Convolutional Neural Network Model for Weapon Identification

Gary Reyes[1,2],Kevin Intriago Narváez[2]
Katherine Andrea Pacheco Lino[2]
[1]Carrera de Sistemas Inteligentes, Universidad Bolivariana del Ecuador, Campus Durán km 5.5 vía Durán Yaguachi, Durán 092405, Ecuador gxreyesz@ube.edu.ec
[2]Facultad de Ciencias Matemáticas y Físicas, Universidad de Guayaquil, Cdla. Universitaria Salvador Allende, Guayaquil 090514, Ecuador, gary.reyesz@ug.edu.ec, kevin.intriagon@ug.edu.ec, katherinepachecol@ug.edu.ec

***Abstract* - This project article a solution to problems with insecurity by detecting the misuse of weapons in public places, it seeks to meet the objectives of creating its own dataset to carry out training and thus be able to evaluate accuracy, precision, sensitivity and F1 Score, validating using different data sets and with these metrics the performance of the model. For the development of the project, the literature related to convolutional neural networks and weapon identification was first investigated, and cameras that allow GPS location to be extracted were also investigated. Then, a dataset made up of network images and own images taken was created. In a real scenario, the model was trained in Google Colab to identify 6 different types of weapons, including firearms and edged weapons. For this, a Yolov8 model was used, which is ideal for detection, classification and segmentation tasks. A program was created in Visual Studio Code to carry out the tests. After the tests were completed, the confusion matrices were created and the different metrics were calculated to evaluate the performance of the model in the different classes. The conclusion was reached after the analysis of the confusion matrices that the model has a medium performance when detecting images of fire (rifles, shotguns, pistols), however it has a low performance when identifying edged weapons (knives, scissors, machetes) especially in knives that are confused with other classes, especially with machetes.**

***Key words:* Convolutional neural network, weapons identification, GPS, Yolov8, Deep Learning, Machine Learning.**

Modelo de Red Neuronal Convolucional para identificación de armas

Gary Reyes[1,2],Kevin Intriago Narváez[2]
Katherine Andrea Pacheco Lino[2]
[1]Carrera de Sistemas Inteligentes, Universidad Bolivariana del Ecuador, Campus Durán km 5.5 vía Durán Yaguachi, Durán 092405, Ecuador gxreyesz@ube.edu.ec
[2]Facultad de Ciencias Matemáticas y Físicas, Universidad de Guayaquil, Cdla. Universitaria Salvador Allende, Guayaquil 090514, Ecuador, gary.reyesz@ug.edu.ec, kevin.intriagon@ug.edu.ec, katherinepachecol@ug.edu.ec

El presente articulo aborda una solución a problemas con la inseguridad detectando el uso indebido de armas en lugares públicos, persigue cumplir con los objetivos de crear un conjunto de datos propio para realizar un entrenamiento y así poder evaluar los indicadores de exactitud, precisión, sensibilidad y F1 Score, validando mediante diferentes conjuntos de datos y con estas métricas el rendimiento del modelo. Para el desarrollo del proyecto primero se investigó la literatura relacionada con las redes neuronales convolucionales y la identificación de armas, además se investigó sobre cámaras que permitan extraer la ubicación GPS, después se creó un conjunto de datos propio conformado por imágenes de la red e imágenes propias tomadas en un escenario real, se entrenó el modelo en Google Colab para que identifique 6 tipos de armas distintas, entre armas de fuego y armas blancas, para esto se usó un modelo de Yolov8 que es ideal para tareas de detección, clasificación y segmentación. Se realizo un programa en Visual Studio Code para poder realizar las pruebas, después de realizadas las pruebas se elaboró las matrices de confusión y se calculó las distintas métricas para evaluar el rendimiento del modelo en las distintas clases. Se llego a la conclusión luego del análisis de las matrices de confusión de que el modelo tiene un rendimiento medio al momento de detectar imágenes de fuego (fusiles, escopetas, pistolas), en cambio tiene un rendimiento bajo al momento de identificar armas blancas (cuchillos, tijeras, machetes) especialmente en los cuchillos que confunde con otras clases sobre todo con machetes.

Palabras clave: Red neuronal convolucional, identificación armas, GPS, Yolov8, Deep Learning, Machine Learning

1. INTRODUCCIÓN

Las redes neuronales convolucionales buscan simular el funcionamiento del cerebro humano para esto tienen múltiples capas y al pasar por cada una es que se realiza la identificación del objeto. Este tipo de algoritmos actualmente son muy usados para identificar todo tipo de objetos y en distintos campos, hasta en el campo de la medicina para identificar tumores. En este caso se usará en el ámbito de la seguridad entrenando un modelo para que reconozca armas en lugares públicos. Investigando trabajos previos relacionados con la temática, se encontró un modelo eficiente para este tipo de tareas y Google Colab para realizar el entrenamiento usando la T4 GPU que nos ofrece mejores características de hardware, ya que este tipo de tareas suelen demandar muchos recursos. Se busca entrenar una red para mejorar la exactitud, precisión, sensibilidad y f1 score al momento de realizar las detecciones y tener un mejor rendimiento del modelo.

A continuación, una breve descripción de lo que se tendrá en cada sección desarrollada:

En la sección de Materiales y Métodos, se detalla la literatura relacionada a la problemática, las herramientas que se usaron para el desarrollado del mismo, el proceso de desarrollo y como se medirá el rendimiento del algoritmo de identificación de armas.

