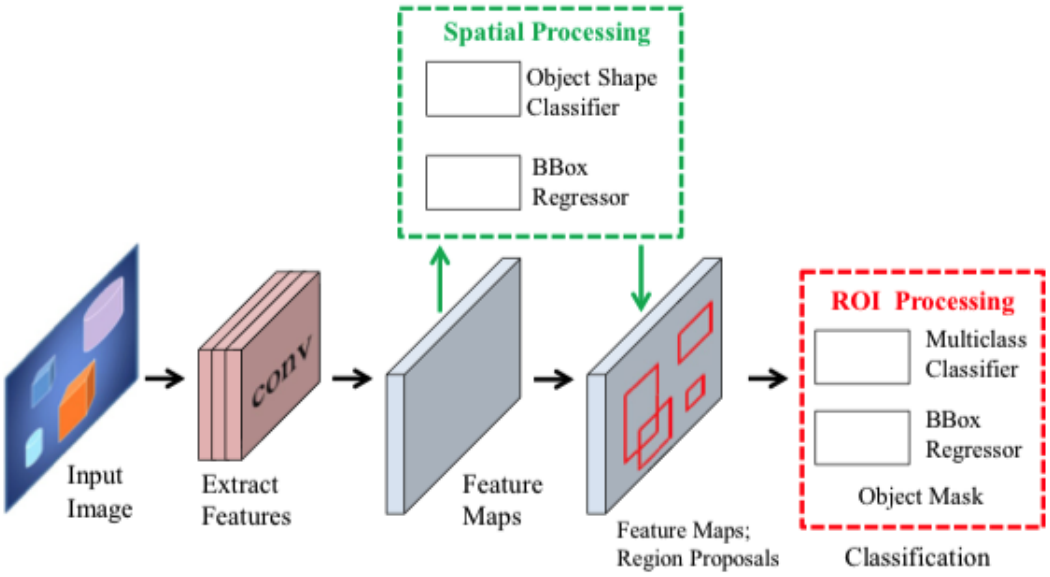


Figura 2.2: Estructura general para la detección seguida por técnicas de segmentación [1].

La segmentación de instancias se puede realizar utilizando técnicas de aprendizaje profundo, como las redes neuronales convolucionales (CNN), que han demostrado ser muy efectivas en la tarea de segmentación de instancias. Estas técnicas se basan en la detección de características

específicas de cada objeto, como su forma, tamaño y textura, y utilizan esta información para separar los objetos individuales en la imagen. La segmentación de instancias basada en aprendizaje profundo se puede clasificar utilizando el número de etapas de su estructura, de acuerdo a esto lo podemos dividir en dos categorías importantes:

1. De una sola etapa

2. De dos etapas (2-stage)

En nuestro caso trabajaremos con un sistema single stage (figura 2.3) [5], estos sistemas suelen ser más rápidos que sus contrapartes de dos etapas debido a que no requiere de preprocesamiento, una red de carácter ligero, un menor número de regiones candidatas y el uso de una subred de detección totalmente convolucional [1].

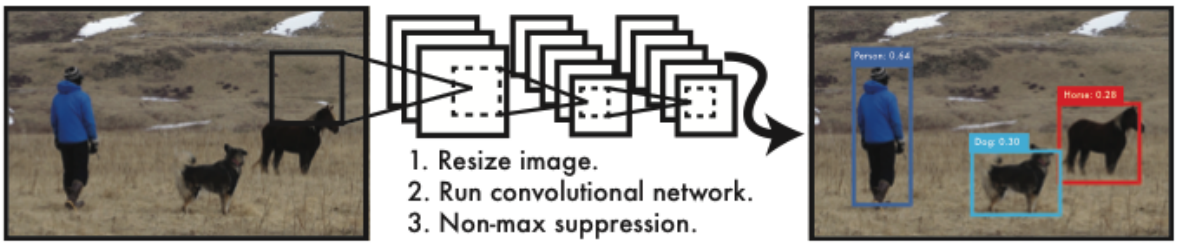


Figura 2.3: Ejemplo de un sistema single stage conocido como YOLO (You Only Look Once).

2.1.6. Detección de objetos

Después de la segmentación de imágenes, se pueden aplicar algoritmos de detección de objetos a las regiones segmentadas para detectar y clasificar objetos según sus características visuales. La detección de objetos generalmente implica dos pasos principales: localización de objetos y clasificación de objetos. En el paso de localización de objetos, el algoritmo identifica la ubicación del objeto dentro de la región segmentada, a menudo mediante la detección del cuadro delimitador o contorno del objeto. En el paso de clasificación de objetos, el algoritmo asigna una etiqueta o categoría al objeto según sus características visuales, como su forma, color o textura. La detección se puede lograr mediante una variedad de técnicas, que incluyen algoritmos de visión por computadora tradicionales y enfoques basados en aprendizaje profundo. Los algoritmos tradicionales a menudo dependen de características y clasificadores creados a

mano, mientras que los enfoques basados en aprendizaje profundo utilizan CNNs para aprender automáticamente características y clasificadores a partir de los datos. La detección de objetos es un componente crítico en muchos sistemas de visión por computadora, ya que permite al sistema identificar y rastrear objetos de interés en entornos del mundo real.

2.1.7. Seguimiento del objeto

El seguimiento de objetos a menudo se realiza después de la detección de objetos en un sistema de visión por computadora, ya que la detección proporciona la ubicación inicial del objeto en el primer fotograma del video o secuencia de imágenes.

Una vez que se ha detectado un objeto en el primer fotograma, se pueden aplicar algoritmos de seguimiento de objetos para rastrear el movimiento del objeto a lo largo del tiempo. Estos algoritmos suelen utilizar una combinación de características del objeto y modelos de movimiento para predecir la ubicación del objeto en fotogramas posteriores del video o secuencia de imágenes. La ubicación del objeto se actualiza en cada fotograma mediante la correspondencia entre la ubicación predicha y la ubicación real del objeto en la imagen.

El seguimiento de objetos se puede lograr utilizando una variedad de técnicas, que incluyen el seguimiento basado en correlación, el seguimiento basado en características y el seguimiento basado en aprendizaje profundo. El seguimiento basado en correlación implica calcular la similitud entre el objeto en el fotograma actual y el objeto en el fotograma anterior, mientras que el seguimiento basado en características utiliza características distintivas del objeto, como sus esquinas o bordes, para rastrear su movimiento. El seguimiento basado en aprendizaje profundo utiliza redes neuronales convolucionales (CNN) para aprender la apariencia del objeto y los patrones de movimiento y predecir su ubicación en fotogramas posteriores.

2.1.8. Toma de decisión

Una vez que se ha detectado, clasificado y completado el seguimiento de un objeto, el sistema debe decidir qué acción tomar en función de la información obtenida.

La toma de decisiones puede involucrar diferentes acciones, dependiendo de la aplicación específica del sistema de visión por computadora. Por ejemplo, en un sistema de vigilancia, la toma de decisiones podría implicar alertar a un operador o activar una alarma si se detecta un

objeto de interés.

2.2. Redes Neuronales Artificiales (ANN)

Las redes neuronales artificiales son, como su nombre indica, redes computacionales que intentan simular el proceso de decisión en redes de células nerviosas (neuronas) (figura 2.4) del sistema nervioso central biológico (humano o animal). Esta simulación es una simulación a grandes rasgos de célula a célula (neurona por neurona, elemento por elemento). Se basa en el conocimiento neurofisiológico de las neuronas biológicas y de redes de estas neuronas biológicas [23].

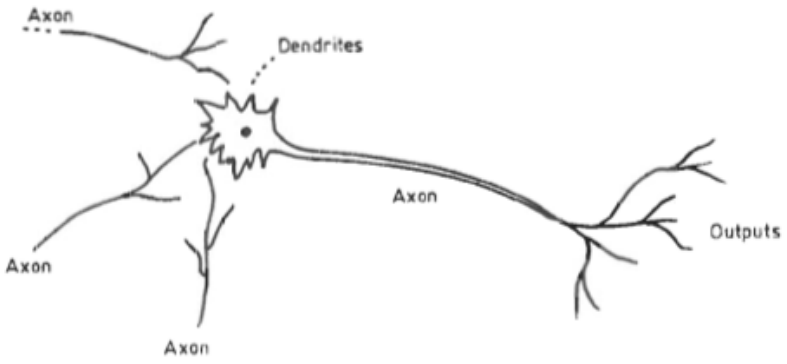


Figura 2.4: Representación gráfica de una neurona biológica

Son caracterizadas por contar con elementos de entrada, conexiones entre componentes llamados neuronas, un conjunto de pesos (en analogía a las sinapsis en el sistema nervioso), generando información que viaja paralelamente a través de la arquitectura neuronal siendo objeto de la aplicación de funciones intermedias, e involucrando un entrenamiento que en cada ocasión podrá ajustarse para alcanzar la salidas adecuada [19]. El componente mas básico de una red neuronal artificial es conocida como perceptrón.

2.2.1. El Perceptrón

El perceptrón, que históricamente es posiblemente la primera neurona artificial, se propuso en 1958, también es la unidad básica de casi todas las redes neuronales artificiales.

Su modelo se asemeja en estructura a una neurona biológica (figura 2.4) como en sus entradas

ponderadas cuyos pesos son ajustables y en su disposición para una salida que es una función de la entrada ponderada anterior a la neurona biológica como en la figura 2.5 [23].

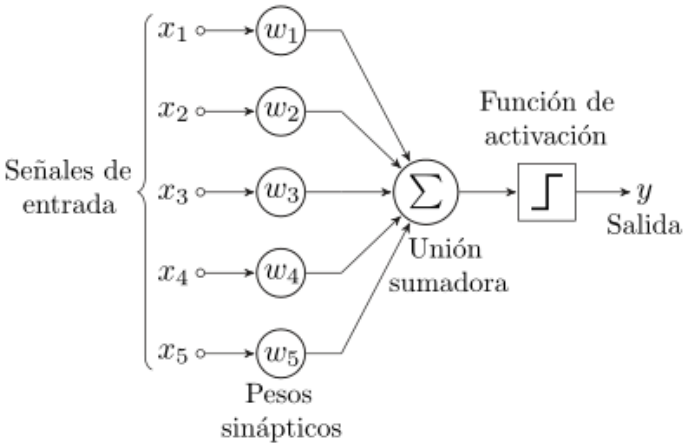


Figura 2.5: Un perceptrón

2.2.2. Funciones de activación de un perceptrón

La información en la salida del perceptrón difiere de la salida de suma, esto se debe a que la salida de un perceptrón se pasa a través de una función de activación, que introduce no linealidad en el modelo. La función de activación permite que el perceptrón aprenda funciones más complejas al mapear la salida de la suma ponderada de las entradas a un rango no lineal. Esta no linealidad es necesaria porque muchos problemas del mundo real no son linealmente separables, y un perceptrón con solo una función de activación lineal no podría aprender estas funciones. Por lo tanto, la función de activación es un componente clave de un perceptrón que le permite aprender funciones más complejas que una simple función de suma. En la figura 2.6 se muestran algunas de las funciones de activación que son utilizadas comúnmente.

