

UNIVERSIDAD TÉCNICA DE AMBATO

FACULTAD DE INGENIERÍA EN SISTEMAS, ELECTRÓNICA E INDUSTRIAL

CARRERA DE TELECOMUNICACIONES

Tema:

SISTEMA DE MONITOREO Y DETECCIÓN DE COMPORTAMIENTO DELINCUENCIAL APLICADO EN MEDIOS DE TRANSPORTE UTILIZANDO VISIÓN ARTIFICIAL

Trabajo de titulación modalidad Proyecto de Investigación, presentado previo a la obtención del título de Ingeniero en Telecomunicaciones

ÁREA: Comunicaciones

LÍNEA DE INVESTIGACIÓN: Electrónica, Telecomunicaciones y Sistemas de control

AUTOR: Jonathan Ronaldo Abril San Lucas

TUTOR: Ing. Jesús Israel Guamán Molina, Mg.

Ambato - Ecuador

julio – 2025

APROBACIÓN DEL TUTOR

En calidad de tutor del trabajo de titulación con el tema: SISTEMA DE MONITOREO Y DETECCIÓN DE COMPORTAMIENTO DELINCUENCIAL APLICADO EN MEDIOS DE TRANSPORTE UTILIZANDO VISIÓN ARTIFICIAL, desarrollado bajo la modalidad Proyecto de Investigación por el señor Jonathan Ronaldo Abril San Lucas, estudiante de la Carrera de Telecomunicaciones, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, de la Universidad Técnica de Ambato, me permito indicar que el estudiante ha sido tutorado durante todo el desarrollo del trabajo hasta su conclusión, de acuerdo a lo dispuesto en el Artículo 17 del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato y el numeral 6.3 del instructivo del reglamento referido.

Ambato, julio 2025.

Firmado electrónicamente por:
JESUS ISRAEL GUAMAN MOLINA
Validar únicamente con FirmaEC

Ing. Jesús Israel Guamán Molina, Mg.

TUTOR

AUTORÍA

El presente trabajo de titulación con el tema: SISTEMA DE MONITOREO Y DETECCIÓN DE COMPORTAMIENTO DELINCUENCIAL APLICADO EN MEDIOS DE TRANSPORTE UTILIZANDO VISIÓN ARTIFICIAL es absolutamente original, auténtico y personal y ha observado los preceptos establecidos en la Disposición General Quinta del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato. En tal virtud, el contenido, efectos legales y académicos que se desprenden del mismo son de exclusiva responsabilidad del autor.

Ambato, julio 2025



Jonathan Ronaldo Abril San Lucas

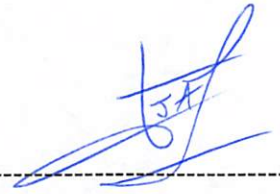
C.C. 1850136779

AUTOR

DERECHOS DE AUTOR

Autorizo a la Universidad Técnica de Ambato para que reproduzca total o parcialmente este trabajo de titulación dentro de las regulaciones legales e institucionales correspondientes. Además, cedo todos mis derechos de autor a favor de la institución con el propósito de su difusión pública, por lo tanto, autorizo su publicación en el repositorio virtual institucional como un documento disponible para la lectura y uso con fines académicos e investigativos de acuerdo con la Disposición General Cuarta del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato.

Ambato, julio 2025.



Jonathan Ronaldo Abril San Lucas

C.C. 1850136779

AUTOR

APROBACIÓN DEL TRIBUNAL DE GRADO

En calidad de par calificador del informe final del trabajo de titulación presentado por el señor Jonathan Ronaldo Abril San Lucas, estudiante de la Carrera de Telecomunicaciones, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, bajo la Modalidad Proyecto de Investigación, titulado SISTEMA DE MONITOREO Y DETECCIÓN DE COMPORTAMIENTO DELINCUENCIAL APLICADO EN MEDIOS DE TRANSPORTE UTILIZANDO VISIÓN ARTIFICIAL, nos permitimos informar que el trabajo ha sido revisado y calificado de acuerdo al Artículo 19 del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato y el numeral 6.4 del instructivo del reglamento referido. Para cuya constancia suscribimos, conjuntamente con el señor Presidente del Tribunal.

Ambato, julio 2025.

