DETECCIÓN DE ESCENAS DE RIESGO EN TRANSPORTE PÚBLICO UTILIZANDO APRENDIZAJE PROFUNDO

DETECTION OF RISK SCENE ON PUBLIC TRANSPORT USING DEEP LEARNING

Francisco García Arellano
Universidad Autónoma Metropolitana, Unidad Azcapotzalco, México
al210204967@azc.uam.mx

Carlos Avilés Cruz
Universidad Autónoma Metropolitana, Unidad Azcapotzalco, México
caviles@azc.uam.mx

Gabriel de Jesús Celis Escudero
Universidad Autónoma Metropolitana, Unidad Azcapotzalco, México
al2232801411@azc.uam.mx

Recepción: 19/marzo/2026 **Aceptación:** 25/mayo/2026

Resumen

Este documento presenta una propuesta para mejorar la seguridad en el transporte público mediante la detección temprana y automática de situaciones potencialmente riesgosas. Para ello, se diseñó y evaluó un sistema de visión artificial basado en redes neuronales convolucionales, capaz de clasificar imágenes en cuatro categorías de niveles de riesgo, las cuales son “no riesgo”, “amenaza”, “arma” y “no arma”. El modelo fue entrenado utilizando una base de datos compuesta de 4272 imágenes estandarizadas y equilibrar las muestras por categoría, el sistema demostró una alta precisión diagnóstica. El modelo alcanzó una exactitud del 98.39% en pruebas generales y el desempeño en validación cruzada por clase son: “no riesgo”, 100%; “amenaza”, 87.91%; “arma”, 89.6% y “no arma”, 87.91%. Estos resultados validan la viabilidad de integrar el sistema en dispositivos inteligentes para monitoreo en tiempo real, facilitando respuestas inmediatas ante actos delictivos.

Palabras Clave: Imágenes, redes neuronales convolucionales, transporte público.

Abstract

This document presents a proposal to improve safety in public transportation by automatically and early detecting potentially risky situations. To this end, we designed and evaluated a computer vision system based on convolutional neural networks that can classify images into four risk level categories: "no risk," "threat," "weapon," and "non-weapon." The model was trained using a balanced database of 4,272 *standardized images, and the system demonstrated high diagnostic accuracy. It achieved* 98.39% *accuracy in general tests, and the cross-validation performance by class is as follows: "No risk":* 100%*; "Threat":* 87.91%*; "Weapon":* 89.6%*; and "Non-weapon":* 87.91%*. These results validate the feasibility of integrating the system into smart devices for real-time monitoring and immediate response to criminal acts.*

Keywords: *Convolutional neural networks, images, public transportation.*

1. Introducción

En los últimos años, el uso de redes neuronales convolucionales para la detección de objetos ha aumentado significativamente y ha probado su robustez en diversos contextos, desde la clasificación de animales hasta la identificación de objetos en imágenes complejas. En este marco, la clasificación de objetos permite inferir el tipo de escenario o entorno presente en una imagen, basándose en los elementos detectados. Asimismo, segmentar zonas específicas dentro de la imagen procesada para determinar si corresponden a ubicaciones particulares dentro del plano geográfico.

Actualmente, existen sistemas de clasificación multiclase basados en redes convolucionales que se especializan en reconocer escenas a partir del contenido visual. Estos sistemas pueden identificar la presencia de individuos portando ciertos objetos, determinar si una escena pertenece a un sitio previamente conocido por el modelo o comprobar si contiene elementos similares a los que ya se encuentran en la base de datos con la que se entrena el modelo. Tales coincidencias permiten inferir con mayor precisión si una imagen representa un escenario específico. Estas coincidencias entre objetos, comportamientos y ubicaciones son interpretadas por

el sistema como "características", las cuales se evalúan en función de la probabilidad de su aparición en la imagen procesada. Para lograr una clasificación precisa, estos sistemas requieren bases de datos extensas y variadas, compuestas por numerosas clases, cada una con una cantidad representativa de imágenes [Parseh, 2022].

En contraste con estos enfoques generalistas, el presente trabajo propone un modelo específicamente diseñado para la clasificación de escenarios de riesgo y no riesgo, considerando la presencia o ausencia de armas en las imágenes. Si bien el comportamiento humano es complejo y, tratándose de imágenes estáticas, resulta aún más difícil predecir las acciones que un individuo podría realizar, el estudio de Surendran [2023] aborda este desafío con éxito. Dicho trabajo demuestra que es posible predecir el comportamiento humano a partir de imágenes fijas, sin requerir un seguimiento temporal continuo de las acciones.