2.2.3. Perceptrón multicapa

En 1969, Minsky y Papert publicaron un libro [24] donde señalaban las graves limitaciones en las capacidades del perceptrón, lo cual hacen evidente a traves de su teorema de representación. Hay una gran clase de problemas que los clasificadores de una sola capa no pueden resolver. Tanto es así, que para una red neuronal de una sola capa con un número creciente de entradas, el número de problemas que se pueden clasificar se convierte en una fracción muy pequeña de la

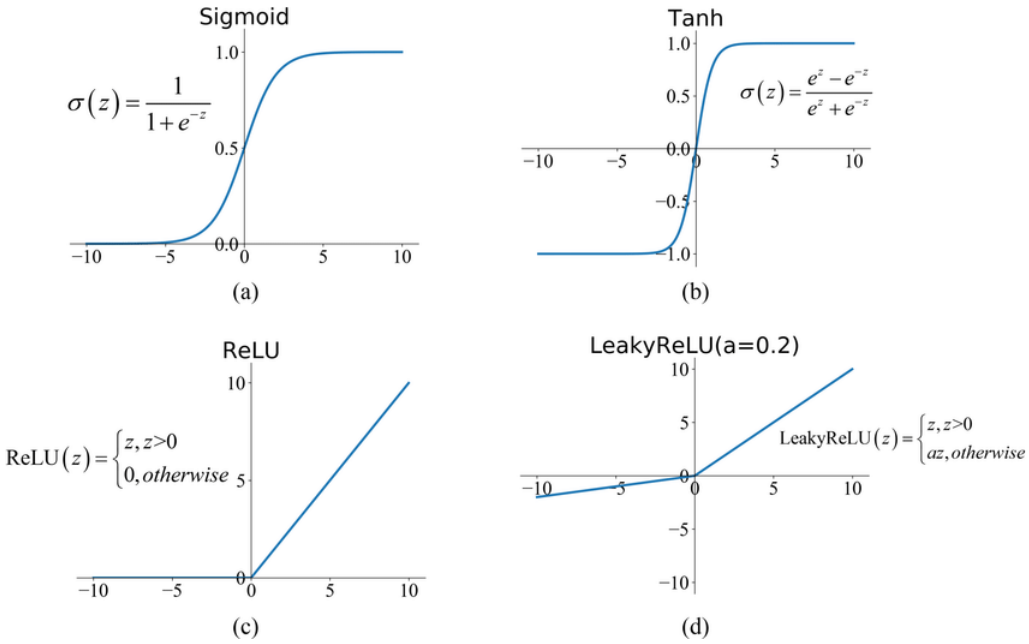


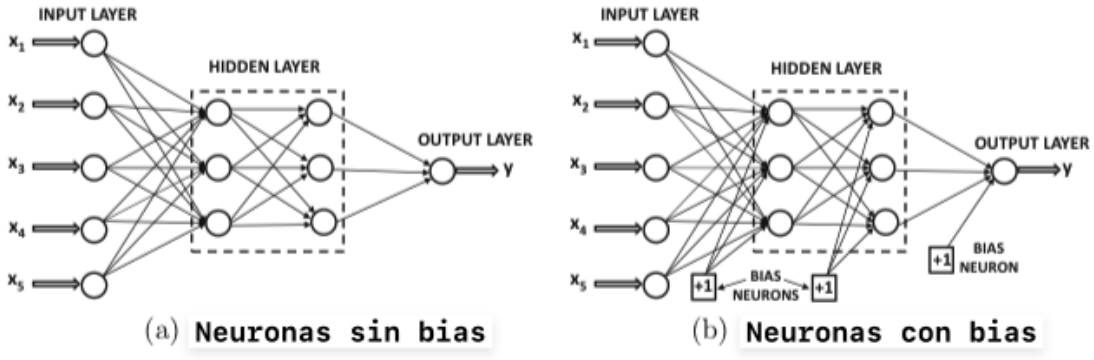
Figura 2.6: Ejemplos mas comunes de funciones de activación

totalidad de los problemas que se pueden formular. Para superar las limitaciones señaladas por Minsky y Papert, fue necesario ir más allá de la ANN de una sola capa. La estructura consiste en capa de entrada, capa oculta (formada por una o varias subcapas) y capa de salida [19]. La arquitectura específica de las redes neuronales multicapa se conoce como redes de avance (feed-forward), porque las capas sucesivas se alimentan entre sí en la dirección de avance desde la entrada hasta la salida. La arquitectura predeterminada de las redes feed-forward supone que todos los nodos de una capa están conectados a los de la siguiente capa. Por lo tanto, Definir la arquitectura de una red neuronal, requiere especificar el número de capas, la cantidad de neuronas por capa, la función de activación y la función de pérdida (figura 2.7) [25].

2.2.4. Factor de aprendizaje

El factor de aprendizaje (learning rate) es uno de los hiperparametros ajustables en los algoritmos de optimización, este determina el tamaño del paso en el que se actualizan los parámetros del modelo durante el entrenamiento.

La tasa de aprendizaje determina la cantidad en la que se ajustan los parámetros del modelo en la dirección del mínimo de la función de pérdida. Si la tasa de aprendizaje es demasiado alta, los parámetros del modelo pueden oscilar o divergir, lo que lleva a una convergencia deficiente



(a) Neuronas sin bias (b) Neuronas con bias

Figura 2.7: Arquitectura básica de una red feed-forward con dos capas ocultas

y un comportamiento inestable. Por otro lado, si la tasa de aprendizaje es demasiado baja, el modelo puede converger muy lentamente o quedarse atrapado en un mínimo local subóptimo [25].

2.2.5. Retropropagación

El algoritmo de retropropagación fue propuesto en 1986 por Rumelhart, Hinton y Williams para establecer pesos y, por lo tanto, para el entrenamiento de perceptrones multicapa.

2.2.5.1. Descenso de gradiente estocástico (SGD) y funciones de pérdida

La mayoría de los problemas de aprendizaje automático se pueden reformular como problemas de optimización sobre funciones objetivo específicas. Por ejemplo, la función objetivo en las redes neuronales se puede definir en términos de optimización de una función de pérdida L, que a menudo es una suma linealmente separable de las funciones de pérdida en los puntos de datos de entrenamiento individuales [25]. Se puede escribir la función de pérdida de una red neuronal de la siguiente forma:

$$L = \sum_{i=1}^{n} L_i \tag{2.1}$$

Aquí, L_i es la pérdida aportada por el i-ésimo punto de entrenamiento. En el descenso de gradiente, se intenta minimizar la función de pérdida de la red neuronal moviendo los parámetros a lo largo de la dirección negativa del gradiente. Por ejemplo, en el caso del perceptrón, los parámetros corresponden a $W = (w_1 \dots w_d)$. Por lo tanto, se debe intentar calcular la pérdida de la función objetivo subyacente en todos los puntos simultáneamente y realizar un descenso de

gradiente. En el descenso de gradiente tradicional, se intenta realizar pasos como los siguientes:

$$\overline{W} \Leftarrow \overline{WW} - \alpha(\frac{\partial L}{\partial w_1}, \frac{\partial L}{\partial w_2} \dots \frac{\partial L}{\partial w_d}) \tag{2.2}$$

Para redes de una sola capa como el perceptrón, el descenso de gradiente se realiza solo con respecto a $\overline{W}$, mientras que para redes más grandes, todos los parámetros de la red deben actualizarse con retropropagación. La cantidad de parámetros puede ser del orden de millones en aplicaciones a gran escala, y se necesita operar simultáneamente todos los ejemplos hacia adelante y hacia atrás a través de la red para calcular las actualizaciones de retropropagación. Dado que la función de pérdida de la mayoría de los problemas de optimización se puede expresar como una suma lineal de las pérdidas con respecto a ejemplos individuales:

$$\frac{\partial L}{\partial \overline{W}} = \sum_{i=1}^{n} \frac{\partial L_i}{\partial \overline{W}} \tag{2.3}$$

Los problemas de aprendizaje automático tienen inherentemente un alto nivel de redundancia entre el conocimiento capturado por diferentes puntos de entrenamiento y, a menudo, se puede emprender de manera más eficiente el proceso de aprendizaje con las actualizaciones específicas de puntos del descenso de gradiente estocástico:

$$\overline{W} \Leftarrow \overline{W} - \alpha \frac{\partial L_i}{\partial W} \tag{2.4}$$

Este tipo de descenso de gradiente se denomina estocástico porque este recorre los puntos en un orden aleatorio. Se debe tener en cuenta que el efecto a largo plazo de las actualizaciones repetidas es aproximadamente el mismo, aunque cada actualización en el descenso de gradiente estocástico solo puede verse como una aproximación probabilística. Cada gradiente local se puede calcular de manera eficiente, lo que hace que el descenso del gradiente estocástico sea más rápido, aunque a expensas de la precisión en el cálculo del gradiente. Sin embargo, una propiedad interesante del descenso de gradiente estocástico es que, aunque no funcione tan bien en los datos de entrenamiento (en comparación con el descenso de gradiente), a menudo funciona de manera comparable (y a veces incluso mejor) en los datos de prueba [25].

Con la finalidad de mejorar el desempeño de la red y alcanzar los valores deseados en la salida, existen distintas alternativas de funciones de pérdida para el cálculo del error entre el valor real y el obtenido. Algunos ejemplos son: Error Cuadrático Medio, Error Absoluto, Error Absoluto Medio Escalado, Entropía Cruzada Categórica, Entropía Cruzada Binaria, Varifocal [19].

2.2.6. Redes Neuronales Profundas

Las redes neuronales profundas (figura 2.8) (DNN, por sus siglas en inglés) son un tipo de red neuronal artificial diseñada para modelar patrones complejos en los datos. Están compuestas por múltiples capas de nodos interconectados, o neuronas, que procesan información y extraen características cada vez más abstractas de los datos de entrada. Los componentes básicos de una DNN incluyen capas de entrada y salida, así como una o más capas ocultas. Cada capa consta de múltiples neuronas, que están conectadas a neuronas en las capas adyacentes mediante conexiones ponderadas. Durante el entrenamiento, los pesos de estas conexiones se ajustan para minimizar la diferencia entre la salida de la red y la salida deseada. Las DNN son ampliamente utilizadas bajo aprendizaje supervisado [25].

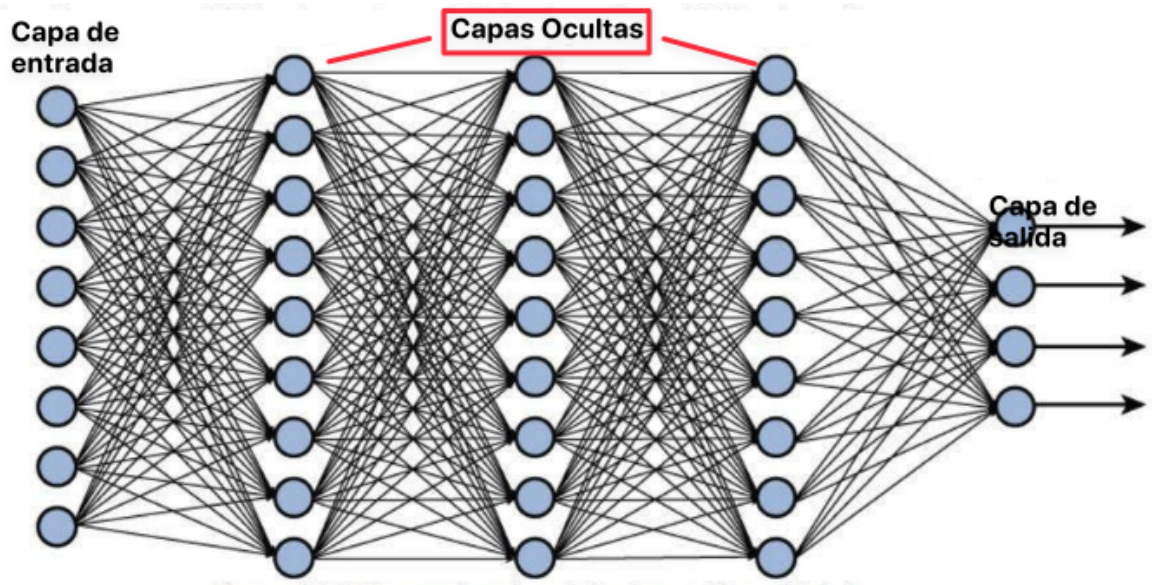


Figura 2.8: Estructura de una DNN

2.3. Redes Neuronales Convolucionales

Las redes neuronales convolucionales (CNN) están diseñadas para funcionar con entradas estructuradas en cuadrícula, que tienen fuertes dependencias espaciales en las regiones locales de la cuadrícula. El ejemplo más obvio de datos estructurados en cuadrícula es una imagen bidimensional. Este tipo de datos exhibe dependencias espaciales, porque las ubicaciones espaciales adyacentes en una imagen a menudo tienen valores de color similares de los píxeles individuales.

Una dimensión adicional captura los diferentes canales de información de color, lo que crea un volumen de entrada tridimensional. Por lo tanto, las características de una red neuronal convolucional tienen dependencias entre sí en función de las distancias espaciales. Otras formas de datos secuenciales como texto, series de tiempo y secuencias también pueden considerarse casos especiales de datos estructurados en cuadrícula con varios tipos de relaciones entre elementos adyacentes.

Una característica definitoria importante de las CNN es una operación, a la que se hace referencia como convolución. Una operación de convolución es una operación de producto escalar entre un conjunto de pesos con estructura de cuadrícula y entradas similares con estructura de cuadrícula extraídas de diferentes localidades espaciales en el volumen de entrada. Este tipo de operación es útil para datos con un alto nivel de ubicación espacial o de otro tipo, como datos de imágenes. Por lo tanto, las redes neuronales convolucionales se definen como redes que usan la operación convolucional en al menos una capa, aunque la mayoría de las redes neuronales convolucionales usan esta operación en múltiples capas.