Se encuentra el funcionamiento del programa en el que realizaremos las pruebas.

En la sección de resultados se muestra el rendimiento del algoritmo de red neuronal, a través de las matrices de confusión, mediante el cálculo de las matrices tanto por clase como de forma global y así poder tener una mejor visión del funcionamiento del algoritmo.

Por último, en la sección 4 de las conclusiones, se tiene las conclusiones del presente estudio.

2. TRABAJOS RELACIONADOS

En el presente artículo se estudiará la identificación de objetos usando algoritmos de Inteligencia Artificial. Una red neuronal convolucional es una arquitectura de red del Deep Learning el cual forma parte del Machine Learning, la CNN está formado por varias capas, las cuales aprenden distintas características de las imágenes a estudiar.

Como se indica en [1], Los sistemas de detección de armas son un desafío dado que requieren la detección con una alta precisión logrando reducir la tasa de falsos positivos y falsos negativos. Además, debido a la naturaleza crítica de la detección se requiere que el tiempo de procesamiento y respuesta sea rápido.

En [2], Las redes neuronales convolucionales han proporcionado un gran avance en visión artificial en los últimos años, en tareas como la detección de objetos en tiempo real y en clasificación de imágenes, siendo utilizada en muchas aplicaciones como la conducción autónoma. En este proyecto

se desarrolló un sistema de detección de armas de fuego cortas en videos de cámaras de seguridad, mediante el uso de redes neuronales convolucionales.

Por último, se tiene en [3], En la actualidad, el rápido desarrollo de las tecnologías de inteligencia artificial es eficaz en el éxito de los algoritmos de aprendizaje profundo en diferentes áreas de aplicación. Estas aplicaciones detectan muchos objetos que ni siquiera el ojo humano puede detectar en la detección de objetos en videos e imágenes con algoritmos de aprendizaje profundo.

2.1. Violencia en Guayaquil

A comienzos del 2024 se declaró conflicto armado interno lo que redujo el crimen en los primeros meses del año, no obstante el crimen repunto y las estadísticas continúan siendo altas, durante el periodo de enero a agosto del 2024 hubo 4239 muertes violentas lo que significa una reducción del 17% en comparación al mismo periodo del 2023 en el cual se registraron 5097 muertes violentas, por esta problemática los habitantes de la ciudad buscan resguardarse del azote de la delincuencia de diferentes maneras, en los últimos años ha aumentado la instalación de rejas metálicas, cámaras de videovigilancia, alarmas y también guardias privados en los barrios de la ciudad.

Con este estudio se busca ayudar a la detección de las armas que son usadas a diario para cometer estos crímenes que se dan en los distintos puntos de la ciudad. En este contexto el uso de la Inteligencia Artificial resulta útil, mediante un algoritmo de red neuronal convolucional entrenado para reconocer armas especificas se puede hacer la identificación de las mismas.

En base al aumento de la delincuencia, surge la necesidad de usar los avances tecnológicos para contribuir a la seguridad de la comunidad, ayudando a la identificando del uso inadecuado de armas de fuego mediante el uso de la inteligencia artificial, así lograr llevar un registro del lugar en donde se están usando y poder llegar a la persona que la porta, logrando su detención y sacando de circulación estas armas que son usados con fines delictivos.

2.2. Violencia con armas de Fuego

La violencia con armas de fuego está ampliamente extendida en todo el mundo, es el medio que más se usa para cometer delitos de asesinatos, por la rapidez con la que cumplen su cometido y ya que son más efectivas que otras armas.

En el mundo hay en circulación más de mil millones de armas de fuego. La inmensa mayoría —el 85%— están en manos de particulares, el 13% están en arsenales militares y el 2% pertenecen a cuerpos encargados de hacer cumplir la ley.[4]

2.3. Inteligencia Artificial

La IA hace referencia a un conjunto de sistemas informáticos de aprendizaje y predicción. Una IA toma decisiones en base a predicciones basadas en datos con lo que ha sido entrenada y otros que adquiere al momento de ser usada. Por ejemplo, un coche autónomo decide por dónde y cómo conducir basándose en grandes datos almacenados y datos que provienen de sus cámaras.[5]

El objetivo de la inteligencia artificial es pensar y actuar como un humano, actualmente se usa en muchos y variados campos para así facilitar el trabajo de las personas como en el ámbito de salud para detectar enfermedades, los smartphones con los asistentes de voz también formar parte de la IA, al igual que las casas automatizadas que ya poseen múltiples características relacionadas con la IA como duchas con la temperatura programable o las luces automáticas. En el caso de este trabajo se enfocará en cómo usar la Inteligencia Artificial en el ámbito de la seguridad.

2.4. Redes Neuronales Artificiales

Una red neuronal es un método de la inteligencia artificial (IA) que enseña a las computadoras a procesar datos de una manera similar a como lo hace el cerebro humano. Se trata de un tipo de proceso de Machine Learning (ML) llamado aprendizaje profundo, el cual utiliza los nodos o las neuronas interconectados en una estructura de capas que se parece al cerebro humano. Crea un sistema adaptable que las computadoras utilizan para aprender de sus errores y mejorar continuamente. De esta forma, las redes neuronales artificiales intentan resolver problemas complicados, como la realización de resúmenes de documentos o el reconocimiento de rostros, con mayor precisión.[6]

Las RNA están formadas por nodos que se encuentran organizados en capas, la señal se envía a través de la capa de entrada y pasa por las capas ocultas que en el caso del Aprendizaje Profundo pueden ser muchas y luego sale por la capa de salida, para lograr mejores resultados es necesario entrenar esta red neuronal con una gran cantidad de datos a través de datos etiquetados o no etiquetados.