Firmado electrónicamente por:
FRANKLIN OSWALDO MAYORGA MAYORGA
Validar únicamente con FirmaEC

Ing. Franklin Oswaldo Mayorga Mayorga, Mg.

PRESIDENTE DEL TRIBUNAL

Firmado electrónicamente por:
FABIAN RODRIGO SALAZAR ESCOBAR
Validar únicamente con FirmaEC

Ing. Fabián Salazar Escobar, Mg.

PROFESOR CALIFICADOR

Firmado electrónicamente por:
GEOVANNI DANILO BRITO MONCAYO
Validar únicamente con FirmaEC

Ing. Geovanni Brito Moncayo, Mg.

PROFESOR CALIFICADOR

DEDICATORIA

A Dios, por ser mi ayuda idónea y mi guía constante dentro de este largo camino.

A mis queridos padres; Darwin y Jazmín, a mi amado hermano Josué, a ellos por ser los hombres y la mujer que yo más admiro en la vida, por ser mi fuente de inspiración y motivación principal, quienes siempre estuvieron conmigo en los buenos y malos momentos. Por eso y más este triunfo es de ustedes también.

A mis amigos de la carrera; los perrines: Kevin, Paúl, Erick Lascano, Erick Bonilla y Cris, con quienes compartí múltiples experiencias dentro y fuera de un aula de clases e hicieron que esta aventura sea inolvidable.

A mis amigos del colegio y de la vida; Ismael, Sebastián, Jimmi, Morita, César Ortiz, Andrés, y Darío; el eterno presidente y líder del club de robótica de quien he aprendido bastante. Todos ellos no solo son mis amigos, sino también mis hermanos.

A mi prima; la Dra. Carolina Lara quien que a pesar de no ser una profesional que pertenezca a mi área de conocimiento, ella siempre fue un ejemplo de superación y constancia para mí, y como resultado de lo recientemente mencionado, ella se convirtió en una fuente de inspiración dentro de mi formación profesional.

A mi abuelo; el Ing. Víctor Mejía, fuente principal de superación y admiración dentro del área, ya que, para mí, él es el mejor electrónico. Él un día me dijo "Hijo sigue electrónica, es complicado, pero tú puedes".

AGRADECIMIENTO

Agradezco a Dios, por siempre colocarme en el lugar correcto, junto con las personas correctas. Por brindarme la vida, la sabiduría y la fortaleza para afrontar cada situación que se me presenta en la vida.

A mis padres; quienes siempre han sido ese impulso de apoyo económico y moral en cada proyecto que me he propuesto realizar en mi vida.

A mi gran Amigo Kevin, por haber estado en los momentos más complicados de la carrera, por haber sido un amigo incondicional en todo momento.

Estoy muy agradecido con mi tutor y guía de tesis; el Ingeniero Jesús Guamán Molina, quien siempre puso a disposición sus conocimientos conmigo.

A mi abuelita; Nenita, quien gracias a Dios la tengo con vida, me apoya, me quiere bastante y sé que está muy orgullosa de mí.

Finalmente, y no menos importante, agradezco a cada uno de mis abuelos que están en el cielo porque su recuerdo me llenaba de esperanza e inspiración. Y sé que estarían muy orgullosos de mí si estuvieran con vida. Mi abuelo Medardo, Evaristo, Lydice, y mi abuelita Alicia; quien siempre me apoyó y estuvo muy atenta a lo que yo necesitaba, demostrando siempre su corazón de madre; con profundo amor y empatía. Quién hasta el día de hoy, a pesar de haber pasado ya 5 años de su partida, aún la recuerdo y me quebranto al hablar de ella.

Jonathan Ronaldo Abril San Lucas.