La arquitectura tradicional de una CNN consta de múltiples capas posteriores a la de partida (Figura 2.9), los estados en cada capa se organizan de acuerdo con una estructura de cuadrícula espacial. Estas relaciones espaciales se heredan de una capa a la siguiente porque cada valor de característica se basa en una pequeña región espacial local en la capa anterior. Es importante mantener estas relaciones espaciales entre las celdas de la cuadrícula, porque la operación de convolución y la transformación a la siguiente capa dependen de manera crítica de estas relaciones. Cada capa en la red convolucional es una estructura de cuadrícula tridimensional, que tiene una altura, un ancho y una profundidad. La profundidad de una capa en una CNN no debe confundirse con la profundidad de la propia red. La palabra "profundidad"(cuando se usa en el contexto de una sola capa) se refiere a la cantidad de canales en cada capa, como la cantidad de canales de colores primarios (p. ej., azul, verde y rojo) en la imagen de entrada o el número de mapas de características en las capas ocultas [24].

La CNN funciona de manera muy similar a una red neuronal de avance tradicional, excepto que las operaciones en sus capas están organizadas espacialmente con conexiones escasas (y cuidadosamente diseñadas) entre capas. Los tipos de capas que comúnmente están presentes en una CNN son (en orden de aparicion):

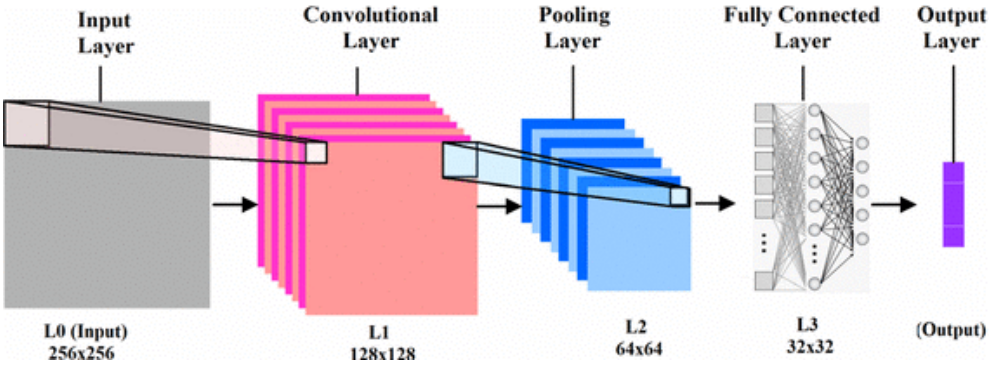


Figura 2.9: Representación visual de la arquitectura de una CNN [2].

2.3.1. Capa Convolucional

En la CNN, los parámetros se organizan en conjuntos de unidades estructurales tridimensionales, conocidas como filtros o núcleos. El filtro suele ser cuadrado en términos de sus dimensiones espaciales, que suelen ser mucho más pequeñas que las de la capa a la que se aplica el filtro (Figura 2.10). La operación de convolución coloca el filtro en cada posición posible en la imagen (o capa oculta) para que el filtro se superponga completamente con la imagen y realice un producto escalar entre los parámetros en el filtro y la cuadrícula coincidente en el volumen de entrada. Tras el haber generado un mapa de características, a este se le aplica una función de activación, la mas comúnmente utilizada siendo ReLU o LeakyReLU. La activación de ReLU no es diferente de una red neuronal tradicional. Además, un conjunto final de capas a menudo está completamente conectado y se asigna de una manera específica de la aplicación a un conjunto de nodos de salida.

2.3.2. Normalización por lotes

La normalización por lotes (Batch Normalization en inglés) es una técnica de regularización que se utiliza en redes neuronales profundas para normalizar las entradas de cada capa en mini lotes durante el entrenamiento. Esto ayuda a reducir el cambio de covariables internas de la red neuronal y acelera el entrenamiento. En la normalización por lotes, se calcula la media y la desviación estándar de cada característica en un mini lote y se normalizan las entradas de la capa utilizando estos valores. Luego se aplican dos parámetros de escala y cambio para permitir que la red neuronal aprenda la mejor representación de los datos normalizados. La normalización por lotes ha demostrado ser efectiva en mejorar la precisión y la velocidad de convergencia de

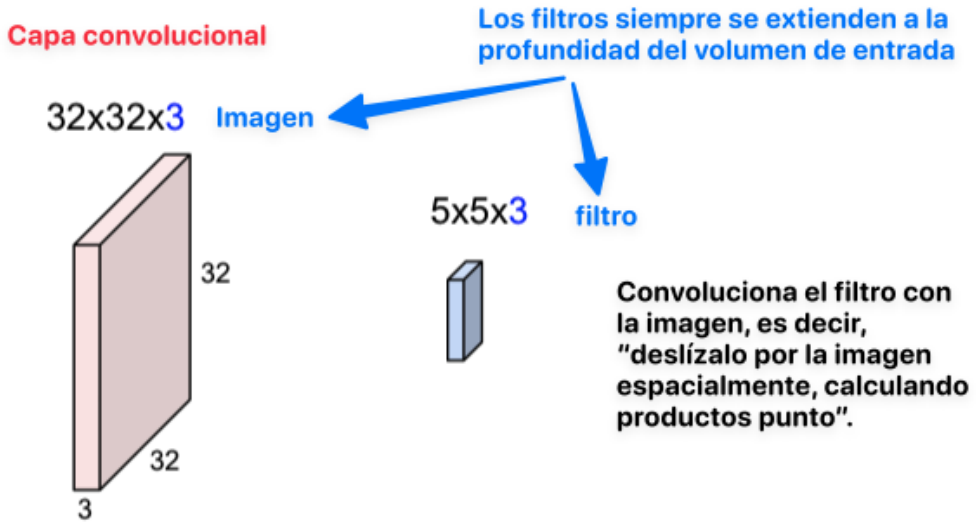


Figura 2.10: Uso de un filtro en una convolución

las redes neuronales profundas.

2.3.3. Capa de agrupamiento

Una capa de agrupamiento es un componente importante de una red neuronal convolucional (CNN) que se utiliza para reducir las dimensiones espaciales del volumen de salida de la capa convolucional anterior. La capa de agrupamiento funciona dividiendo la imagen de entrada en un conjunto de regiones rectangulares no superpuestas y luego calculando el máximo, promedio o suma de cada región (Figura 2.11). Este proceso reduce el tamaño de los mapas de características, lo que ayuda a reducir la carga computacional y el sobreajuste. Hay diferentes tipos de capas de agrupamiento, como la agrupación máxima, la agrupación promedio y la agrupación L2, y se pueden usar de diferentes maneras dependiendo de la aplicación específica y la arquitectura de la red.

2.3.4. Capa completamente conectada

Es un tipo de capa de red neuronal en la que cada neurona está conectada con todas las neuronas de la capa anterior, lo que permite la transformación de características de alto nivel extraídas de capas anteriores en puntajes de clase o predicciones. En una CNN, las capas totalmente conectadas se colocan típicamente al final de la red, tomando la salida de las capas

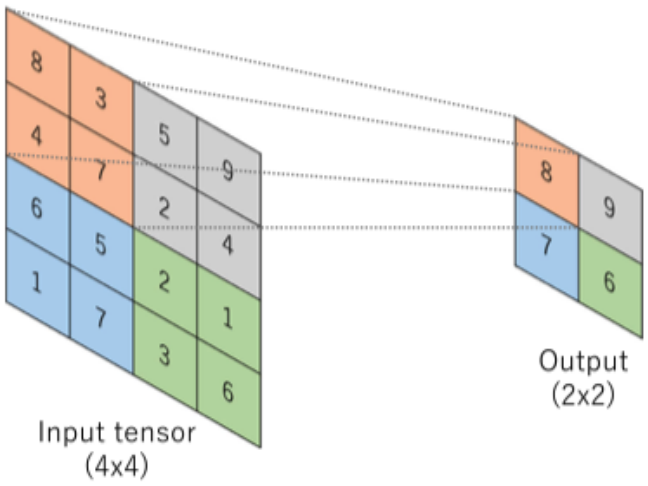


Figura 2.11: Funcionamiento de una capa de agrupamiento máximo [3]

convolucionales y de agrupamiento y aplanándola en un vector, que luego se alimenta a la capa totalmente conectada. La capa totalmente conectada luego aplica un conjunto de pesos y sesgos a cada neurona en el vector para producir la salida final. La capa totalmente conectada es una herramienta poderosa para aprender relaciones no lineales complejas entre los datos de entrada y salida.

2.4. Arquitecturas de Redes Convolucionales comunmente utilizadas para la detección de objetos

2.4.1. Faster R-CNN

Esta arquitectura fue presentada en el 2015 [4] y funciona introduciendo una Red de Propuesta de Regiones (RPN) que comparte características convolucionales con la red de detección de objetos. La RPN genera propuestas de regiones deslizando una pequeña red sobre el mapa de características convolucionales de salida de la red principal. Estas propuestas se refinan y clasifican luego por la red de detección de objetos, que se implementa como una red Fast R-CNN. El uso de la RPN permite que Faster R-CNN logre un rendimiento de detección de objetos de vanguardia, siendo más rápido y preciso que los modelos anteriores.

Si bien Faster R-CNN es un modelo de detección de objetos poderoso y ampliamente utilizado, también tiene algunas desventajas. Una gran desventaja es su complejidad computacional,

lo que puede hacer que sea lento para entrenar y difícil de implementar en dispositivos con recursos limitados. El componente RPN de Faster R-CNN puede ser particularmente costoso en términos de recursos computacionales, ya que implica ejecutar una red separada sobre toda la imagen (Figura 2.12) para generar propuestas de región.

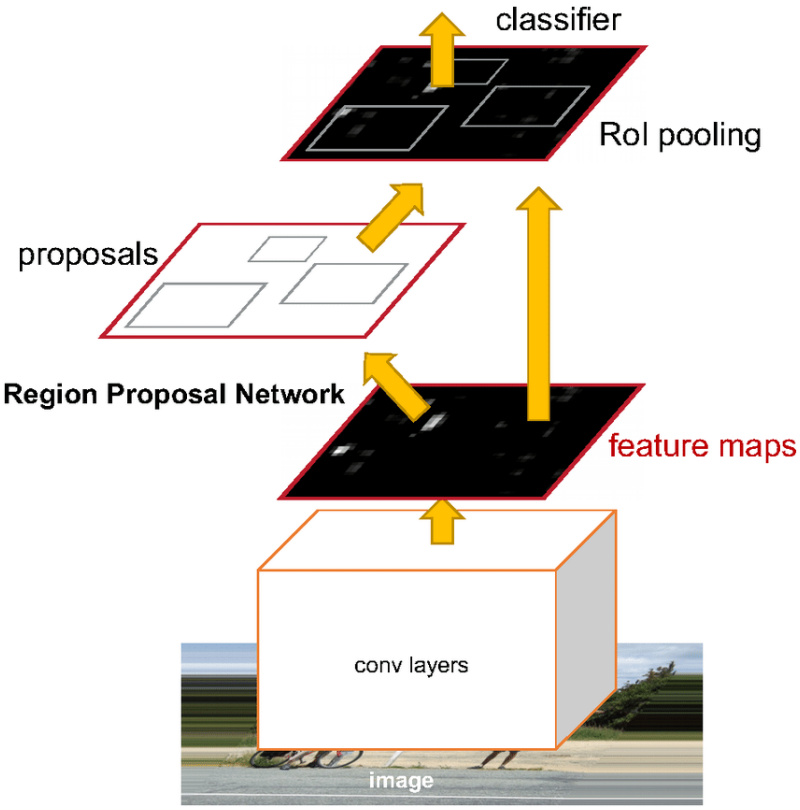


Figura 2.12: Arquitectura básica de una red feed-forward con dos capas ocultas [4]

2.4.2. YOLO

Es un modelo de detección de objetos en tiempo real que fue presentado en [5], este modelo es conocido por su velocidad y precisión, y se ha convertido en un modelo popular para la detección de objetos en tiempo real en una variedad de aplicaciones.

Como se mencionó con anterioridad una de sus principales ventajas es su velocidad, ya que puede procesar imágenes en tiempo real en una sola GPU. Además, es capaz de detectar múltiples objetos dentro de una sola imagen y clasificarlos en diferentes categorías. YOLO también tiene una arquitectura relativamente simple, lo que lo hace fácil de implementar y modificar.