2.5. Redes Neuronales Convolucionales

Una red neuronal convolucional (CNN), también conocida como ConvNet, es un tipo especializado de algoritmo de aprendizaje profundo diseñado principalmente para tareas que requieren reconocimiento de objetos, como la clasificación, la detección y la segmentación de imágenes. Las CNN se emplean en diversos casos prácticos, como vehículos autónomos, sistemas de cámaras de seguridad y otros.[7]

La Red neuronal convolucional funciona con múltiples capas: capa convolucional, capa de agrupación y capa totalmente conectada. Siendo las capas convolucionales donde ocurre la mayor parte del proceso, las primeras capas reconocen características simples de la imagen de entrada como el color o borde, según avanza por las siguientes capas que también pueden ser convolucionales va reconociendo características más complejas, así hasta que identifica el objeto, luego la capa

de agrupación que barre toda la entrada con un filtro que se encarga de disminuir el tamaño de los mapas de características dados por la capa de convolución lo que ayuda a que el entrenamiento sea más rápido y se consuman menos recursos computacionales, por ultimo tenemos la capa totalmente conectada que se encarga de conectar los nodos de esta capa con los nodos de la capa anterior permitiendo identificar patrones más complejos.

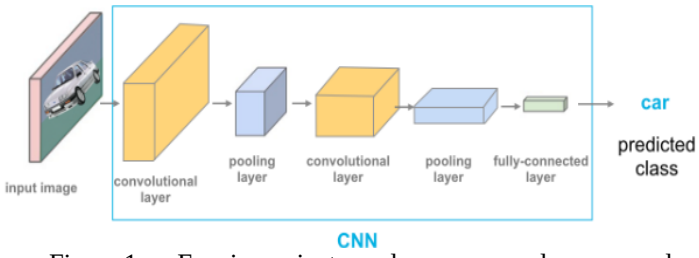

Fig. 1. Funcionamiento de una red neuronal convolucional.[8]

2.6. Aprendizaje Supervisado

El aprendizaje supervisado es un método del aprendizaje automático, que se basa en que el modelo es entrenado con un conjunto de datos con entradas y salidas (etiquetas), para así lograr una mejor predicción con nuevos datos. El aprendizaje supervisado se usa en dos casos: Clasificación; que es donde se asigna una etiqueta a una predicción como diferenciar patines de patineta. Y Regresión: que predice valores continuamente como el ejemplo las predicciones del clima.

En los algoritmos ML de aprendizaje supervisados los datos de ingreso se denominan datos de entrenamiento y tienen una etiqueta o resultado conocido, como spam/no-spam o un precio de acciones a la vez. Un modelo se prepara a través de un proceso de capacitación en el que se requiere hacer predicciones y se corrige cuando esas predicciones son erróneas. El proceso de capacitación continua hasta que el modelo alcanza el nivel deseado de precisión en los datos de capacitación.[9]

2.7. Yolov8

You Only Look Once, versión 8, es la última versión de los modelos de redes neuronales de la familia Yolo desarrollado por Ultralytics, son modelos diseñados para identificar en tiempo real con una alta precisión en comparación con versiones anteriores y manteniendo la velocidad, contiene un conjunto de datos llamado COCO que viene preentrenado para reconocer objetos particulares.

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. [10]

Yolov8 funciona mejor que sus predecesoras como Yolov5 o Yolov7, yolov8 esta hecho para detectar en tiempo real y segmentar, es mas sencillo al realizar el entrenamiento y el uso de hardware es moderado en comparación con modelos Faster R-CNN que requiere mas recursos y no es ideal para la identificación en tiempo real, otra diferencia es el uso Anchor Based(con anclas) que lo usan modelos como Yolov5, Yolov7 y Faster R-CNN se basa en cajas fijas predefinidas en cada parte de la imagen lo que hace el procesamiento mas lento, el modelo predice cual anchor box es el mejor para el objeto, si los objetos no coinciden bien con los anchors predefinidos, el modelo sufre.

En cambio, en modelos como Yolov8, no parte de cajas fijas, el modelo aprende directamente a decidir donde hay un objeto, predice el tamaño y posición directamente sin basarse en anclas prediseñadas lo que hace que el modelo trabaje de forma más rápida al momento de realizar una detección.

Se utilizan los resultados de las predicciones para generar las salidas finales que indican la presencia y ubicación de objetos en la imagen durante la etapa de detección. Asimismo, este establecerá las categorías a las que pertenece ese objeto y otras características relevantes. El enfoque modular de YOLOv8 posibilita procesar y analizar adecuadamente las características de la imagen en cada fase, lo que contribuye a su elevado rendimiento en la detección de objetos. [11]

Como se observa en la arquitectura de yolov8 en la figura 2, se tiene tres bloques, Backbone, Neck y Head.

En el bloque de Backbone se encuentra la extracción de características; Conv: Capas de convolución normales (filtros que detectan bordes, esquinas, etc.).

C2f: Módulo de atajos (shortcut) mejorado, parecido a lo que antes era CSP en YOLOv5 pero más optimizado (reduce parámetros).