ÍNDICE GENERAL DE CONTENIDOS

APROBACIÓN DEL TUTOR ..... **ii**

AUTORÍA ..... **iii**

DERECHOS DE AUTOR ..... **iv**

APROBACIÓN DEL TRIBUNAL DE GRADO ..... **v**

DEDICATORIA ..... **vi**

ÍNDICE GENERAL DE CONTENIDOS ..... **viii**

ÍNDICE DE TABLAS ..... **xii**

ÍNDICE DE FIGURAS ..... **xiv**

ÍNDICE DE ANEXOS ..... **xxii**

RESUMEN EJECUTIVO ..... **xxiii**

ABSTRACT ..... **xxiv**

CAPÍTULO I. MARCO TEÓRICO **1**

1.1 Tema de investigación 1

1.1.1 Planteamiento del problema 1

1.2 Antecedentes investigativos 3

1.3 Fundamentación teórica 6

1.3.1 Sistemas de Seguridad..... 7

1.3.2 Sistemas de seguridad en medios de transporte 7

1.3.3 Sistemas de seguridad Privada 10

1.3.4 Sistemas electrónicos de seguridad 10

1.3.5 Sistemas de monitoreo por video 12

1.3.6 Comportamiento Delincuencial en Transporte Público. 19

1.3.7 Tipos de comportamiento delincuencial en medios de transporte público. 20

1.3.8 Machine Learning. 21

1.3.9 Deep Learning 48

1.3.10 Visión Artificial. 55

1.3.11 Control de la información 66

1.3.12 Módulos GPS para Sistemas de Navegación y Geolocalización 68

1.4 Objetivos 72

1.4.1 Objetivo general 72

1.4.2 Objetivos específicos 72

CAPÍTULO II. METODOLOGÍA 73

2.1 Materiales 73

2.2 Métodos 74

2.2.1 Modalidad de la investigación 74

2.2.2 Población y muestra 74

2.2.3 Recolección de información 74

2.2.4 Procesamiento y análisis de datos 75

CAPÍTULO III. RESULTADOS Y DISCUSIÓN 77

3.1 Estado Actual 77

3.1.1 Estructura general 82

3.1.2 Selección del hardware 84

3.1.3 Especificaciones de la propuesta 94

3.2 Construcción del Banco de datos 96

3.2.1 Detección de objetos amenazantes 96

3.2.2 Identificación de posturas corporales sospechosas 100

3.2.3 Obtención de datos de geoposicionamiento (latitud y longitud) con tecnología GPS 101

3.3 Creación del modelo 104

3.3.1 Fase de entrenamiento y validación del modelo 105

3.4.2 Plataforma de desarrollo para el control de la información 118

3.4 Diseño y Montaje de componentes dentro del sistema. 127

3.4.1 Diseño Estructural del Prototipo del Sistema. 127

3.4.2 Montaje de los componentes sobre la carcasa impresa en 3D. 131

3.5 Verificación de funcionamiento del sistema 135

3.5.1 Evaluaciones en condiciones reales del sistema de detección de objetos y posturas corporales. 136

3.5.2 Evaluaciones en condiciones reales del sistema GPS. 139

3.5.3 Pruebas de funcionamiento con el sistema de detección y monitoreo completo. 143

3.5.4 Parámetros de validación. 146

3.5.5 Validación de los resultados 148

3.6 Despliegue 153

3.6.1 Implementación del prototipo 154

3.6.2 Estructura operativa. 157

3.6.3 Requisitos mínimos de hardware y software para la implementación. 158

3.6.4 Presupuesto 159

CAPÍTULO IV. CONCLUSIONES Y RECOMENDACIONES 162

4.1 Conclusiones 162

4.2 Recomendaciones 164

REFERENCIAS BIBLIOGRÁFICAS 166

ANEXOS 171

ÍNDICE DE TABLAS

Tabla 1. Tecnologías actuales de la capa física del automóvil. 8

Tabla 2. Tipos de Dispositivos de Captura 12

Tabla 3. Características de los dispositivos de captura más utilizados. 13

Tabla 4. Características técnicas de varias plataformas cloud de entrenamiento para machine Learning. 40

Tabla 5. Análisis comparativo de enfoques para proyectos de Machine Learning. ... 45

Tabla 6. Descripciones de las versiones de Yolo más importantes 59

Tabla 7. Sistemas de identificación de objetos. 62

Tabla 8. Análisis comparativo de las distintas unidades de procesamiento de datos presentadas. 65

Tabla 9. Análisis comparativo de las distintas Plataformas de desarrollo para el control de la información. 67

Tabla 10. Análisis comparativo de los distintos tipos de módulos GPS para sistemas de navegación y geolocalización. 71

Tabla 11. Criterios de recolección de información. 75

Tabla 12. Comparativa entre distintas Cámaras Comerciales 88

Tabla 13. Especificaciones técnicas de los tres tipos de cámaras analizadas. 90

Tabla 14. Especificaciones técnicas de los dos microcontroladores analizados. 91

Tabla 15. Especificaciones técnicas de las Unidades de procesamiento analizadas .. 94