Una desventaja de YOLO es que puede tener dificultades para detectar objetos pequeños, ya que utiliza un solo mapa de características para hacer predicciones en toda la imagen. Además, YOLO puede ser sensible al ruido de fondo y puede producir falsos positivos en escenas con mucho desorden. Finalmente, YOLO puede tener dificultades para detectar objetos que están parcialmente ocultos o tienen formas inusuales. Este modelo tuvo un impacto significativo en la comunidad de visión por computadora, ya que demostró el potencial para la detección de objetos en tiempo real en una variedad de aplicaciones además, este modelo inspiró el desarrollo de otros modelos basados en esta arquitectura (figura 2.13).

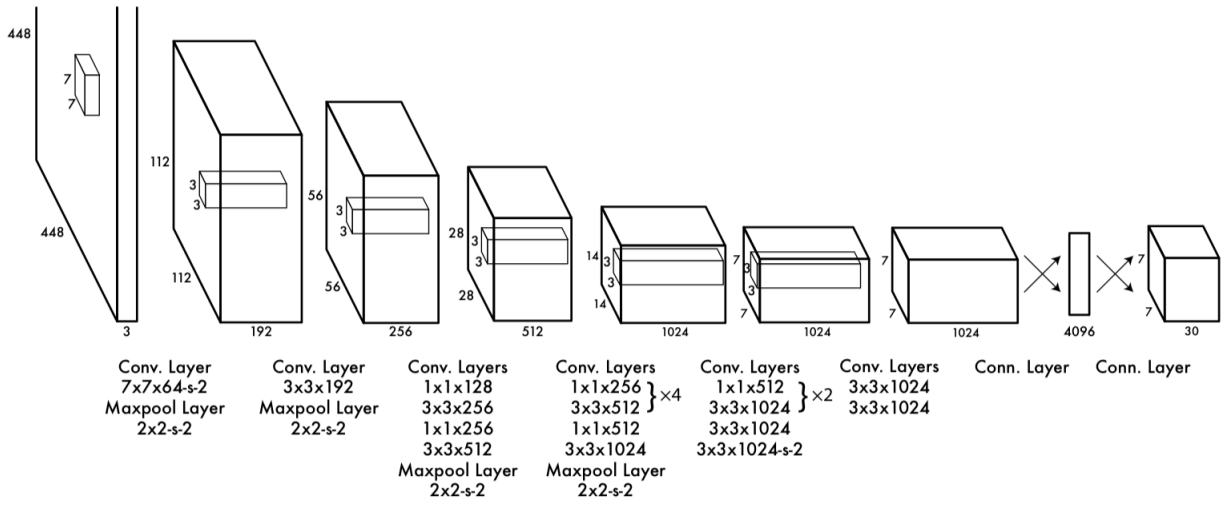


Figura 2.13: Arquitectura YOLO [5]

2.4.3. YOLOv8

YOLOv8 es la octava versión de YOLO creada por el equipo Ultralytics, fue publicada en su versión inicial el 13 de enero del 2023, tomando como base el trabajo inicial realizado en YOLO [6]. La principal ventaja de esta arquitectura es su alta precisión a comparación de versiones anteriores, especialmente en sus modelos menos complejos conocidos como Yolov8 Nano (Yolov8n) y Yolov8 Small (Yolov8s) y, ademas de tener una alta precisión, sus velocidades son mayores a los demás modelos (figura 2.14, $mAP50$ es explicado en la "Sección 4.2 - Métricas")

Además, a pesar de que Yolov8 cuenta con una arquitectura (figura 2.15) mas compleja que versiones anteriores, esta sigue siendo capaz de funcionar fácilmente en tiempo real en hardware no tan costoso.

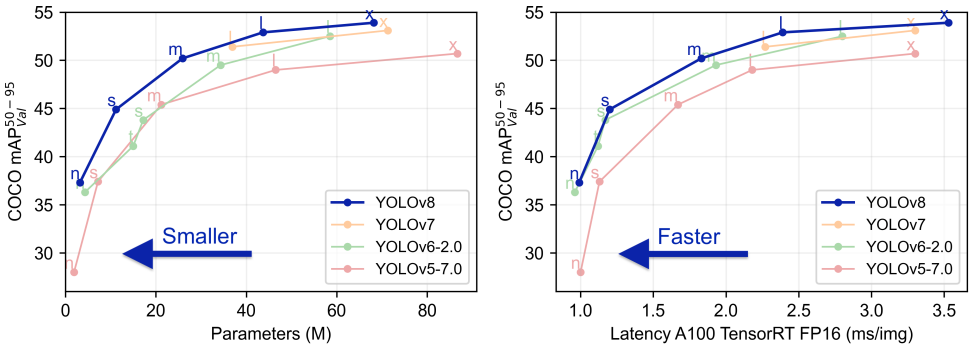


Figura 2.14: Gráficas comparando la precisión y la velocidad entre los diferentes modelos basados en YOLO [6]

La red (figura 2.15) utiliza una red de pirámide de características (FPN) para detectar objetos de diferentes tamaños y escalas dentro de una imagen. Esta FPN consta de múltiples capas que detectan objetos a diferentes escalas, lo que permite al modelo detectar objetos grandes y pequeños dentro de una imagen.

Ademas, la arquitectura se puede dividir en dos partes principales: "la columna vertebral (backbone) y la cabeza (head)", las cuales constan de diferentes submódulos los cuales son una combinación de capas convolucionales (Conv y C2f), capas de agrupamiento (MaxPool2d), capas de muestreo superior (Upsample), formulas de concatenación de tensores (Concat) y formulas de perdida (Bbox. Loss y Cls. Loss).

La columna vertebral consiste de una versión modificada de CSPDarknet53, consta de 53 capas convolucionales y emplea una técnica llamada "cross-stage partial connections"(conexiones parciales entre etapas) para mejorar el flujo de información entre las diferentes capas de la red.

La cabeza de YOLOv8 consta de múltiples capas de convolución seguidas de una serie de capas totalmente conectadas. Estas capas son responsables de predecir las cajas delimitadoras, las puntuaciones de objetividad y las probabilidades de clase para los objetos detectados en una imagen, además, una de las principales características de YOLOv8 es el uso de un mecanismo de autoatención en la cabeza de la red, este mecanismo permite al modelo centrarse en diferentes partes de la imagen y ajustar la importancia de diferentes características en función de su relevancia para la tarea.

Para finalizar, esta arquitectura permite utilizar un modelo pre-entrenado en el conjunto de datos COCO [26] el cual es un modelo general para detección de objetos, en este articulo se

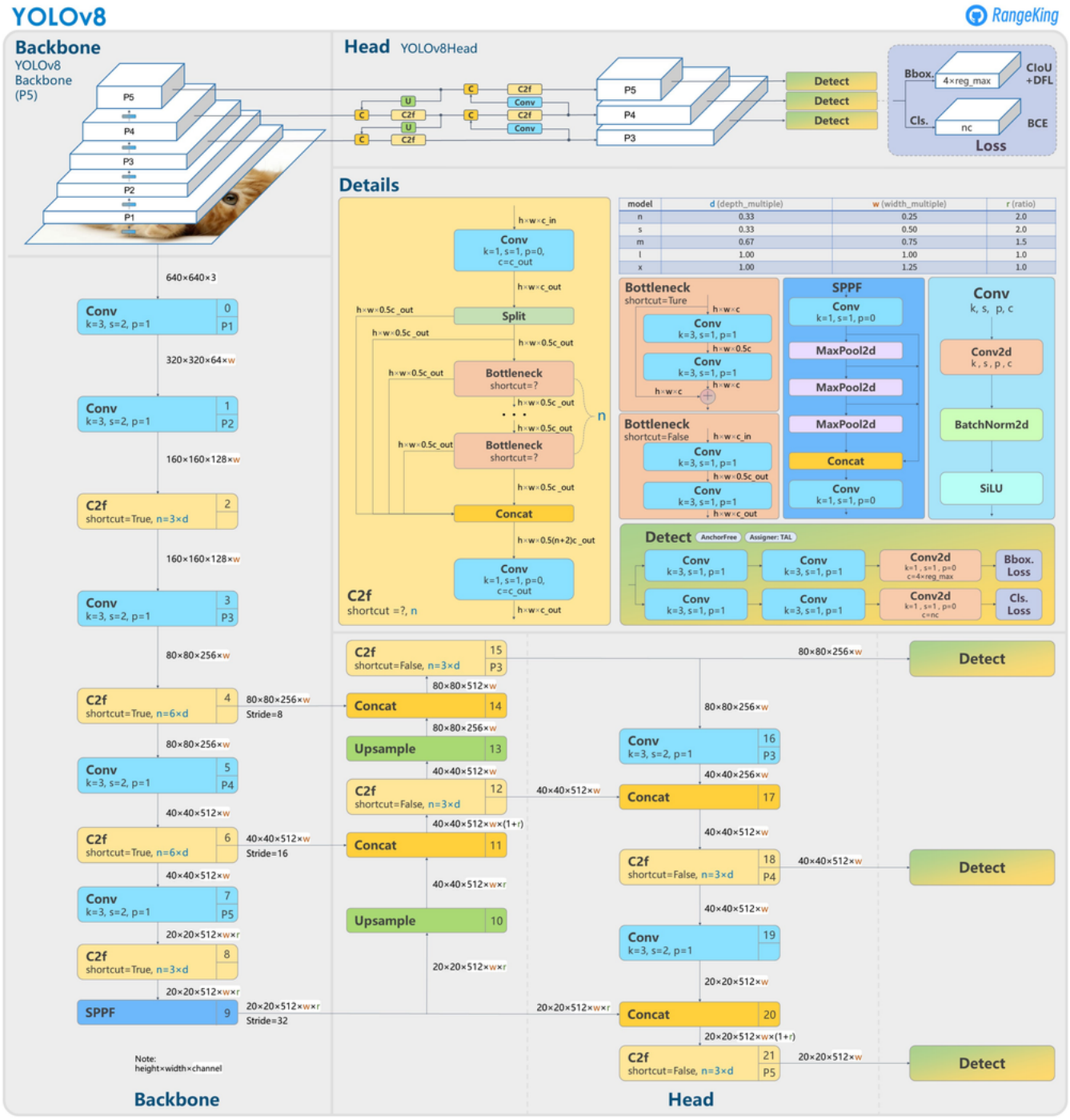


Figura 2.15: Arquitectura de la red YOLOv8 [7]

decidió utilizar dicho modelo pre-entrenado.

2.5. Estereoscopía

La estereoscopía es una técnica de visualización en la que se utilizan dos imágenes ligeramente diferentes para crear la ilusión de profundidad tridimensional. Esta técnica se basa en la percepción visual humana de la disparidad binocular, es decir, la diferencia en la posición de los dos ojos que permite al cerebro percibir la profundidad y la distancia de los objetos en el mundo real. En el caso de este estudio, se requiere que se recopilen imágenes con una cámara izquierda a cierta distancia y una cámara derecha en una distancia distinta.

2.5.1. Calibración de cámaras

La calibración y orientación de la cámara es un requisito previo necesario para la extracción de información métrica 3D precisa y fiable de las imágenes. Se considera que una cámara está calibrada si se conocen los parámetros de distancia principal, desplazamiento del punto principal y distorsión de la lente [27].

Para poder llevar a cabo dicha calibración necesitamos colocar ambas cámaras a la misma altura (figura 2.5.1), esto con la intención de que los cambios entre las imágenes sean únicamente visibles en el plano horizontal y que el error vertical sea el mínimo posible o inexistente.

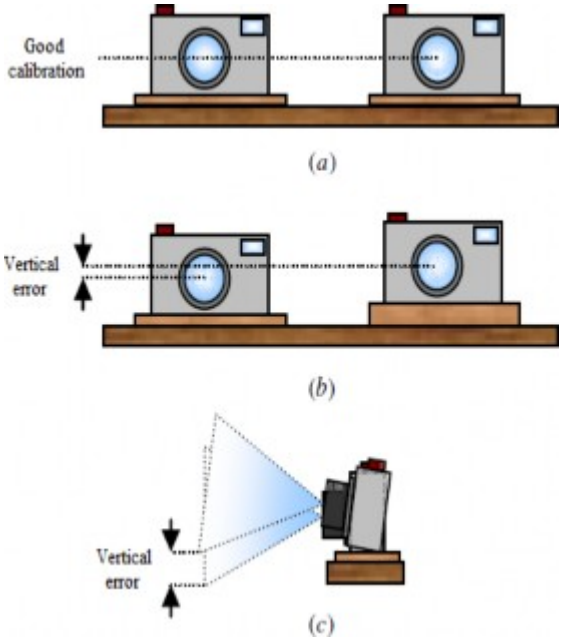


Figura 2.16: a) Posición de cámaras correcta. b) Posición incorrecta debido a la presencia de un error vertical c) Representación gráfica del error vertical existente [8]

El modelo de Zhang (figura 2.17) es un método de calibración de cámaras que usa técnicas de calibración tradicionales (conocidas como puntos de calibración) y técnicas de autocalibración (correspondencia entre los puntos de calibración cuando están en diferentes posiciones). Para realizar una calibración completa por este método son necesarias al menos tres imágenes diferentes del objetivo a calibrar, ya sea moviendo el objetivo o la misma cámara.