SPPF: "Spatial Pyramid Pooling - Fast", que junta contexto espacial de varias resoluciones.

En el Segundo bloque Neck o Fusión de Características que se encuentra dentro del Backbone se tiene:

Upsample: Agranda (escala) mapas de características.

Concat: Junta características de diferentes niveles.

Porque queremos detectar armas grandes y pequeñas en la misma imagen.

Mezclar resoluciones ayuda a no perder detalles finos ni contexto amplio.

Por último, se tiene Head o Predicciones aquí se tiene el cambio mas importante que ya no usa Anchor Based, siendo sustituido por Anchor Free que como se indicó previamente ayuda a optimizar el procesamiento de las imágenes.

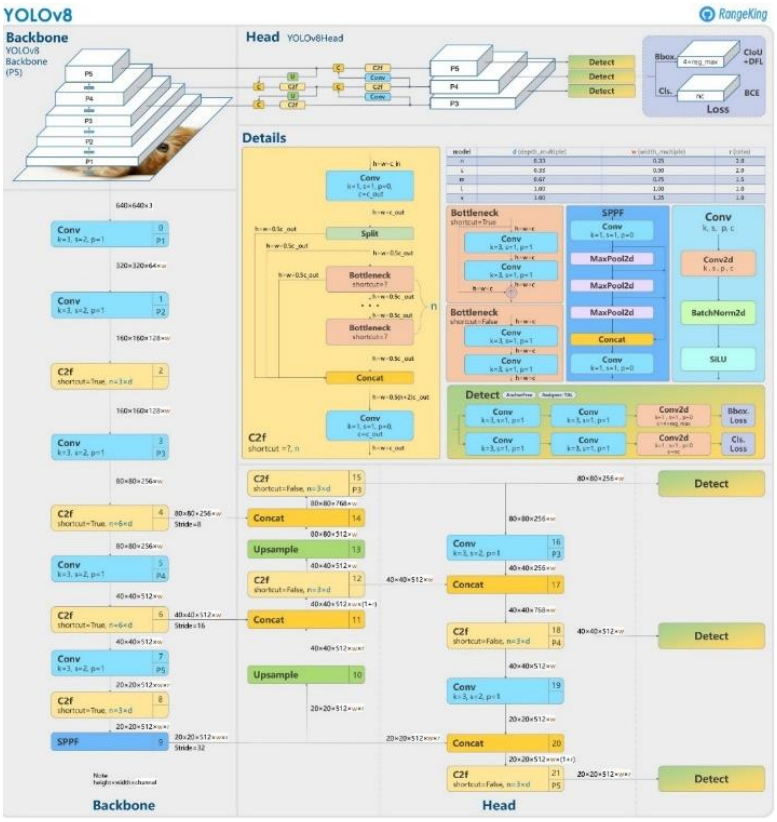

Fig. 2 Arquitectura de Yolov8. [12]

2.8. Google Colab

Google Colab es un entorno gratuito de Google que permite ejecutar código de Python para proyectos de Machine Learning, Deep Learning y procesamiento intensivo. Permite el acceso a GPU y TPU de forma gratuita si no se tiene el acceso a una GPU de forma local.

Google Colab es una excelente opción a la hora de realizar entrenamientos para detección de objetos por su capacidad de trabajo con hardware potente, aunque en su versión gratuita el tiempo es limitado y debe administrarse a la hora de trabajar.

2.9. Visual Studio Code

Visual Studio Code (VS Code) es un editor de código abierto gratuito para el desarrollo y la depuración de aplicaciones web y en la nube modernas, que está disponible de forma gratuita en Linux, OS X y Windows. VS Code es compatible con más de 30 lenguajes de programación, marcado y bases de datos diferentes, algunos de los cuales son JavaScript, C#, C++, PHP, Java, HTML, R, CSS, SQL, Markdown, TypeScript, Less, Sass, JSON, XML y Python. El editor ultrarrápido no solo satisface a los desarrolladores con la depuración integrada (incluidas las aplicaciones ASP.NET 5 y Node.js): incluso la compatibilidad con Git está disponible con solo presionar un botón. Numerosos atajos de teclado se encargan de su productividad en el trabajo. A diferencia de Visual Studio 2015 y sus predecesores, Visual Studio Code no trabaja con archivos de proyecto, sino con archivos y carpetas. El entorno se puede enriquecer de forma flexible mediante el uso de extensiones que permanecen independientes del sistema operativo subyacente. Code realmente califica como el complemento perfecto para las herramientas de desarrollo preferidas. [13]

2.10. Python

La sintaxis de Python es tan sencilla y cercana al lenguaje natural que los programas elaborados en Python parecen pseudocódigo. Por este motivo se trata además de uno de los mejores lenguajes para comenzar a programar.[14]

Está abierto a todos sin costo porque Python es de código abierto y su portabilidad hace posible su uso en muchas plataformas. El lenguaje en sí es liviano, abreviado y muy adecuado para la creación rápida de prototipos, aunque es lo suficientemente fuerte como para escribir aplicaciones significativas. [15]

3. Materiales y Métodos

3.1. Desarrollo del proyecto

Para la creación del conjunto de datos propio se comenzó realizando una búsqueda de imágenes en la red de los diferentes tipos de armas (pistolas, fusiles, escopetas, cuchillos, tijeras y machetes), completando 1380 imágenes entre imágenes tomadas de internet e imágenes propias tomadas en un barrio de la ciudad y en una finca en Vinces.