Tabla 16. Comparativa entre las versiones de los tipos de modelos Yolo versión 8 y 11. 106

Tabla 17. Parámetros que se tomaron en cuenta para cada uno de los tamaños de Yolo versión 8 y 11. 107

Tabla 18. Resultados de cada uno de los entrenamientos de Yolo realizados. 107

Tabla 19. Párametros de evaluación del modelo Yolov8 m con 40 épocas. 112

Tabla 20. Matriz de confusión de armas blancas 146

Tabla 21. Matriz de confusión de armas de fuego. 147

Tabla 22. Matriz de confusión de la detección de gorras 147

Tabla 23. Matriz de confusión de la Detección de Posturas Sospechosas. 148

Tabla 24. Parámetros de validación de cada una de las clases 152

Tabla 25. Requisitos mínimos de hardware y software para la implementación. 158

Tabla 26. Costos de implementación 160

ÍNDICE DE FIGURAS

Figura 1. Dominio Intravehicular 8

Figura 2. Dominio Intervehicular y Análisis de datos 9

Figura 3. Esquema Básico CCTV Análogo. 17

Figura 4. Esquema Básico CCTV IP 18

Figura 5. Disciplinas en las que se apoya Machine Learning 22

Figura 6. Aprendizaje supervisado. Fuente: Elaboración propia. 25

Figura 7. Aprendizaje no supervisado. Fuente: Elaboración propia. 26

Figura 8. Conjunto de datos sin valores asignados. 26

Figura 9. Conjunto de datos divididos en cuatro grupos. 27

Figura 10. Tipos de aprendizaje automático. 28

Figura 11. Metodología para Machine Learning. 28

Figura 12. Predicciones del modelo del vecino más próximo en el conjunto de datos de forja. 30

Figura 13. Ejemplo de un modelo de regresión lineal 31

Figura 14. Un árbol de decisión para distinguir entre varios animales. 33

Figura 15. Máquina de soporte vectorial 34

Figura 16. Clasificación binaria con SVM 34

Figura 17. Esquema Perceptrón básico 35

Figura 18. Perceptrón multicapa con una sola capa oculta. 36

Figura 19. Gráfico de la función seno generado en Google Colab, con etiquetado y leyenda añadidos para mayor claridad 37

Figura 20. Creación de entornos seguros de aprendizaje automático con Amazon SageMake. 38

Figura 21. Integración de Microsoft Azure Machine Learning. 39

Figura 22. Flujo de trabajo propuesto por la metodología CRISP-DM. 41

Figura 23. Flujo de trabajo de la metodología SEMMA. 44

Figura 24. Diagrama de confusión de un modelo de clasificación binaria. 47

Figura 25. Desarrollo de una red neuronal convolucional. Capa de partida (imagen inicial, o entrada). 50

Figura 26. Esquema de un pixel. 50

Figura 27. Preprocesamiento. 51

Figura 28. Convolución 52

Figura 29. Función del Kernel: La imagen que se genera se reduce con rasgos que predominan en comparación con la imagen de origen. 52

Figura 30. Imagen convolucionada del ejemplo anterior con un tamaño de pool o filtro 53

Figura 31. Esquema de entrenamiento de una red neuronal convolucional 53

Figura 32. Esquema general y resumido del funcionamiento de un modelo CNN con tres capas convolucionales. 54

Figura 33. Etapas a considerar dentro de un proceso de visión artificial. 55

Figura 34. Detección de objetos con YOLO. 57

Figura 35. El funcionamiento de YOLO para la identificación de objetos. 58

Figura 36. Comparación del rendimiento de diferentes versiones de YOLO en términos de mAP (precisión promedio) y latencia en milisegundos por imagen en el conjunto de datos COCO..... 59

Figura 37. Esquema de una Red de Propuestas de Regiones (RPN) para detección de objetos. 60

Figura 38. Proceso de generación de propuestas de regiones con RPN 61

Figura 39. Predicción de cuadros en SSD: (a) Imagen con GT, (b) Mapa 8×8, (c) Mapa 4×4. 62