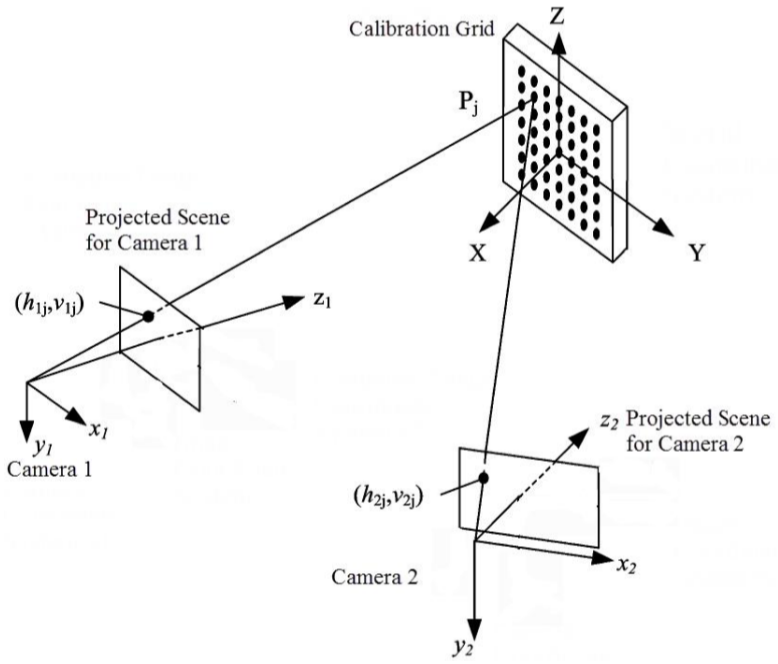


Figura 2.17: Esquema de calibración para un sistema de visión con 2 cámaras [9]

En la figura 2.17 se presentan los parámetros necesarios para llevar a cabo la calibración de las cámaras y estos son: X, Y, Z: Espacio de la malla de calibración. x_1 , y_1 , z_1 : Espacio de cámara 1 (o izquierda). x_2 , y_2 , z_2 : Espacio de cámara 2 (o derecha). h_{1j} , v_{1j} : Punto de la cámara 1 donde se proyecta la imagen de la escena. h_{2j} , v_{2j} : Punto de la cámara 2 donde se refleja la imagen de la escena. La principal aportación de este método es el extraer los parámetros intrínsecos K (distancia focal y el punto principal de ambas cámaras) ademas de los parámetros de calibración extrínsecos R y T. Los parámetros extrínsecos están relacionados con la posición 3D de las cámaras y expresan la transformación de un punto del mundo 3D a las coordenadas 2D de la cámara. Donde R denota la rotación y T una traducción, en la calibración estereoscópica, estos valores expresan la posición de una cámara (es decir, la derecha) en relación con la otra (es decir, la izquierda) [9].

Capítulo 3

Metodología

En este capítulo se presenta una metodología para la detección de armas tipo pistola y la estimación de la distancia a la que se encuentran. Esto es logrado aplicando la arquitectura YOLOv8 para la distinción de clases, por medio de aprendizaje profundo supervisado con CNNs. Considerando las referencias del estado del arte, los resultados obtenidos son positivos, y en algunos casos, superiores a los de aplicaciones afines. Además de obtener resultados prometedores en la detección, se presenta como aporte la estimación de la distancia a la que se encuentra la pistola detectada, ya que no se encontró literatura que presente un método similar al presentado.

El dataset (conjunto de imágenes originales con su respectiva imagen etiquetada o ground truth) utilizado para el entrenamiento fue obtenido de [14] y ampliado con imágenes originales por parte del equipo de trabajo. Estas imágenes fueron capturadas en el Instituto Tecnológico de La Paz, específicamente, en su laboratorio de sistemas computacionales. La utilidad de este dataset puede ser extendida para mejorar su precisión o incluso aumentar la cantidad de clases de detección.

3.1. Fases

La metodología llevada a la práctica se integró en las siguientes fases (ver Figura 3.1):

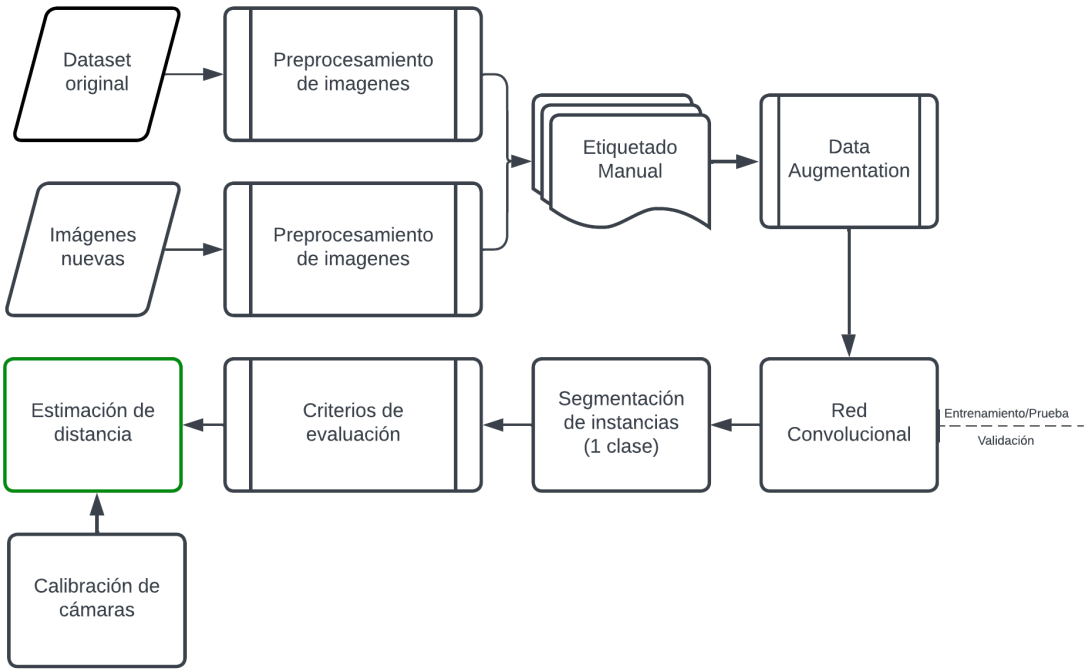


Figura 3.1: Fases de la metodología implementada

3.2. Software

Fueron utilizados principalmente 4 programas para el desarrollo de este estudio:

- Python: Todos los procedimientos metodológicos se llevaron a cabo utilizando este lenguaje de programación.
- PyTorch: es una librería de aprendizaje automático que proporciona una plataforma para construir y entrenar redes neuronales profundas. YOLOv8 fue aplicada mediante el uso de esta librería.
- OpenCV: es una librería diseñada para proporcionar una amplia gama de herramientas y algoritmos para el procesamiento de imágenes. Fue utilizada para desarrollar el algoritmo de estimación de distancia.
- Roboflow: es una plataforma basada en la nube para la gestión y preparación de conjuntos de datos de imágenes para su uso en aplicaciones de visión por computadora. La preparación del conjunto de datos fue llevado a cabo en esta plataforma.

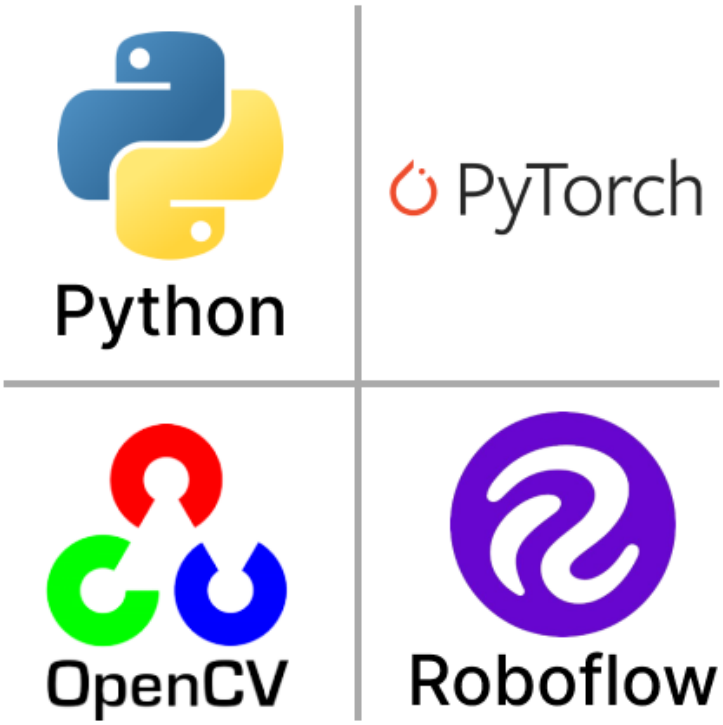


Figura 3.2: Software utilizado

3.3. Obtención de datos

Los datos se obtuvieron de [14], en donde se presentó un conjunto de datos que consta de 3000 imágenes de armas tipo pistola, algunas imágenes fueron capturadas por el equipo mientras que otras fueron recopiladas de redes sociales y resultados en metabuscadores. Además de estas imágenes, se capturaron 500 imágenes originales las cuales constan de personas con un arma tipo pistola en mano, así como imágenes del área de control en las que será probado el algoritmo sin armas de fuego presentes. Esto con la finalidad de que la CNN logre detectar cuando un arma no está presente, logrando así evitar falsos positivos.

3.4. Preprocesamiento y elaboración de dataset

El preprocesamiento del conjunto de imágenes consistió en:

- Cambio de formato de imagen: Se estableció el formato JPG, debido a que es uno de los formatos más ligeros aceptados por la red.

- Estandarización de tamaño a 640x640 pixeles.

Con la herramienta RoboFlow Annotate (https://roboflow.com) se creó el nuevo conjunto de datos que cuenta con imágenes originales, así ampliando la capacidad del dataset original (ver figura 3.3) conformando el limite de "*ground truth*"de forma manual con la única etiqueta de clase: **Pistol**.

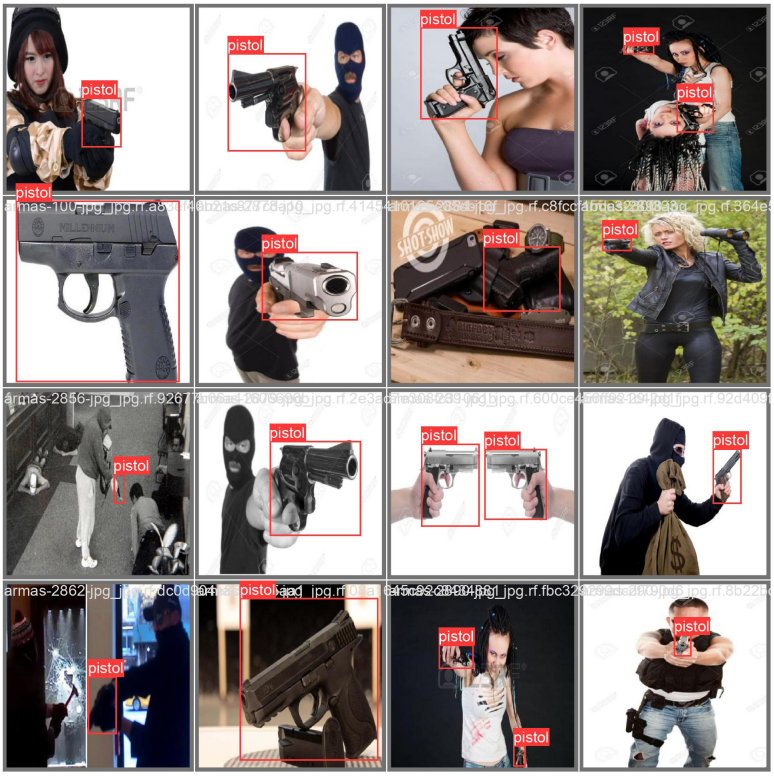


Figura 3.3: Imágenes presentes en el dataset original que fueron utilizadas en el entrenamiento, las etiquetas son consideradas ground truth

Las imágenes fueron separadas en diferentes categorías:

- Imágenes de entrenamiento
- Imágenes de validación
- Imágenes de prueba

Además, de dicha división las imágenes de entrenamiento fueron aumentadas con los siguientes métodos:

- Rotación de la imagen entre -15 a $+15$ grados

- Voltear las imágenes horizontal y verticalmente
- Conversión a escala de grises
- Aumento de brillo entre $-20\,\%$ a $+20\,\%$

El aumento de datos se realizó con el objetivo de aumentar el tamaño de los conjuntos de datos y ayudar a evitar el sobreajuste durante el entrenamiento del modelo (overfitting). Al aumentar el tamaño del conjunto de datos, la red se expone a una mayor variedad de datos, lo que le ayuda a aprender características más robustas y hacer predicciones más precisas. También ayuda a reducir la sensibilidad a pequeñas variaciones en los datos mediante la introducción de transformaciones generadas aleatoriamente, como rotación y escalado, para aumentar el conjunto de datos existente. De esta manera, el modelo de CNN es capaz de generalizarse mejor ya que se entrena sobre una gama más amplia de puntos de datos. Tras realizar el aumento mencionado, el entrenamiento se realizó con 7800 imágenes y la validación con 741 imágenes.