El modelo de Surendran emplea redes preentrenadas y robustas como *AlexNet*, *SqueezeNet*, *ResNet* y *DenseNet* para generar un mapa de características por clase, lo que le permite alcanzar una precisión del 94.7%. Aunque el enfoque propuesto en el presente estudio no cuenta con una arquitectura tan robusta, se han obtenido resultados favorables mediante un modelo adaptado específicamente para predecir acciones en escenarios con menor complejidad y número de variables.

Los seres humanos poseemos una capacidad innata para reconocer con rapidez el tipo de escenario en el que nos encontramos, procesando visualmente una gran cantidad de información con apenas un vistazo. Sin embargo, las máquinas no gozan de esta eficiencia cognitiva y enfrentan mayores dificultades para identificar entornos de manera confiable. En este sentido Yeo [2020] propone un modelo de red neuronal convolucional inspirado en la forma en que los humanos reconocemos escenas: identificando no solo los objetos principales, sino también aquellos presentes en el fondo de la imagen.

Este método logró mejorar la precisión de arquitecturas convencionales como *VGG*, *Inception*, *ResNet*, *ResNeXt*, *Wide-ResNet*, *DenseNet* y *MnasNet*, alcanzando una tasa de acierto del 90.7%. El modelo segmenta las imágenes para extraer cada objeto presente, incluyendo aquellos en segundo plano. Los resultados indican que

aplicar segmentación semántica es más eficiente y ligero que desarrollar redes especializadas para la clasificación de escenarios generales. En otras palabras, esta técnica es idónea para escenas con profundidad significativa, donde los objetos lejanos son cruciales para la clasificación.

Por otro lado, trabajos previos como el de Nadeem [2024] han propuesto conjuntos de datos sintéticos exitosos para la detección de violencia armada, como *Weapon Violence Dataset 2.0,* generado a partir de escenas controladas en un entorno de videojuego, persisten desafíos importantes al trasladar dichos modelos a contextos del mundo real. Aunque este tipo de recursos facilita la anotación automática y el control completo sobre variables del entorno, también implica ciertas limitaciones.

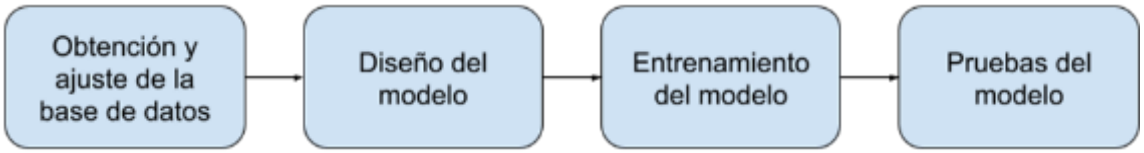
Específicamente, la carencia de realismo visual y conductual, así como la ausencia de comportamientos impredecibles, pueden inducir “sesgos estructurales” en los modelos de aprendizaje profundo entrenados sobre estas bases de datos. Dichos sesgos podrían disminuir la eficacia del sistema al aplicarse en condiciones no controladas, lo cual subraya la necesidad de combinar estos recursos sintéticos con datos reales o mecanismos de validación cruzada que fortalezcan su capacidad de generalización.

En la sección de métodos se describen los módulos en los que se divide el proyecto para el diseño del sistema de clasificación de escenarios. En la sección de resultados se presentan los valores de exactitud y pérdida obtenidos durante el entrenamiento del modelo de aprendizaje profundo, utilizando el método de validación. La sección de discusión expone los principales contratiempos y fallas encontrados durante el desarrollo del proyecto, así como posibles soluciones para abordarlos. Finalmente, en la sección de conclusiones se proponen mejoras y actualizaciones que podrían implementarse en futuros trabajos relacionados.

2. Métodos

Las etapas de desarrollo del proyecto se muestran en la Figura 1. La base de datos utilizada en este proyecto se recolectó mediante un proceso de curación manual de fuentes digitales de acceso público, principalmente de plataformas de video como YouTube [YouTube, 2005] y redes sociales como Facebook [Facebook,

2004]. La información utilizada proviene de diferentes escenarios que representan situaciones de riesgo en entornos de transporte público. Estos escenarios incluyen la presencia de armas, la inminencia de un acto delictivo, así como contextos sin ningún tipo de peligro.



Fuente: elaboración propia.