Además, usando armas de juguetes se realizó la toma de imágenes reales con una cámara celular que permite la ubicación GPS para así poder extraerla después, la descarga de estas imágenes se realizó directamente desde la nube para no perder la información de la metadata.

El siguiente paso fue realizar el etiquetado y asignación de la clase correspondiente se usó LabelMe que es una herramienta gratuita y de código abierto, es una herramienta de uso local por lo que no es necesaria la conectividad a internet ayudando a la protección de los datos, posee una interfaz sencilla y fácil de manejar.

Cuando se termina con el etiquetado se guardan todas las etiquetas en formato JSON en el directorio que se haya seleccionado, posteriormente para realizar el entrenamiento y se debe convertir a formato Yolo.

Para realizar el entrenamiento primero se debe verificar que la carpeta que se creó al convertir las imágenes a formato Yolo con el nombre YOLODataset este dividida en images y labels donde el 80% es para entrenamiento y el 20% para validación.

Se procede a entrenar el modelo en Google Colab que es un servicio gratuito de Google que, aunque permite trabajar por tiempo limitado, al trabajar directamente en el navegador, no se

debe instalar nada de manera local ahorrando tiempo y recursos, permite usar GPU avanzadas, al ingresar se debe verificar la GPU, se usa T4 GPU.

Se realiza la configuración necesaria para el entrenamiento para luego proceder con el mismo.

- Se Importa la unidad del Drive
- Se Instala Ultralytics
- Se Verifica Pytorch
- Si no está instalado, instalar
- Ahora ejecutar el script de entrenamiento, aquí se configuran las épocas que en este caso se usó 150, model se usa yolov8m y como batch se configura 30.

Para poder usar y probar el entrenamiento para obtener resultados, se desarrolló un programa usando Visual Studio Code en el cual se tiene las opciones de:

- Procesar imágenes de un directorio
- Procesar videos de un directorio
- Procesar cámara en vivo

Al ejecutar el programa se puede escoger entre estas tres opciones, al procesar las imágenes se selecciona un directorio el cual será procesado imagen por imagen, haciendo las detecciones y entregando la información de la metadata disponible, generando un archivo de Excel en el que se guarda toda esta información, adicional al ejecutar el programa durante la detección se muestra la ubicación GPS siempre que esté disponible.

Fig. 3. Funcionamiento del algoritmo detectando en imágenes. Elaboración Propia.

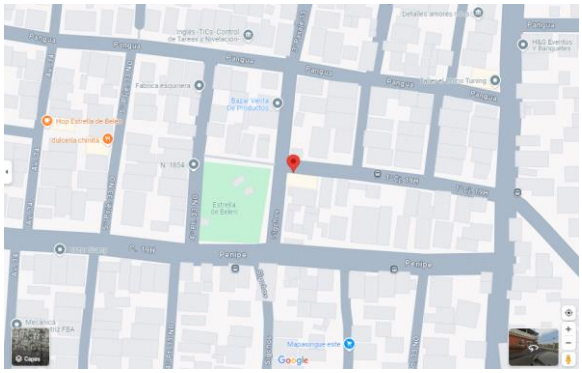

Fig. 4. Ubicación al ingresar en el mapa las coordenadas. Elaboración propia

En la segunda opción: procesamiento de videos de un directorio, el programa procesara los videos cuadro por cuadro, entregando como resultados las capturas de las detecciones y el Excel con las detecciones.

	O	P	Q	R	S	T	U	V
ta	Create Date	PS Altitud	PS Date/Tir	PS Latitud	PS Longitu	GPS Position	Class	Confid
15	2024:12:15 14:36:02	69.199	N/A	-2,12267	-79,9151	-2.1226694444444445, -79.91509722222223	machete	0.99
15	2024:12:15 14:36:03	69.199	N/A	-2,12267	-79,9151	-2.1226694444444445, -79.91509722222223	machete	0.98
15	2024:12:15 14:36:04	69.199	N/A	-2,12267	-79,9151	-2.1226694444444445, -79.91509722222223	machete	0.98
15	2024:12:15 14:36:04	69.199	N/A	-2,12267	-79,9151	-2.1226694444444445, -79.91509722222223	machete	0.98
15	2024:12:15 14:36:06	69.199	N/A	-2,12267	-79,9151	-2.1226694444444445, -79.91509722222223	escopeta	0.99
15	2024:12:15 14:36:07	69.199	N/A	-2,12267	-79,9151	-2.1226694444444445, -79.91509722222223	escopeta	0.99
15	2024:12:15 14:36:12	69.199	N/A	-2,12267	-79,9151	-2.1226694444444445, -79.91509722222223	escopeta	0.98
15	2024:12:15 14:37:27	69.199	N/A	-2,12273	-79,915	-2.122727777777778, -79.91498055555556	escopeta	0.99
15	2024:12:15 14:37:28	69.199	N/A	-2,12273	-79,915	-2.122727777777778, -79.91498055555556	escopeta	0.98
15	2024:12:15 14:37:44	69.199	N/A	-2,1227	-79,9152	-2.1227027777777776, -79.91515277777778	cuchillo	0.99
15	2024:12:15 14:37:45	69.199	N/A	-2,1227	-79,9152	-2.1227027777777776, -79.91515277777778	machete	0.98
15	2024:12:15 14:37:45	69.199	N/A	-2,1227	-79,9152	-2.1227027777777776, -79.91515277777778	machete	0.98
15	2024:12:15 14:37:56	69.199	N/A	-2,1227	-79,9152	-2.1227027777777776, -79.91515277777778	fusil	0.98
15	2024:12:15 14:38:04	69.199	N/A	-2,12263	-79,9151	-2.1226305555555556, -79.91509166666667	machete	0.99
15	2024:12:15 14:38:05	69.199	N/A	-2,12263	-79,9151	-2.1226305555555556, -79.91509166666667	machete	1.00

Fig. 5. Excel con toda la información metadata disponible de las imágenes procesadas por el algoritmo. Elaboración Propia.