Figura 40. Placa electrónica Rasberry Pi 4 63

Figura 41. Placa electrónica Banana Pi M2 Zero..... 64

Figura 42. Placa electrónica NVIDIA Jetson Nano 65

Figura 43. Módulo GPS u-blox 6..... 68

Figura 44. Módulo Ublox ZED-F9P 70

Figura 45. Sistemas de seguridad más utilizados entre los encuestados..... 78

Figura 46. Frecuencia de Incidentes Delictivos en Medios de Transporte. 79

Figura 47. Principales Deficiencias o Puntos Vulnerables de los Sistemas de Seguridad Actuales..... 80

Figura 48. Componentes del Sistema y Análisis de Efectividad. 81

Figura 49. Propuesta y Aceptación de un Sistema de Monitoreo Basado en Visión Artificial. 82

Figura 50. Estructura general del sistema propuesto. 83

Figura 51. Diagrama de campo de visión focal..... 86

Figura 52. Cámara Microsoft LifeCam HD-3000..... 89

Figura 53. Cámara Webcam Logitech C270. 89

Figura 54. Unidad de Procesamiento NVIDIA Jetson Orin Nano. 93

Figura 55. Etiquetado de la clase "gorras" y "cuchillo". 97

Figura 56. Etiquetado de la clase "armas". 98

Figura 57. Distribución del dataset para su descarga desde la plataforma Roboflow. 99

Figura 58. Estructura del dataset en formato YOLO. 99

Figura 59. Fotograma que recrea un comportamiento anómalo con arma de fuego para el etiquetado o clasificación de posturas corporales. 100

Figura 60. Código para el Procesamiento y transmisión de datos con Arduino Uno 102

Figura 61. Interfaz gráfica de Node-red. 103

Figura 62. Configuraciones básica del puerto serial. 103

Figura 63. Configuraciones básicas del protocolo MQTT. 104

Figura 64. Esquema general de entrenamiento del sistema para los modelos (YOLO y SVM). 105

Figura 65. Resultados del modelo escogido; Yolo versión 8 medium con 60 épocas. 108

Figura 66. Matriz de confusión del modelo seleccionado. 108

Figura 67. Curva de precisión que se maneja en función del nivel de confianza dedicada hacia tres clases específicas: armas, cuchillos y gorras. 109

Figura 68. Curva Recall-confianza, misma que representa la relación entre el Recall (la tasa de verdaderos positivos) y los distintos niveles de confianza para las clases evaluadas. 110

Figura 69. Correlación entre el F1-score y diversos valores de confianza 111

Figura 70. Resultados del modelo escogido; Yolo versión 8 medium con 40 épocas. 112

Figura 71. Matriz de confusión. 113

Figura 72. Curva de precisión que es manejada en función del nivel de confianza, dirigida hacia tres clases muy específicas, aquellos son; armas, cuchillo y gorras. 114

Figura 73. Curva de Precisión-Confianza, y aquella es una herramienta fundamental e importante para la evaluación del rendimiento dentro de un modelo de detección de objetos. 115

Figura 74. Curva F1-Confianza, donde evalúa el equilibrio o la relación entre la precisión y el recall del modelo a diferentes umbrales de confianza. 116

Figura 75. Curvas ROC para cada clase que forman parte del entrenamiento. 117

Figura 76. Matriz de confusión que se encarga de evaluar el rendimiento de un modelo de clasificación para los tres tipos de clases "AsaltoArma", "AsaltoCuchillo" y "Normal". 118

Figura 77. Paquetes instalados dentro del entorno virtual “venv”. 119

Figura 78. Tráfico de información por el puerto 5000. 119

Figura 79. Tráfico de información por el puerto 1883. 119

Figura 80. Diagrama entidad-relación de la base de datos. 123

Figura 81. Función con la que se lleva a cabo los registros de las detecciones y los datos del módulo GPS cuando existe una detección. 123

Figura 82. Interfaz gráfica de la autenticación de los usuarios. 124

Figura 83. Funcionamiento del sistema dentro del archivo “index.html”. 125

Figura 84. Interfaz gráfica donde se registran las detecciones. 125

Figura 85. Alerta difundida por Slack 127

Figura 86. Base de la carcasa ya extruida, bordeada, y con las debidas aberturas para el flujo de aire, dentro del entorno de Fusion 360. 129