Figura 1 Diagrama a bloques de los pasos a realizar en el proyecto.

Para asegurar la uniformidad en el análisis, todas las imágenes fueron estandarizadas a una resolución de $640 \times 358\, píxeles$ y convertidas al formato RGB, lo cual permite establecer un formato consistente para su uso posterior en el entrenamiento del modelo.

En la Figura 2 se muestra gráficamente cada tipo de escena por las que está compuesta la base de datos y su correcta clasificación. Los fotogramas seleccionados incluyen una variedad de situaciones que permiten definir cuatro clases distintas, según el nivel de riesgo y la visibilidad del arma en la escena:

- No riesgo: Son todas esas escenas donde no se identifica presencia de arma ni intención de asalto como se puede observar en la Figura 2a, es una escena donde únicamente se puede visualizar a los pasajeros de transporte público en condiciones normales, sin agresión alguna.
- Amenaza: Da inicio a una situación de riesgo, donde se aprecia la presencia del arma que se utiliza para cometer un asalto, el arma no es visible en su totalidad desde el ángulo de visión, un ejemplo de este escenario se muestra en la Figura 2b.
- Arma: Se detecta la presencia explícita de un arma en el campo visual del sistema, con esto se asegura que existe una situación de riesgo (Figura 2c).
- No arma: Existe una amenaza potencial, esta clase representa todas esas escenas donde se está llevando a cabo un asalto, pero el arma no es visible en la escena, este tipo de escenarios se ejemplifican en la Figura 2d.

Cada fotograma fue etiquetado manualmente de acuerdo con una de estas cuatro clases. Posteriormente, se realizó un proceso de limpieza y homogeneización de la base de datos para asegurar un equilibrio entre clases, con el fin de evitar un entrenamiento sesgado del modelo de aprendizaje profundo.

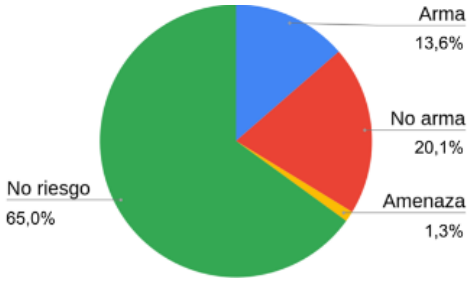
Durante el análisis preliminar de la distribución de clases, se identificó un desequilibrio significativo en la cantidad de fotogramas por clase, esto se puede visualizar en la Figura 3. Para corregir esta disparidad, se tomó como referencia la clase menos representada y se aplicaron técnicas de aumento de datos, permitiendo así nivelar la cantidad de muestras en cada categoría. Como resultado, la base de datos final quedó compuesta por $4{,}272$ fotogramas, distribuidos equitativamente en $1{,}067$ fotogramas por clase, garantizando así condiciones adecuadas para el entrenamiento del modelo.

a) Clase "no riesgo". *b) Clase "amenaza".*

c) Clase "arma". *d) Clase "no arma".*

Fuente: elaboración propia.

Figura 2 Descripción gráfica de cada una de las clases.



Fuente: elaboración propia

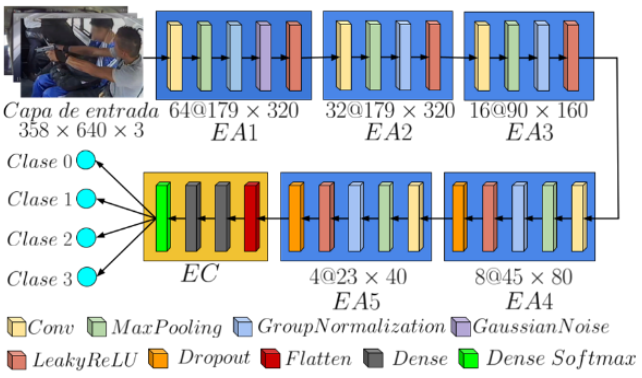
Figura 3 Gráfica representativa de la base de datos desbalanceada.

Una vez completado el proceso de clasificación de los fotogramas, se procede al diseño del modelo de aprendizaje profundo. El modelo propuesto se basa en los principios de implementación de sistemas inteligentes mediante el uso de redes neuronales y herramientas como Keras y TensorFlow [Géron, 2022].