3.5. Algoritmo de estimación de distancia

Una vez que el algoritmo YOLO detecta un objeto, este mismo proporciona las coordenadas de los contenedores de fronteras de todos los objetos detectados. Estas coordenadas son presentadas como los parámetros:

- X_1
- X_2
- Y_1
- Y_2

los cuales son utilizados en la expresión matemática para la estimación de distancia basada en estereoscopía (figura 3.4).

En la figura 3.4 se muestran los parámetros necesarios para la estimación de distancia a la que se encuentra el objeto detectado:

- S_L y S_R representan a la cámara izquierda y derecha respectivamente.

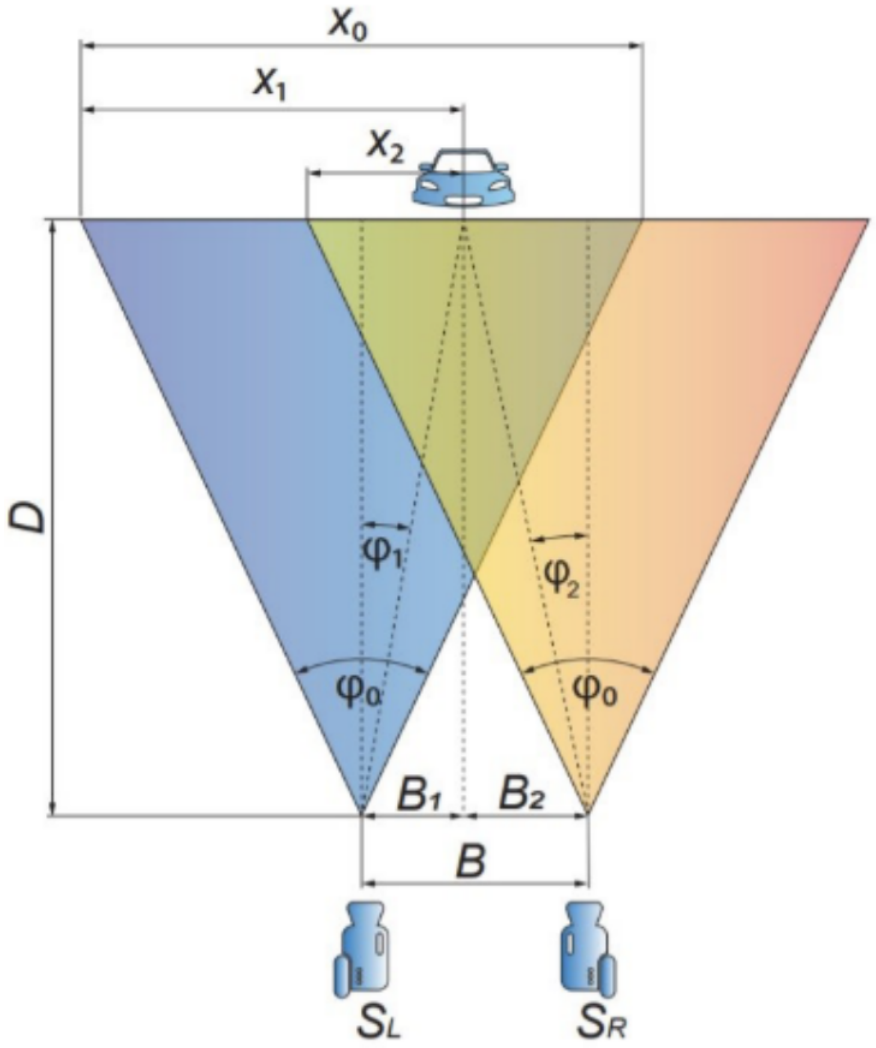


Figura 3.4: Parámetros importantes para la obtención de distancia mediante estereoscopía [8]

- B es la distancia a la que se encuentran las cámaras entre ellas.
- φ_0 representa el campo de visión (field of view, FoV) de las cámaras (este valor cambia de acuerdo a cada cámara, y se puede obtener en sus especificaciones técnicas).
- D es la distancia existente entre las cámaras y el objeto detectado.

La ecuación para obtener la distancia D se representa de la siguiente manera:

$$D = \frac{B}{tan(\varphi_1) + tan(\varphi_2)} \tag{3.1}$$

En donde φ_1 y φ_2 son los ángulos entre el eje del lente de la cámara y la dirección al objeto detectado. Otros tres párametros importantes son:

- X_1 representa el número de pixeles horizontales de las imágenes
- X_1 (imagen izquierda) y X_2 (imagen derecha) son el número de píxeles entre el punto medio del borde horizontal del cuadro delimitador de los objetos y el borde izquierdo de la imagen.

$$D = \frac{B \times X_0}{2\tan(\frac{\varphi_0}{2})(X_1 - X_2)} \tag{3.2}$$

Con la ecuación 3.2 podemos estimar la distancia de cualquier objeto que esté presente en ambas imágenes. El algoritmo para la obtención de la distancia consta de los siguientes criterios

1. En ambas detecciones el objeto debe pertenecer a la misma clase (pistola).
2. El objeto en la imagen de la cámara derecha debe estar mas cerca de la orilla izquierda a comparación del mismo objeto en la imagen izquierda (ecuación 3.3).

$$X_i + W_i < (X_d + W_d)\ y\ (X_i < X_d) \tag{3.3}$$

3. El tamaño de los cuadros delimitadores deben ser lo más parecidos posibles de acuerdo a un umbral estimado. Utilizando la (ecuación 3.4) donde W_i y W_d son el ancho del cuadro delimitador izquierdo y derecho respectivamente, y, H_i y H_d son la altura del cuadro delimitador izquierdo y derecho respectivamente. La obtención del valor final del umbral se llevó a cabo variando el valor de este mismo en diferentes pruebas. El valor inicial fue de 0.75 y este mismo fue variando en cada prueba hasta 0.95 en incrementos de 0.05 siendo el valor 0.90 el que brindo mejores resultados.

$$(\frac{min(W_i, W_d)}{max(W_i, W_d)} > 0.9)\ y\ (\frac{min(H_i, H_d)}{max(H_i, H_d)} > 0.9) \tag{3.4}$$

4. La diferencia entre los centroides en el plano vertical (y) debe ser lo mas pequeña posible, de acuerdo a un umbral definido (formula 3.5). Dicho umbral también fue obtenido mediante la variación de su valor en múltiples pruebas. El valor inicial fue de 1 pixel y este mismo fue variando en cada prueba hasta 10 pixeles en incrementos de 1 pixel siendo el valor de 5 pixeles el que brindo mejores resultados.

$$|(Cy_i - Cy_d)| < 5\ pixeles \tag{3.5}$$

Estos criterios son utilizados para verificar que el objeto que está siendo detectado en la cámara izquierda es el mismo objeto que está siendo detectado en la cámara derecha, en dado caso de que se cumplan todos los criterios, el algoritmo llevara a cabo su ultimo paso, el calcular la distancia mediante el uso de la ecuación 3.2.

Capítulo 4

Resultados

4.1. Entrenamiento

El algoritmo optimizador utilizado durante el entrenamiento fue el descenso de gradiente estocástico (SGD por sus siglas en ingles); con un factor de aprendizaje (learning rate) de 0.01; el valor IoU predeterminado es de 0.6; 100 epocas de entrenamiento y un tamaño de lote (batch size) de 64. Los procesos fueron codificados en Python. Para la fase de entrenamiento fue utilizado un equipo con procesador Intel Core i7 9750H, RAM 16GB, así como una Unidad Gráfica de Procesamiento (GPU, por sus siglas en inglés) en tarjeta Nvidia TITAN X. El entrenamiento tomó aproximadamente 4 horas, los resultados que se muestran en la figura 4.1 fueron obtenidos durante el entrenamiento con el uso del conjunto de validación, el cual consta de 741 imágenes que nunca han sido, ni serán, presentadas al modelo.

Las imágenes seleccionadas fueron divididas de la siguiente forma: 70 % para entrenamiento, 20 % para prueba y 10 % para validación.

4.2. Métricas

Como se ilustró en la figura 2.14, COCO es un conjunto de datos de subtítulos, segmentación y detección de objetos a gran escala. COCO mAP (mean average precision por sus siglas en inglés) toma en cuenta la precisión y el recordatorio de detección de objetivos en dicho conjunto de datos. Cuando hablamos de $mAP50-95$ debemos tomar en cuenta diferentes umbrales de IoU (Intersección sobre Unión, o Intersección sobre Unión en español). IoU es un valor entre

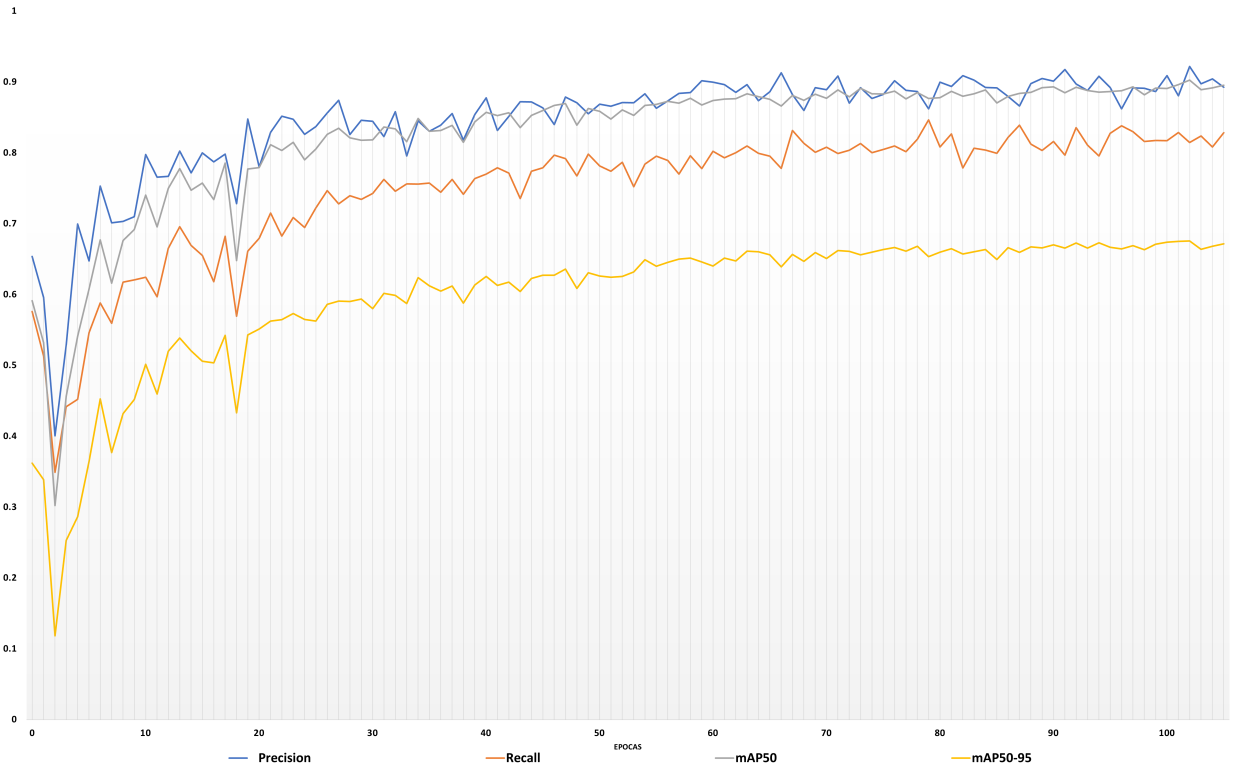


Figura 4.1: Resultados del entrenamiento

0 a 1 que cuantifica el grado de superposición entre dos cajas delimitadoras (Figura 4.2). El área de superposición es el área donde se superponen la caja delimitadora de verdad básica (o ground truth en inglés) y la caja delimitadora predicha por el modelo y el área de unión toma en cuenta el área total cubierta entre ambas cajas delimitadoras.