Figura 87. Espacio donde se ajustó el microcontrolador Arduino UNO, junto con el módulo GPS. 129

Figura 88. Compartimento dedicado para la cámara web 130

Figura 89. Diseño estructural final de la carcasa. 130

Figura 90. Carcasa recién impresa, lista para el montaje de cada uno de los componentes mencionados. 131

Figura 91. Componentes ya colocados dentro de la estructura 3D impresa. 132

Figura 92. Compartimento dedicado para la webcam HD (720p). 132

Figura 93. Sistema embebido colocado en su respectivo compartimento. 133

Figura 94. Antena del módulo GPS colocado en la parte externa de la estructura impresa. 134

Figura 95. Unidad de procesamiento NVIDIA Jetson Orin Nano colocada en la base de la estructura. 134

Figura 96. Conexión al sistema de monitoreo por medio de una pantalla LCD. 135

Figura 97. Alerta temprana de una de las pruebas efectuadas. 136

Figura 98. Delito flagrante con gorra y cuchillo. 137

Figura 99. Delito flagrante con gorra y arma. 137

Figura 100. Ubicación marcada en Google Maps del delito flagrante. 138

Figura 101. Delito flagrante con arma. 138

Figura 102. Delito flagrante con cuchillo. 139

Figura 103. Función que crea el tracking a partir del archivo .txt. 140

Figura 104. Inicio de la Ruta 1. 140

Figura 105. Trayecto total de la ruta de prueba 1. 141

Figura 106. Inicio de la Ruta 1 142

Figura 107. Trayecto total de la ruta de prueba 2. 143

Figura 108. Interfaz gráfica del sistema de monitoreo y detección delincuencial. .. 144

Figura 109. Difusión de la alerta de detección de la prueba funcionamiento. 144

Figura 110. Difusión de la alerta de detección de un arma de fuego. 145

Figura 111. Alerta de otra prueba de funcionamiento del sistema 145

Figura 112. Difusión de la alerta de detección de un arma de fuego dentro de otro entorno 146

Figura 113. Funcionamiento del sistemas e integración con la aplicación de difusión Slack. 153

Figura 114. Colocación del prototipo en la parte trasera de la gaveta del vehículo en el que se implementó el sistema 154

Figura 115. Colocación de la webcam en la esquina izquierda del parabrisas del vehículo. 155

Figura 116. Salida de 12V para la alimentación de la Unidad de Procesamiento Nvidia Jetson Nano. 155

Figura 117. Ingreso al sistema de monitoreo desde un ordenador, para la toma de pruebas de cobertura del módulo GPS. 156

Figura 118. Integración total del sistema dentro de la interfaz gráfica final 157

Figura 119. Estructura operativa del sistema mediante un esquema de conexiones de los equipos utilizados. 157

ÍNDICE DE ANEXOS

Anexo A. Categorización de variables. 171

Anexo B. Pruebas del sistema. 172

Anexo C. Especificaciones técnicas de la NVIDIA Jetson Orin 8 GB. 174

Anexo D. Código del sistema. 174

Anexo E. Especificaciones técnicas de la webcam HD (720p) de Sinovisión. 175

RESUMEN EJECUTIVO

Hoy en día, los distintos sistemas de videovigilancia dentro de los medios de transporte son dependientes en gran parte de la monitorización y del control humano, y como consecuencia de ello, los gastos asumidos por los propietarios de los medios de transporte son elevados para llevar a cabo este objetivo. De cierta forma, se ha realizado la propuesta de un sistema de monitoreo y detección de comportamiento delincuencial aplicado en medios de transporte utilizando visión artificial, La implementación del sistema se la realiza dentro de una NVIDIA Jetson Orin Nano con una webcam de 720p y un módulo GPS Neo 6m para georreferenciación. El sistema utiliza librerías como OpenCV y MediaPipe para el procesamiento de video y la estimación de posturas corporales, integrando modelos de IA (YOLO y SVM) para la detección de objetos sospechosos y de posturas amenazantes. Los eventos se clasifican como "Normal", "Sospechoso" o "Delito Flagrante" basándose en umbrales combinados. Las alertas, que contienen capturas y coordenadas, se envían en tiempo real a Slack y se gestionan desde una interfaz web Flask. Absolutamente todas las detecciones se guardan dentro de una base de datos SQLite para trazabilidad y futuras mejoras. Este sistema logra automatizar el monitoreo de seguridad, mejorando la capacidad de respuesta y prevención del delito.