La arquitectura se compone de cinco etapas principales de extracción de características (EA1–EA5), las cuales integran técnicas de regularización para asegurar una mejor generalización y evitar el sobreajuste del modelo [Brownlee, 2019], seguidas de una etapa final de clasificación (EC):

- EA1: Capa convolucional con 64 filtros.
- EA2: Capa convolucional con 32 filtros.
- EA3: Capa convolucional con 16 filtros.
- EA4: Capa convolucional con 8 filtros, incorporando *dropout* (apagado aleatorio de neuronas) para mejorar la capacidad de generalización.
- EA5: Capa convolucional con 4 filtros, también con *dropout* para reforzar la regularización previa a la clasificación.

Finalmente, la etapa de clasificación (EC) recibe las características extraídas y procesadas en las etapas anteriores, generando la predicción final. Cada una de estas etapas está conformada por un conjunto específico de capas, diseñadas para optimizar la representación de los datos y mejorar el rendimiento del modelo. En la Figura 4 se muestra la representación gráfica de las capas que componen cada etapa.



Fuente: elaboración propia.

Figura 4 Descripción gráfica de las capas que componen el modelo.

La primera capa es una convolucional (Conv2D) que aplica 64 filtros de tamaño 7×7 a la entrada, utilizando padding igual a “same” para conservar las dimensiones espaciales originales de 358×640. Esta capa permite extraer patrones espaciales relevantes de la entrada, cuya dimensión inicial es 358×6403, y produce una salida de tamaño $64@358 \times 640$.

A continuación, se utiliza una capa de submuestreo (MaxPooling2D) con stride igual a 2×2, también con padding igual a “same”, lo cual reduce las dimensiones espaciales a la mitad manteniendo la proporción, resultando en una salida de tamaño $64@179 \times 320$. Esta operación reduce el sobreajuste y el costo computacional del modelo. Seguidamente, se aplica una capa de normalización por grupos (GroupNormalization), que divide los canales de salida en 16 grupos para normalizar sus valores. Esta operación no altera las dimensiones de salida, que permanecen en $64@179 \times 320$.

Luego, se incorpora una capa de ruido gaussiano (GaussianNoise) que añade ruido aleatorio a la entrada, con el objetivo de aumentar la robustez del modelo. Esta capa tiene una media cero por defecto, y se emplea una desviación estándar típica de 0.01. La adición de ruido no modifica la forma del tensor, por lo tanto, la salida conserva las dimensiones $64@179 \times 320$.

La siguiente capa es una función de activación LeakyReLU, configurada con un parámetro Alpha igual a 0.02. Esta función introduce un pequeño gradiente negativo en valores negativos para evitar el problema de las neuronas “muertas” (dying ReLU), lo que favorece una mejor propagación del error durante el entrenamiento.

Posteriormente, se emplea otra capa convolucional con 32 filtros de tamaño 7×7, manteniendo el padding igual a “same”, lo que produce una salida de tamaño $32@179 \times 320$. Esta es seguida por una nueva capa MaxPooling2D con stride igual a 22, lo que reduce la dimensión a $32@90 \times 160$.

La salida se normaliza nuevamente con GroupNormalization, esta vez dividiendo los canales en 8 grupos. Se aplica otra función LeakyReLU para asegurar la activación de valores previamente suprimidos. Luego, se utiliza otra capa convolucional con 16 filtros de tamaño 7×7, obteniendo una salida de tamaño $16@90 \times 160$.

Una nueva operación de submuestreo mediante MaxPooling2D con stride igual a 2×2 reduce aún más la dimensión a $16@45 \times 80$. Este patrón de capas se repite para permitir una reducción progresiva y una abstracción jerárquica de las características.

En resumen:

- Las capas Conv2D extraen características con un número decreciente de filtros (32, 16, 8 y 4).
- Las capas MaxPooling2D realizan submuestreo para reducir la dimensionalidad.
- Las capas GroupNormalization estabilizan la activación mediante normalización por grupos.
- Las capas LeakyReLU reactivan neuronas que podrían haber quedado sin actualizar durante el entrenamiento.
- Finalmente, se incorpora Dropout en las capas 4 y 5 para desactivar aleatoriamente neuronas durante el entrenamiento, reduciendo así el riesgo de sobreajuste.

Después de completar todas las etapas anteriores, la entrada a la última sección del modelo tiene dimensiones $4@12 \times 20$. Esta etapa corresponde a la parte de clasificación del modelo, implementada mediante una red completamente conectada (fully connected), compuesta por las capas Flatten y Dense.

La capa Flatten convierte el tensor tridimensional (altura, ancho, canales) en un vector unidimensional. En este caso, transforma la entrada $4@12 \times 20$ en un vector de tamaño 960, que sirve como entrada para las capas densas posteriores.