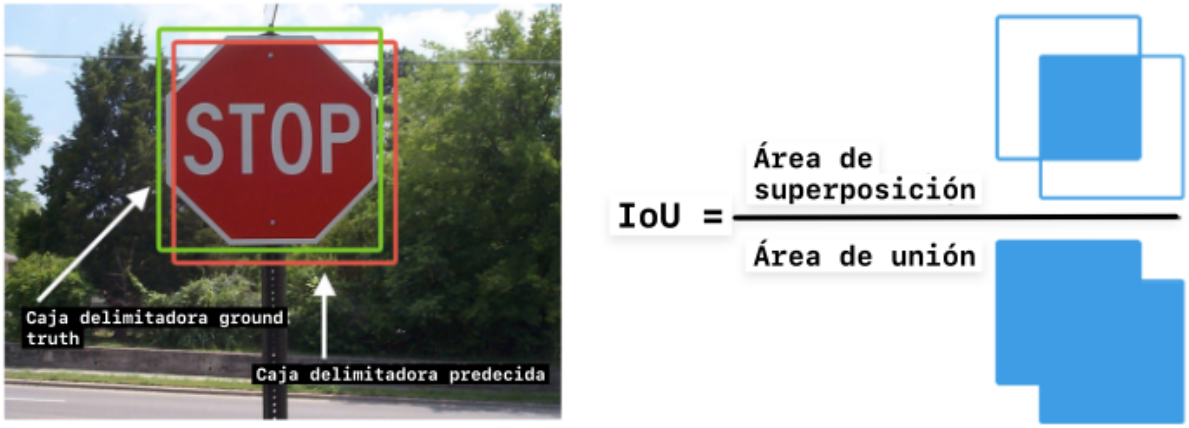


Figura 4.2: Representación visual de los parámetros necesitados para obtener la *IoU*

Además de la IoU, es necesario obtener el valor de precisión (ecuación 4.1), el cual es el por-

centaje de predicciones correctas realizadas por el modelo entrenado. En términos más simples, la precisión es la relación entre los verdaderos positivos y todos los positivos. Cabe mencionar que este valor es dependiente del umbral de IoU especificado, lo que significa que este puede mejorar o empeorar de acuerdo a dicho umbral. Por ejemplo, en nuestro campo de estudio, esta métrica nos estaría diciendo "qué tan correcto es nuestro modelo cuando este mismo detecta que hay un arma en la imagen". Los **verdaderos positivos** son definidos como una detección correcta de una caja delimitadora ground truth, en nuestro caso es "cuando una pistola se encuentra en la imagen y nuestro modelo la detecto" mientras que los falsos positivos son una detección incorrecta de un objeto inexistente o una detección fuera de un objeto existente, en nuestro caso sería "cuando en la imagen no se encuentra una pistola pero el modelo detecto una". Los falsos positivos son considerados como un error tipo I, esto debido a que a pesar de que es un error en la detección este no tiene repercusiones tan catastróficas como el caso de un falso negativo, por ejemplo, si nuestro modelo detecta una pistola cuando no hay una pistola presente simplemente podemos anular esa detección.

$$Precisión = \frac{Verdaderos\ Positivos}{Verdaderos\ Positivos + Falsos\ Positivos} \tag{4.1}$$

También necesitamos obtener el valor conocido como **exhaustividad** (recall) (ecuación 4.2). Este valor, que igualmente es un porcentaje (al multiplicarse por 100), define qué tan bueno es nuestro modelo para identificar verdaderos positivos correctamente. Igual que la precisión este valor es dependiente del umbral de IoU especificado. Entonces, en nuestro campo de estudio nos diría "de todas las imágenes que cuentan con una pistola cuantas de ellas detectamos correctamente como una pistola". Los falsos negativos son definidos como una caja delimitadora ground truth no detectada, en nuestro caso sería "la imagen cuenta con una pistola pero nuestro modelo detecto que no hay ninguna pistola en ella". Además, los falsos negativos son errores de alta importancia, por ejemplo, si nuestro modelo no detecta una pistola en la escena cuando si hay una pistola esto puede facilitar un tiroteo o un asesinato.

$$Recall = \frac{Verdaderos\ Positivos}{Verdaderos\ Positivos + Falsos\ Negativos} \tag{4.2}$$

El uso de precisión (ecuación 4.1) y de recall (ecuación 4.2) puede parecer confuso, pero lo podemos presentar de esta manera: "la precisión se centra en qué tan correctas son las detecciones, mientras que el recall se centra en la completitud de las detecciones". Un sistema

Tabla 4.1: Métricas obtenidas tras evaluar el modelo entrenado

Imagenes	Instancias	Precisión	IoU	Recall	mAP50.
741	783	0.922	0.6	0.815	0.903

con alta precisión puede perder algunos objetos, mientras que un sistema con alto recall puede tener muchos falsos positivos. Equilibrar estas dos métricas es crucial para lograr un buen rendimiento general en un sistema de detección de objetos. mAP50 (ecuación 4.3), se calcula obteniendo al valor promedio de precisión (ecuación 4.1) para cada clase y luego obteniendo la media de la precisión en todas las clases cuando el umbral IoU (figura 4.2) es igual a 0.50, $mAP50-95$ se obtiene de la misma manera con el único cambio siendo que cuenta con múltiples umbrales IoU desde 0.50 hasta 0.95, con valores tomados cada 0.05, dando un total de 10 valores, donde luego se toma el valor promedio. Cuanto mayor sea el indicador, mejor será la detección del modelo [28].

$$\begin{aligned} mAP &= \frac{1}{n}\sum_{k=1}^{k=n} AP_k \\ AP_k &= Precisión\ promedio\ de\ la\ clase\ k \\ n &= el\ número\ de\ clases \end{aligned} \tag{4.3}$$

Las métricas presentadas durante el entrenamiento son las mismas que se presentaron en las ecuaciones (4.1), (4.2), así como $mAP50$ y $mAP50-95$. Se guardaron los pesos que brindaron los mejores resultados en el set de evaluación y estos mismos fueron utilizados en los experimentos en tiempo real. Los resultados brindados por estos pesos fueron los siguientes (tabla 4.1):

4.2.1. Pruebas de reconocimiento

En la figura 4.3 se incluyen algunos ejemplos de los resultados obtenidos al evaluar el modelo con imágenes presentes en el conjunto de evaluación, en este caso únicamente se esta presentando la detección de la pistola, no se está intentando obtener la distancia a la que se encuentra el objeto.

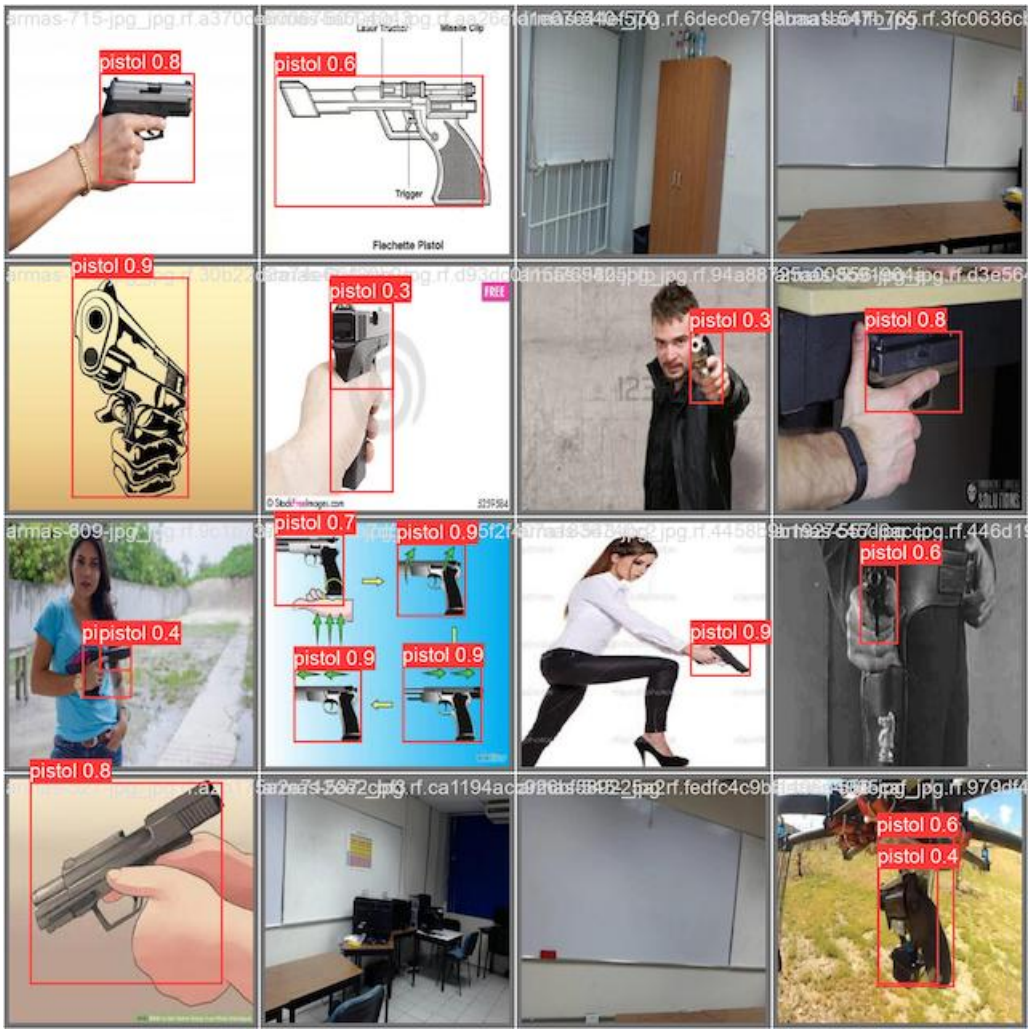


Figura 4.3: Predicciones realizadas por el modelo, los valores encima de la caja representan la confianza del modelo

4.2.2. Resultados de detección del arma y estimación de distancia

Las pruebas de reconocimiento en tiempo real fueron llevadas en un área controlada en el Instituto Tecnológico de La Paz, esto con la finalidad de experimentar a diferentes distancias sin distracciones por parte del mundo exterior, ademas, la confianza mínima del modelo se colocó en un 0.60 puesto que se consideró que esta confianza es lo suficientemente elevada como para eliminar falsos negativos durante el funcionamiento. Las cámaras utilizadas durante los experimentos son del modelo *"Logitech C920 Pro HD Webcam, 1080p "*.

En este punto se está evaluando que tan viable es utilizar este modelo para la detección de armas tipo pistola en tiempo real además de definir el tamaño del error respecto a la obtención

de distancia.

Los resultados de la estimación de distancia se presentan en la tabla 4.2 y en la figura 4.5, además mostramos un ejemplo de la detección y estimación en la figura 4.4.