La opción de cámara en vivo, permite procesar el modelo de identificación con la cámara en vivo, así mismo guarda los fotogramas de las detecciones y en ambos casos muestra la ubicación GPS durante la ejecución y guarda toda la información metadata disponible.

3.2. Matriz de Confusión

Una matriz de confusión (o matriz de error) es un método de visualización para los resultados del algoritmo clasificador. Más específicamente, es una tabla que desglosa el número de instancias reales de una clase específica frente al número de instancias previstas para esa clase. Las matrices de confusión son una de las varias métricas de evaluación que miden el rendimiento de un modelo de clasificación. Se pueden utilizar para calcular otras métricas de rendimiento del modelo, como la precisión y la coincidencia, entre otras.[16]

	Positivo real	Negativo real
Positivo previsto	**Verdadero positivo (VP)**: Es un correo electrónico de spam clasificado correctamente como correo electrónico spam. Estos son los mensajes de spam que se envían automáticamente a la carpeta de spam.	**Falso positivo (FP)**: Un correo electrónico que no es spam clasificado erróneamente como tal. Estos son los correos electrónicos legítimos que terminan en la carpeta de spam.
Negativo previsto	**Falso negativo (FN)**: Un correo electrónico de spam clasificado erróneamente como no spam. Son correos electrónicos de spam que el filtro de spam no detecta y que llegan a la carpeta Recibidos.	**Verdadero negativo (VN)**: Un correo electrónico que no es spam y que se clasificó correctamente como tal. Estos son los correos electrónicos legítimos que se envían directamente a la carpeta Recibidos.

Fig. 6. Resultados de una matriz de confusión [17]

- Verdaderos Positivos (VP): Estos son los valores donde la clase real era positivo y el modelo también lo detecto como positivo.
- Falsos Positivos (FP): Estos son los valores donde la clase real era negativa y el modelo lo detecto como positivo.
- Falsos Negativos (FN): Estos son los valores donde la clase real era positivo y el modelo lo detecto como negativo
- Verdaderos Negativos (VN): Estos son los valores donde la clase real era negativo y el modelo lo detecto como negativo.

De estos valores se puede calcular varias métricas para medir el funcionamiento del modelo de red neuronal para detección de armas. Las métricas a continuación se usaron para medir el rendimiento del modelo en las matrices por clase.

Exactitud: Este indicador permite calcular la cantidad de predicciones positivas que en realidad son correctas.

$$Exactitud = \frac{VP + VN}{VP + VN + FN + FP}$$

Precisión: Este indicador permite calcular la cantidad predicciones correctas del modelo tanto positivas como negativas.

$$Precisión = \frac{VP}{VP + FP}$$

Sensibilidad o recall: Este indicador permite calcular la proporción de verdaderos positivos que fueron correctamente identificados.

$$Sensibilidad\ o\ Recall = \frac{VP}{VP + FN}$$

F1 Score: Esta métrica es una combinación de la precisión y la sensibilidad.
Un valor cercano a 1 indica un buen rendimiento del modelo, mientras que un valor cercano a 0 indica un mal rendimiento.

$$F1\ Score = 2 * \frac{Precisión * Sensibilidad}{Precisión + Sensibilidad}$$

Adicional, en las matrices multiclase se calculó dos métricas globales:

$$Exactitud = \frac{\sum TP}{Total\ de\ instancias}$$

F1 Score Macro: Combina la precisión y la sensibilidad de todas las clases. Les da la misma importancia a todas las clases sin importar el número de instancias que se tenga de cada una. Para esta métrica primero debemos calcular la precisión, sensibilidad y F1 score por clase con las fórmulas estudiadas previamente.

$$F1\ Score\ Macro = \frac{\sum F1\ de\ todas\ las\ clases}{Número\ de\ clases}$$

Primero se debe extraer los verdaderos positivos, falsos positivos y falsos negativos por clase.

Para los verdaderos positivos por clase se tiene los valores de la diagonal, para los falsos positivos serán la suma de los valores de la columna de la clase menos el valor de la diagonal y para los falsos negativos serán la suma de los valores de la fila de la clase menos el valor de la diagonal.

3.2.1. Matrices de confusión por clases

Para la elaboración de las matrices de confusión se tomó 116 imágenes, en las matrices se tiene filas y columnas, donde las filas son las predicciones y las columnas los valores reales, aquí se procesó las imágenes por cada clase.