A continuación, se emplea una capa Dense con aproximadamente 293 neuronas, utilizando la función de activación SELU (Scaled Exponential Linear Unit), la cual proporciona propiedades de auto-normalización que favorecen la estabilidad en la propagación del gradiente durante el entrenamiento. Luego, se introduce una segunda capa Dense que reduce la dimensionalidad, con alrededor de 146 neuronas, manteniendo la misma activación SELU para conservar la estabilidad del modelo.

Finalmente, se incluye la capa de salida, también de tipo Dense, compuesta por 4 neuronas y una función de activación Softmax, que transforma los valores de salida en probabilidades, permitiendo realizar una clasificación multiclase.

En resumen, el modelo propuesto es una red neuronal convolucional profunda que incorpora mecanismos de extracción jerárquica de características, normalización por grupos (GroupNormalization), adición de ruido (GaussianNoise), y regularización mediante Dropout. Las funciones de activación empleadas incluyen LeakyReLU en las capas convolucionales y SELU en las densas, mientras que la capa final con activación Softmax permite clasificar entradas en una de las cuatro clases definidas.

La arquitectura del modelo se diseñó desde cero como una red personalizada. Se optó por esta configuración propia en lugar de arquitecturas preentrenadas más robustas (como ResNet o VGG) con el objetivo de priorizar la ligereza estructural. Este diseño permite alcanzar un equilibrio óptimo entre la profundidad necesaria para identificar características complejas de riesgo y la eficiencia computacional requerida para el procesamiento en tiempo real. Como resultado, el modelo cuenta con un total de $468{,}145$ parámetros de aprendizaje, lo que representa una estructura ligera y eficiente, ideal para su despliegue en sistemas embebidos de transporte público sin comprometer la capacidad de procesamiento en tiempo real.

El modelo se entrenó utilizando un script desarrollado en Python versión $3.9.18$ con el respaldo de las librerías *TensorFlow* y *Keras*. Se configuró una tasa de aprendizaje de 0.0005 mediante el optimizador *Adam*. Debido a que el tamaño de la base de datos es amplio, se utilizó una separación por lotes para optimizar el uso de la memoria, definiendo un tamaño de lote de 64. El entrenamiento se extendió por un total de $2{,}000$ épocas, lo que permitió una convergencia estable de los valores de exactitud y pérdida reportados.

3. Resultados

El sistema fue evaluado mediante el método de resustitución, el cual es comúnmente empleado para medir el rendimiento de modelos de clasificación o regresión al utilizar el mismo conjunto de datos con el que el modelo fue entrenado.

En otras palabras, este método permite estimar la capacidad del modelo para predecir correctamente las etiquetas de los datos que ya ha observado durante el proceso de entrenamiento.

En cuanto al uso de los diferentes subconjuntos de datos, se establecieron las siguientes fases:

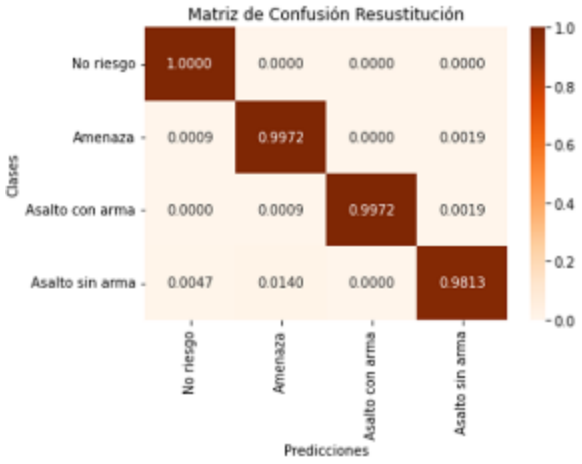
- Conjunto de entrenamiento: utilizado para ajustar los parámetros del modelo.
- Conjunto de validación: empleado para evaluar el rendimiento del modelo sobre datos previamente vistos, permitiendo afinar hiperparámetros y prevenir el sobreajuste.
- Conjunto de prueba: destinado a medir la capacidad de generalización del modelo mediante métricas como la matriz de confusión.

Esta estrategia permite verificar la capacidad del sistema para identificar correctamente escenarios de riesgo en imágenes nuevas, pero coherentes con las condiciones del conjunto original.