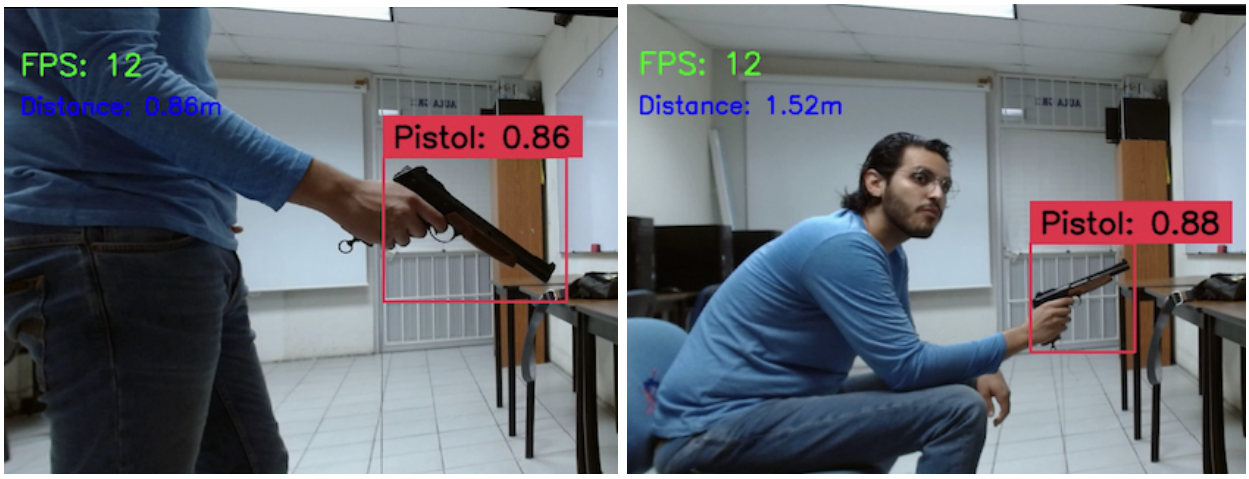


Figura 4.4: Imágenes de los experimentos llevados a cabo.

Como se logra ver en la figura 4.4 el modelo corre a 12 cuadros por segundo de manera constante los cuales son más que suficientes para un funcionamiento en tiempo real, además podemos apreciar como el modelo detecta el objeto con diferentes niveles de confianza de acuerdo a la cámara que lo esté detectando.

4.3. Conclusiones

El uso de las redes convolucionales aplicadas a sistemas de videovigilancia constituyen una alternativa viable para la detección de armas con alta precisión, especialmente utilizando un algoritmo tipo YOLO ya que esté facilitó el funcionamiento de este sistema en tiempo real.

Además, no solamente se logró el uso del sistema en tiempo real, si no que se logró superar la precisión de múltiples trabajos en el estado del arte. La precisión final obtenida en este trabaho es de 92.2 % (tabla 4.2) ha demostrado ser mas elevada que lo logrado con arquitecturas tipo FASTER R-CNN y VGG16 SSD comó se presenta en [14] y [28] donde se alcanzó una precisión máxima de 84.21 % y 84.6 % respectivamente. El algoritmo de estimación de distancia demostró ser lo suficientemente funcional hasta una distancia de 3.1 metros, manejando un error promedio de 0.093 metros o 9.3 centímetros. Sin embargo, dicho algoritmo se encuentra

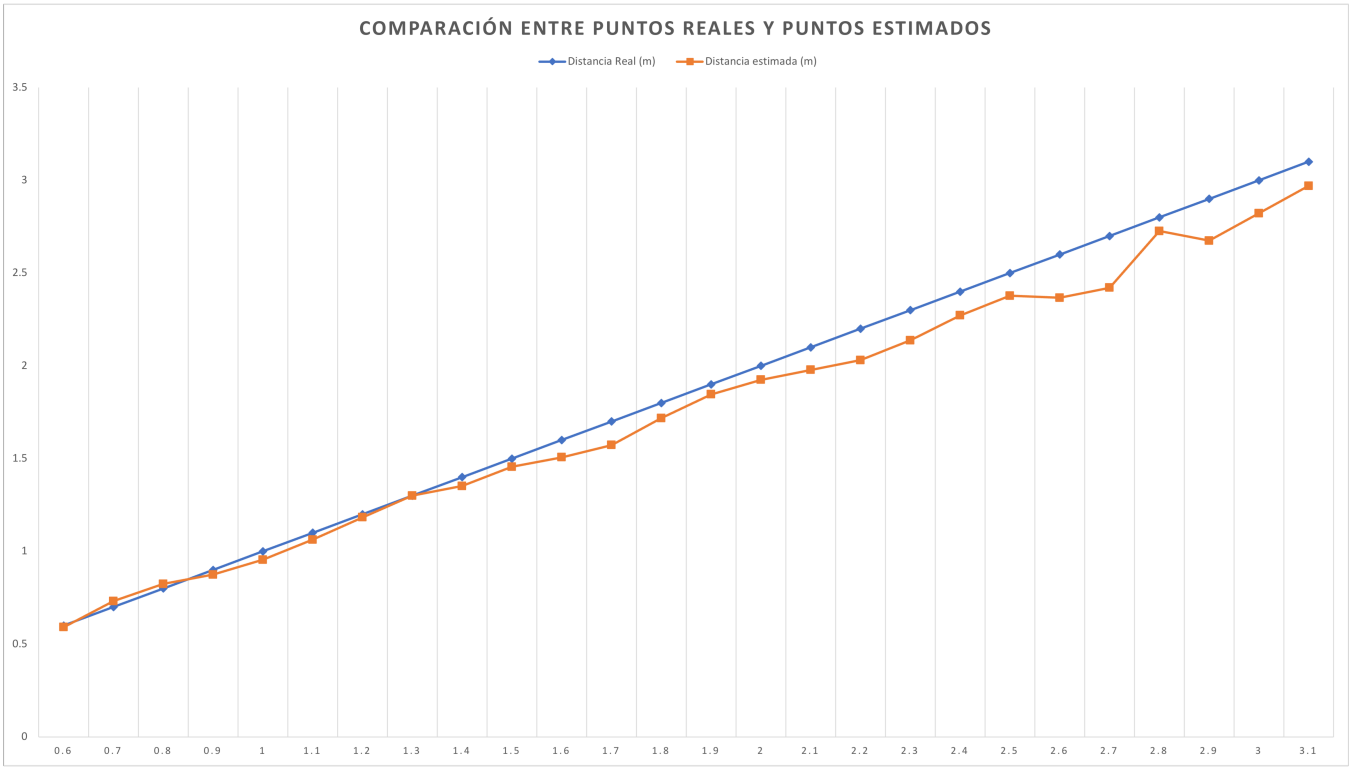


Figura 4.5: Comparación gráfica entre la distancia real y la distancia estimada

limitado por la detección precisa del arma, la cual demostró tener problemas en la detección del objeto a distancias mayores a 3 metros, ya que este mismo si logra detectar el objeto pero con una confianza demasiado baja (menor a 0.5). Como consideraciones futuras, se debe ampliar el algoritmo de estimación de distancia mediante el uso de un sistema de reconocimiento de esquinas, con la intención de no solo lograr la estimación de la distancia del objeto si no que también estimar el tamaño real del objeto. Esto último con la intención de lograr eliminar detecciones que se consideren demasiado grandes como para realmente ser un arma tipo pistola.

Tabla 4.2: Resultado de las pruebas de estimación de distancia

Distancia Real (m)	**Distancia Estimada (m)**	**Error (m)**
0.6	0.591	0.009
0.7	0.732	-0.032
0.8	0.824	-0.024
0.9	0.875	0.025
1.0	0.954	0.046
1.1	1.063	0.037
1.2	1.183	0.017
1.3	1.301	-0.001
1.4	1.352	0.048
1.5	1.456	0.044
1.6	1.507	0.093
1.7	1.573	0.127
1.8	1.718	0.082
1.9	1.846	0.054
2.0	1.925	0.075
2.1	1.978	0.122
2.2	1.978	0.170
2.3	2.138	0.162
2.4	2.272	0.128
2.5	2.378	0.122
2.6	2.367	0.233
2.7	2.421	0.279
2.8	2.726	0.074
2.9	2.675	0.225
3.0	2.822	0.0178
3.1	2.97	0.13
	Error promedio (m)	**.093**

Bibliografía

[1] Abdul Mueed Hafiz and Ghulam Mohiuddin Bhat. A survey on instance segmentation: state of the art. *International journal of multimedia information retrieval*, 9(3):171–189, 2020.

[2] Qaisar Abbas, Mostafa EA Ibrahim, and M Arfan Jaffar. A comprehensive review of recent advances on deep vision systems. *Artificial Intelligence Review*, 52(1):39–76, 2019.

[3] Rikiya Yamashita, Mizuho Nishio, Richard Kinh Gian Do, and Kaori Togashi. Convolutional neural networks: an overview and application in radiology. *Insights into imaging*, 9:611–629, 2018.

[4] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.

[5] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[6] Glenn Jocher and Ultralytics. Github yolov8 - ultralytics, 1 2023. Accessed on February 13, 2023.

[7] RangeKing. Brief summary of yolov8 model structure, 1 2023. Accessed on February 13, 2023.

[8] Bojan Strbac, Marko Gostovic, Zeljko Lukac, and Dragan Samardzija. Yolo multi-camera object detection and distance estimation. *2020 Zooming Innovation in Consumer Technologies Conference, ZINC 2020*, pages 26–30, 5 2020.

[9] Saul Martínez Chavelas and Saul Martinez Diaz. Algoritmo paralelo en gpu para el rastreo de armas de fuego tipo pistola tesis. 2019.

[10] INEGI. Datos preliminares revelan que en 2020 se registraron 36579 homicidios, 7 2021.

[11] ENVIPE. Encuesta nacional de victimizació y percepción sobre seguridad (envipe) 2020, 7 2020.

[12] Santiago Gómez, Daniel Mejía, and Santiago Tobón. The deterrent effect of surveillance cameras on crime. *Journal of policy analysis and management*, 40(2):553–571, 2021.

[13] M Grega, A Matiolanski, P Guzik, and M Leszczuk. Automated detection of firearms and knives in a cctv image. 2016.

[14] Roberto Olmos, Siham Tabik, and Francisco Herrera. Automatic handgun detection alarm in videos using deep learning. *Neurocomputing*, 275:66–72, 1 2018.

[15] Rohit Kumar Tiwari and Gyanendra K Verma. A computer vision based framework for visual gun detection using harris interest point detector. *Procedia Computer Science*, 54:703–712, 2015.

[16] Shenghao Xu and Kevin Hung. Development of an ai-based system for automatic detection and recognition of weapons in surveillance videos. In *2020 IEEE 10th Symposium on Computer Applications & Industrial Electronics (ISCAIE)*, pages 48–52. IEEE, 2020.

[17] Jesus Salido, Vanesa Lomas, Jesus Ruiz-Santaquiteria, and Oscar Deniz. Automatic handgun detection with deep learning in video surveillance images. *Applied Sciences*, 11(13):6085, 2021.

[18] Tufail Sajjad Shah Hashmi, Nazeef Ul Haq, Muhammad Moazam Fraz, and Muhammad Shahzad. Application of deep learning for weapons detection in surveillance videos. In *2021 international conference on digital futures and transformative technologies (ICoDT2)*, pages 1–6. IEEE, 2021.

[19] Jacquelina J. Padilla Arballo, Saul Martinez Diaz, Marco A. Castro Liera, and Jorge E. Luna Taylor. Deteccion de cambio en superficie costera mediante la segmentacion de imagenes

aereas utilizando redes neuronales convolucionales. *Padi Boletin Cientifico de Ciencias Basicas e Ingenierias del ICBI*, 10:136–144, 10 2022.

[20] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[21] Yi Li, Haozhi Qi, Jifeng Dai, Xiangyang Ji, and Yichen Wei. Fully convolutional instance-aware semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2359–2367, 2017.

[22] Sergey Zagoruyko, Adam Lerer, Tsung-Yi Lin, Pedro O Pinheiro, Sam Gross, Soumith Chintala, and Piotr Dollár. A multipath network for object detection. *arXiv preprint arXiv:1604.02135*, 2016.

[23] Daniel Graupe. *Principles of artificial neural networks*, volume 7. World Scientific, 2013.

[24] Marvin Minsky and Seymour Papert. An introduction to computational geometry. *Cambridge tiass., HIT*, 479(480):104, 1969.

[25] C Aggarwal Charu. *Neural networks and deep learning: a textbook*. Spinger, 2018.

[26] Andreas Veit, Tomas Matera, Lukas Neumann, Jiri Matas, and Serge Belongie. Coco-text: Dataset and benchmark for text detection and recognition in natural images. *arXiv preprint arXiv:1601.07140*, 2016.

[27] Fabio ; Remondino, Clive Fraser, and Fabio Remondino. Digital camera calibration methods. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XXXVI:266–272, 2006.

[28] DURSHETI SUSMITHA and S VIJAY KUMAR. Weapon detection using artificial intelligence and deep learning for security applications. *Journal of Engineering Sciences*, 14(01), 2023.