Tabla 1. Matriz de Confusión Fusil

		Realidad	
		Fusil	No Fusil
Predicción	Fusil	19	15
	No Fusil	3	79

Tabla 2. Matriz de Confusión Escopeta

		Realidad	
		Escopeta	No Escopeta
Predicción	Escopeta	15	22
	No Escopeta	1	78

Tabla 3. Matriz de Confusión Pistola

		Realidad	
		Pistola	No Pistola
Predicción	Pistola	9	3
	No Pistola	13	91

Tabla 4. Matriz de Confusión Tijera

		Realidad	
		Tijera	No Tijera
Predicción	Tijera	15	47
	No tijera	0	54

Tabla 5. Matriz de Confusión Cuchillo

		Realidad	
		Cuchillo	No Cuchillo
Predicción	Cuchillo	0	1
	No Cuchillo	30	85

Tabla 6. Matriz de Confusión Machete

		Realidad	
		Machete	No Machete
Predicción	Machete	4	19
	No Machete	9	84

3.2.2. Matriz de Confusión imágenes reales

A continuación, se tomó el conjunto de imágenes tomadas en un escenario real en un parque de un barrio de la ciudad de Guayaquil (norte), aquí tenemos 171 imágenes formadas por 5 de las 6 clases las cuales son pistola, escopeta, machete, tijera y cuchillo adicional se incorporó una fila y una columna para registrar las imágenes en las que no hubo detección.

Tabla 7. Matriz de Confusión Multiclase imágenes reales

		Predicción					
		Pistola	Escopeta	Machete	Cuchillo	Tijera	No detecto
Realidad	Pistola		1	1	1		25
	Escopeta		18	2			37
	Machete		1				18
	Cuchillo		1			1	31
	Tijera					15	19
	Sin predicción						

3.2.3. Matriz de Confusión imágenes de la red

Para la elaboración de la siguiente matriz de confusión multiclase se extrajeron 150 imágenes de la red, 25 por cada clase; pistola, fusil, escopeta, machete, cuchillo y tijera adicional se incorporó una fila y una columna para registrar las imágenes en las que no hubo detección.

Tabla 8. Matriz de Confusión Multiclase imágenes de la red

		Predicción						
		Pistola	Escopeta	Fusil	Machete	Cuchillo	Tijera	No detecto
Realidad	Pistola	2	3	5	0	0	5	10
	Escopeta	0	18	1	0	0	6	0
	Fusil	0	4	18	0	0	1	2
	Machete	0	2	0	14	0	4	5
	Cuchillo	0	0	1	12	0	10	2
	Tijera	0	0	0	0	0	25	0
	No detecto							

4. RESULTADOS

A continuación, se presentarán los resultados de las pruebas realizadas del modelo de red neuronal, entrenada en Google Colab:

Resultados prueba 1 realizada con 116 imágenes extraídas de la red, donde observamos el rendimiento por clase.

Tabla 9. Resultados Prueba 1

Clase	Exactitud	Precisión	Sensibilidad	F1 Score
Fusil	0,84	0,56	0,86	0,68
Escopeta	0,80	0,41	0,94	0,57
Pistola	0,86	0,75	0,41	0,53
Tijera	0,59	0,24	1	0,39
Cuchillo	0,73	0	0	0
Machete	0,76	0,17	0,31	0,22

Nota: Grados de exactitud, precisión, sensibilidad y F1 Score obtenidos de las matrices por clase en la primera prueba. Elaboración Propia.

Resultados prueba 2 realizada con 171 tomadas en un escenario real (parque), donde observamos el rendimiento de 5 de las 6 clases y el rendimiento global del modelo.

Tabla 10. Resultados Prueba 2

Clase	Precisión	Sensibilidad	F1 Score
Pistola	0	0	0
Escopeta	0,86	0,32	0,46
Machete	0	0	0
Cuchillo	0	0	0
Tijera	0,94	0,44	0,60

Nota: Grado de precisión, sensibilidad y F1 Score obtenidos de la matriz multiclase en la segunda prueba. Elaboración propia.

Tabla 11. Resultados Globales Prueba 2

Métricas Globales	
Exactitud	F1 Score Macro
0,19	0,21

Nota*:* Grado de exactitud global y F1 Score Macro calculados en la segunda prueba. Elaboración propia.

A continuación, resultados prueba 3 realizada con 150 imágenes extraídas de la red aquí veremos el rendimiento del algoritmo con las 6 clases y el rendimiento global del modelo.

Tabla 12. Resultados Prueba 3

Clase	Precisión	Sensibilidad	F1 Score
Pistola	1	0,08	0,15
Escopeta	0,67	0,72	0,69
Fusil	0,72	0,72	0,72
Machete	0,54	0,56	0,55
Cuchillo	0	0	0
Tijera	0,49	1	0,66

Nota: Grado de precisión, sensibilidad y F1 Score obtenidos de la matriz multiclase en la tercera prueba. Elaboración propia.

Tabla 13. Resultados Globales Prueba 3

Métricas Globales	
Exactitud	F1 Score Macro
0,51	0,46

Nota: Grado de exactitud global y F1 Score Macro calculados en la tercera prueba. Elaboración propia.

5. CONCLUSIONES

En la actualidad, cada vez más se usan los algoritmos de detección en distintos campos y aplicaciones. En este caso se aplica al tema de la seguridad para lograr mejorar la detección de armas lugares públicos.