Para realizar esta evaluación, se generó una nueva base de datos asegurando que la partición de datos estuviera balanceada, es decir, que tanto el conjunto de entrenamiento (80%) como el de prueba (20%) contaran con una distribución equitativa de fotogramas por clase. Esta estrategia busca evitar que el modelo aprenda de forma sesgada, favoreciendo una clase sobre otra o que no logre aprender adecuadamente si alguna clase está subrepresentada. Para lograr una distribución equilibrada, se implementó un método en Python que selecciona fotogramas aleatorios de cada clase de manera proporcional y balanceada. Una vez realizada esta partición, se procedió a reentrenar el modelo utilizando esta nueva base de datos.

Los resultados del rendimiento del modelo tras esta evaluación pueden visualizarse en la Figura 5 y se describen a continuación:

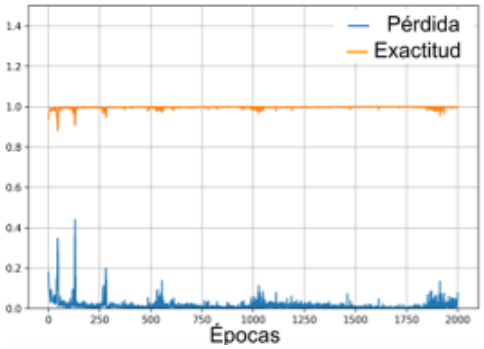
- Clase “No riesgo”: 1.0000.
- Clase “Amenaza”: 0.9972.
- Clase “Asalto con arma”: 0.9972.
- Clase “Asalto sin arma”: 0.9813.



Fuente: elaboración propia

Figura 5 Matriz de confusión obtenida por el método de resustitución.

Estos resultados indican que el modelo presenta un alto nivel de precisión en la clasificación de las diferentes situaciones de riesgo. En particular, se evidencia un desempeño óptimo en todas las clases “No riesgo”, “Amenaza”, “Asalto con arma” y “Asalto sin arma”. En conjunto, estos valores permiten concluir que el modelo es capaz de identificar y clasificar adecuadamente escenarios de riesgo en entornos de transporte público. La Figura 6 ilustra la evolución conjunta de dos métricas clave del proceso de entrenamiento del modelo: la función de pérdida y la exactitud, a lo largo de un total de 2000 épocas obtenidas por el método de resustitución.



Fuente: elaboración propia.

Figura 6 Métricas de calidad obtenidas por el método de resustitución.

Durante las primeras 600 épocas, se observa un comportamiento inestable en la función de pérdida, con varios picos pronunciados que alcanzan valores superiores a 0.5. Este comportamiento sugiere que el modelo atravesó una fase inicial de

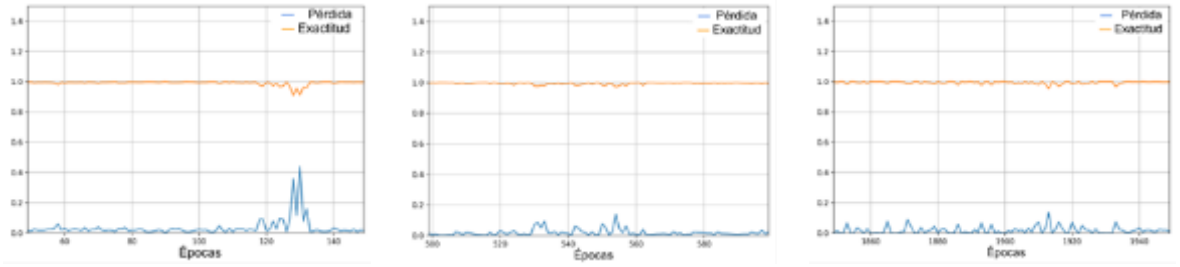
dificultad para ajustar sus pesos de manera eficiente, posiblemente debido a una etapa de exploración amplia dentro del espacio de parámetros.

A partir de la época 600, los valores de pérdida comienzan a estabilizarse en torno a niveles muy bajos, cercanos a cero, aunque aún se presentan fluctuaciones esporádicas. Estos picos residuales pueden atribuirse a reconfiguraciones internas en los pesos del modelo que, sin embargo, no comprometen su capacidad de generalización.

En cuanto a la precisión, esta se mantiene elevada desde las primeras épocas, con valores cercanos a 1.0, lo cual indica que el modelo logró aprender patrones significativos del conjunto de datos desde etapas tempranas. A pesar de que se evidencian algunas caídas puntuales, especialmente alrededor de las épocas 100, 500 y 1900, estas no afectan de forma crítica el rendimiento global, ya que el modelo se recupera rápidamente y retoma valores por encima de 0.997.

Este comportamiento conjunto refleja un modelo robusto y estable, sin indicios de sobreajuste severo (*overfitting*), capaz de mantener un alto rendimiento incluso frente a pequeñas perturbaciones en la optimización.

En la Figura 7 se observa con mayor facilidad las caídas puntuales del entrenamiento alrededor de las épocas 100, 500 y 1900. Por esta razón se recomienda considerar los modelos almacenados en las épocas 600, 1000 y 1500, ya que presentan un equilibrio favorable entre precisión constante y pérdidas mínimas. Estas versiones son idóneas para su implementación en aplicaciones en tiempo real, validación cruzada en nuevos entornos, o como punto de partida para técnicas de aprendizaje por transferencia o ajustes posteriores.



a) Época 100. *b) Época 500.* *c) Época 1900.*

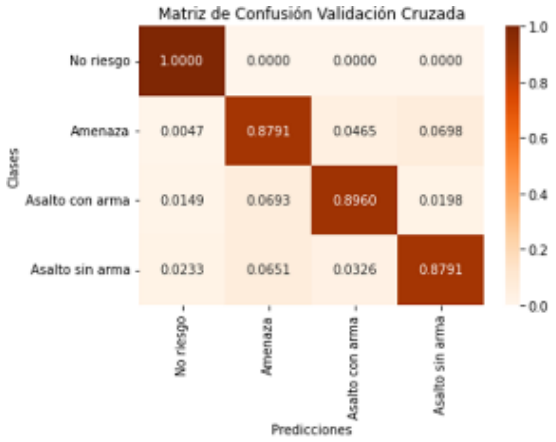
Fuente: elaboración propia.

Figura 7 Métricas de calidad del modelo en el método de resustitución.

En la Figura 8 se muestra la matriz de confusión obtenida al entrenar el modelo por medio del método de validación cruzada, los resultados son los siguientes:

- Clase “No riesgo”. Se clasifica correctamente el 100% de las veces (1.0 en la diagonal principal), sin confusiones con otras categorías.
- Clase “Amenaza”. Alto acierto (87.91%) en su identificación. El modelo presenta confusión en la clasificación como “Asalto con arma” (4.65%) y en clasificación como “Asalto sin arma” (6.98%).
- Clase “Asalto con arma”. Buen desempeño con un 89.60% de acierto. Confusiones ocasionales con “Amenaza” (6.93%).
- Clase “Asalto sin arma”. Precisión de 87.91%. Confusiones con “Amenaza” (6.51%).

El modelo presenta excelente precisión para la clase “No riesgo” y un buen desempeño para el resto de las categorías, aunque con algunas confusiones entre los tipos de riesgo (“Amenaza”, “Asalto con arma” y “Asalto sin arma”). Esto es común cuando las clases comparten características visuales o contextuales, y podría mejorarse ampliando la diversidad de ejemplos en el entrenamiento.

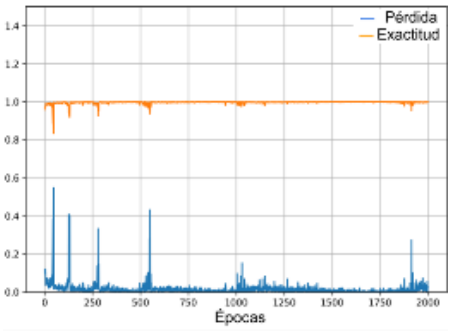


Fuente: elaboración propia.

Figura 8 Matriz de confusión obtenida por el método de validación cruzada.

La Figura 9 muestra la evolución de la pérdida y la exactitud entrenada con el método de validación cruzada a lo largo de 2000 épocas de entrenamiento. Se observa que la exactitud se mantiene muy cercana al 100% durante casi todo el

entrenamiento, con ligeras caídas puntuales y la pérdida es muy baja en general, con picos aislados en algunas épocas; el gráfico indica que el modelo alcanza rápidamente un alto rendimiento y lo mantiene estable a lo largo del entrenamiento.



Fuente: elaboración propia.

Figura 9 Métricas de calidad obtenidas por el método de validación cruzada.