- Los métodos de detección de objetos usando aprendizaje profundo son de gran utilidad acompañado de la tecnología necesaria para identificar situaciones de peligro en las que se ven involucradas armas de todo tipo en conjunto con la ubicación donde ocurren estos incidentes es una aplicación de gran utilidad y relevancia para la seguridad.
- Usar un modelo de YOLOV8 es una buena opción ya que es un modelo especialmente diseñado para tareas de detección, segmentación y clasificación de acceso gratuito al igual que Google Colab que nos ofrece buenas características de hardware para entrenar el modelo.
- Crear un conjunto de datos propio, aunque puede resultar demandante en cuanto el tiempo empleado, es ideal porque el investigador o desarrollador del proyecto escoge el mismo las imágenes de acuerdo a sus necesidades tanto como para entrenar, validación y luego probar el funcionamiento del algoritmo.
- El modelo tiene rendimiento medio en cuanto detección de armas de fuego y en cuanto a armas blancas el algoritmo tiene un rendimiento bajo tendiendo a confundir las

armas. Tuvo un rendimiento mejor al realizar las pruebas con imágenes extraídas de la red ya sea por la calidad de imágenes o por la cercanía de los objetos que en las imágenes reales tomadas con la cámara celular se veían un poco distantes. Como recomendación en cuanto a la identificación de armas blancas, se tiene que se puede aumentar la cantidad de imágenes al realizar el entrenamiento, procesando imágenes de diferentes tipos y con diferente iluminación. El GPS en las imágenes y videos funciona bien entregándonos las coordenadas del lugar donde fue tomada la imagen o video, mientras que en la cámara en vivo solo nos da una ubicación aproximada basándose en la dirección IP.

REFERENCIAS

[1] Jorge Sebastián Gutiérrez Valverde, "DESARROLLO DE UN SISTEMA PARA EL ENVÍO AUTOMÁTICO DE INFORMACIÓN DE ALERTA EN EVENTOS DE ASALTO EN EL INTERIOR DE AUTOMÓVILES DE TRANSPORTE PÚBLICO," Universidad de las Fuerzas Armadas, Sangolquí, 2022.

[2] David Orlando Romero Mogrovejo, "DESARROLLO DE UN SISTEMA DE DETECCIÓN DE ARMAS DE FUEGO CORTAS EN EL MONITOREO DE VIDEOS DE CÁMARAS DE SEGURIDAD," Universidad Politécnica Salesiana, Cuenca, 2018.

[3] M. Burgaz and C. Budak, "Detection of Object (Weapons) with Deep Learning Algorithms from Images Obtained by Unmanned Aerial Vehicles," *DÜMF Mühendislik Dergisi*, vol. 2, pp. 263–270, 2022, doi: 10.24012/dumf.1116534.

[4] "Amnistía Internacional." [Online]. Available: https://www.amnesty.org/es/what-we-do/arms-control/gun-violence/#:~:text=La violencia con armas de fuego puede robar a la,a los centros de salud.

[5] J. Cárdenas, "Artificial intelligence, research and peer-review: Future scenarios and action strategies," *Revista Espanola de Sociologia*, vol. 32, no. 4, pp. 1–15, 2023, doi: 10.22325/fes/res.2023.184.

[6] "AWS." [Online]. Available: https://aws.amazon.com/es/what-is/neural-network/#:~:text=Una red neuronal es un,de rostros%2C con mayor precisión.

[7] K. Zoumana, "Introducción a las redes neuronales convolucionales (CNN)." [Online]. Available: https://www.datacamp.com/es/tutorial/introduction-to-convolutional-neural-networks-cnns

[8] C. Cézanne, "Redes neuronales convolucionales." [Online]. Available: https://cezannec.github.io/Convolutional_Neural_Networks/

[9] J. F. Correa Wachter, C. F. Henao Villas, F. Henao Villa, and D. A. García Arango, "Análisis del aporte del aprendizaje de máquinas a la seguridad de la información," *InGente Americana*, vol. 1, no. 1, pp. 9–20, 2021, doi: 10.21803/ingecana.1.1.407.

[10] A. Schcolnik-Elias, S. Martínez-Díaz, J. E. Luna-Taylor, and I. Castro-Liera, "Detección de armas tipo pistola mediante el uso de redes convolucionales con una arquitectura tipo YOLO y estereoscopía.," 2023.

[11] J. A. G. Blanco, J. C. S. Soto, J. A. V. Farias, and L. E. A. Carranza, "Implementación De Yolov8 Para La Clasificación De Cubos De Colores En Tiempo Real," pp. 6–7, 2023.

[12] RangeKing, "Brief summary of YOLOv8 model structure." [Online]. Available: https://github.com/ultralytics/ultralytics/issues/189

[13] T. Kahlert and K. Giza, "Visual Studio Code - Code Editing. Redefined," *Microsoft*, vol. 1, no. March, pp. 1–26, 2016, [Online]. Available: http://download.microsoft.com/download/8/A/4/8A48E46A-C355-4E5C-8417-E6ACD8A207D4/VisualStudioCode-TipsAndTricks-Vol.1.pdf

[14] R. Gonzaléz, *Python para todos*.

[15] Z. A. Aziz, D. Naseradeen Abdulqader, A. B. Sallow, and H. Khalid Omer, "Python Parallel Processing and Multiprocessing: A Rivew," *Academic Journal of Nawroz University*, vol. 10, no. 3, pp. 345–354, 2021, doi: 10.25007/ajnu.v10n3a1145.

[16] E. Kavlagoklu and J. Murel, "¿Qué es una matriz de confusión?" [Online]. Available: https://www.ibm.com/mx-es/topics/confusion-matrix

[17] "Machine Learning - Conceptos de Aprendizaje Automático." [Online]. Available: https://developers.google.com/machine-learning/crash-course/classification/thresholding?hl=es-419