4. Discusión

El modelo fue entrenado utilizando una base de datos compuesta por fotogramas capturados en entornos de transporte público, lo que le ha permitido especializarse en la identificación y clasificación de escenarios propios de este contexto. Esta focalización ha favorecido un alto nivel de exactitud y eficiencia dentro del dominio de aplicación, posicionándolo como una herramienta robusta para entornos similares. Si bien el modelo muestra un desempeño sobresaliente en el ámbito para el que fue diseñado, una línea de trabajo prometedora consiste en ampliar la base de datos con imágenes provenientes de entornos más variados. Esto podría potenciar su capacidad de generalización, permitiéndole adaptarse a contextos operativos diversos y consolidando aún más su versatilidad y confiabilidad. Esta mejora se propone como un paso futuro para ampliar el impacto del sistema. La implementación del modelo en vehículos de transporte público constituye una solución tecnológica funcional y de alto valor, capaz de detectar y clasificar escenarios de riesgo en tiempo real. Esta capacidad facilita la toma de decisiones automatizadas o asistidas, orientadas a reforzar la seguridad de los usuarios. Una detección temprana y precisa de situaciones potencialmente peligrosas puede ser determinante para prevenir incidentes y salvaguardar la integridad de la ciudadanía, contribuyendo de manera directa a entornos de transporte más seguros.

5. Conclusiones

El presente proyecto logró exitosamente su objetivo principal: desarrollar un sistema de clasificación de escenas de riesgo en transporte público utilizando redes neuronales convolucionales.

El modelo diseñado alcanzó altos niveles de precisión en la detección de situaciones peligrosas, destacando una exactitud del 100% en escenas sin riesgo, del 99% en amenazas visibles y del 98% en asaltos donde el arma no es perceptible. Este sistema demuestra ser una herramienta efectiva para su implementación en tiempo real dentro de unidades de transporte público, lo que permitiría activar protocolos preventivos ante situaciones sospechosas o peligrosas, contribuyendo así a la seguridad de los usuarios.

Una de las posibles aplicaciones prácticas de este sistema consiste en su implementación dentro de un dispositivo físico, como un módulo embebido o una cámara inteligente, capaz de ejecutar el modelo y realizar predicciones de forma autónoma. Esta integración permitiría la detección automática de situaciones de riesgo mientras se graban las escenas, facilitando respuestas oportunas para mejorar la seguridad de los pasajeros; sin embargo, se reconoce que el modelo presenta limitaciones en su capacidad de generalización, dado que fue entrenado con datos específicos de un entorno particular. Por ello, una de las mejoras propuestas consiste en ampliar la base de datos con escenas de diferentes contextos, lo que permitiría aumentar la robustez y adaptabilidad del sistema.

En resumen, este trabajo constituye un avance significativo en la aplicación de inteligencia artificial para la seguridad pública, con potencial de desarrollo hacia sistemas más versátiles y generalizables mediante futuras actualizaciones.

6. Referencias y Bibliografía

[1] Brownlee, J. Better Deep Learning: Train Faster, Reduce Overfitting, and Make Better Predictions. Machine Learning Mastery, Vermont, U.S.A. December 2019.

[2] Facebook. Meta Platforms Inc. Servicio de Redes Sociales y Medios Sociales en Línea, Menlo Park, California, Estados Unidos. Febrero 2004.

[3] Géron, A. Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems. O'Reilly Media, tercera edición, Sebastopol, California. Septiembre 2022.

[4] Nadeem, M. S, Kurugollu, F, Atlam, H. F, Franqueira, V. N. Weapon Violence Dataset 2.0: A Synthetic Dataset for Violence Detection. Data in Brief, Vol. 54, 110448, Ámsterdam, Países Bajos. June 2024.

[5] Parseh, M. J, Rahmanimanesh, M, Keshavarzi, P, Kasaei, S. H. Semantic-Aware Visual Scene Representation. International Journal of Multimedia Information Retrieval, Vol. 11, No. 4, pp. 619–638, Berlin, Alemania. December 2022.

[6] Surendran, R, J, A, Hemanth, J. D. Recognition of Human Action for Scene Understanding Using World Cup Optimization and Transfer Learning Approach. PeerJ Computer Science, Vol. 9, e1396, Londres, Reino Unido. Abril 2023.

[7] Yeo, W. H, Heo, Y. J, Choi, Y. J, Kim, B. G. Place Classification Algorithm Based on Semantic Segmented Objects. Applied Sciences, Vol. 10, No. 24, 9069, Basilea, Suiza. December 2020.

[8] YouTube. Google LLC. Plataforma para Compartir Videos y Red Social, San Bruno, California, U. S. A, 2005.