Palabras clave: Visión artificial, detección de delitos, Jetson Orin Nano, YOLO, estimación de postura, seguridad en transporte, alertas en tiempo real, GPS.

ABSTRACT

Nowadays, the different video surveillance systems within the means of transport are largely dependent on monitoring and human control, and as a result, the costs incurred by the owners of the means of transport are high to carry out this objective. In a way, a proposal has been made for a monitoring and detection system of delinquent behavior applied to means of transport using artificial vision. The system is implemented inside an NVIDIA Jetson Orin Nano with a 720p webcam and a Neo 6m GPS module for georeferencing. The system uses libraries such as OpenCV and MediaPipe for video processing and body posture estimation, integrating AI models (YOLO and SVM) for the detection of suspicious objects and threatening postures. Events are classified as "Normal", 'Suspicious' or "Flagrant Crime" based on combined thresholds. Alerts, containing captures and coordinates, are sent in real time to Slack and managed from a Flask web interface. Absolutely all detections are stored within a SQLite database for traceability and future enhancements. This system automates security monitoring, improving responsiveness and crime prevention.

Keywords: Machine vision, crime detection, Jetson Orin Nano, YOLO, posture estimation, transportation security, real time alerts, GPS.

CAPÍTULO I. MARCO TEÓRICO

1.1 Tema de investigación

SISTEMA DE MONITOREO Y DETECCIÓN DE COMPORTAMIENTO DELINCUENCIAL APLICADO EN MEDIOS DE TRANSPORTE UTILIZANDO VISIÓN ARTIFICIAL

1.1.1 Planteamiento del problema

América Latina enfrenta altos índices de criminalidad, especialmente en espacios públicos y sistemas de transporte, donde la violencia y el crimen organizado son frecuentes. La región concentra el 37% de los homicidios a nivel mundial, pese a representar solo el 8% de la población global. Los robos en medios de transporte son comunes y afectan tanto a pasajeros como conductores. A esto se suma la falta de tecnologías efectivas para monitorear y prevenir estos delitos. Países como México y Brasil, que enfrentan altos niveles de criminalidad, han implementado tecnologías, pero su alcance es limitado y no lo suficientemente inclusivo o preventivo. Esto subraya la necesidad de soluciones innovadoras como sistemas basados en visión artificial para mejorar la seguridad en los transportes públicos [1].

Ecuador no está exento de estas problemáticas. En ciudades principales como Quito y Guayaquil, el transporte público ha sido escenario de constantes robos y agresiones. Según estudios, la percepción de inseguridad en transporte público supera el 70% de los usuarios, debido a la insuficiente presencia policial y sistemas de monitoreo ineficaces. Además, los presupuestos destinados a la seguridad pública suelen ser limitados, priorizando la reacción en lugar de la prevención. Aunque existen iniciativas de videovigilancia, estas no están integradas con tecnologías avanzadas como la visión artificial, que permitiría identificar patrones sospechosos y prevenir delitos en tiempo real [2].

En Ambato, los problemas asociados a la seguridad en los medios de transporte son particularmente perceptibles en las rutas interurbanas y áreas de gran afluencia de viajeros, tales como terminales y mercados. Las denuncias de hurtos, agresiones y

acoso sexual son comunes, y la ausencia de un sistema de vigilancia eficaz intensifica la percepción de inseguridad. Los sistemas de monitoreo contemporáneos son primitivos y no facilitan análisis automatizados, lo que restringe la habilidad para reaccionar ante incidentes. La implementación de un sistema de visión artificial podría ofrecer una respuesta eficaz, facilitando la identificación precoz de conductas inusuales y optimizando la eficacia de la implementación. protección de los usuarios [3].

Las urbes contemporáneas se encuentran con un problema cada vez mayor de inseguridad en el transporte público, lo que impacta en la movilidad y el bienestar de sus residentes. Este problema no solo provoca una sensación de riesgo en los usuarios, sino que también afecta de manera adversa el crecimiento social y económico de las urbes. La ausencia de sistemas de seguridad avanzados en los medios de transporte, unida al incremento de conductas delictivas, empeora esta circunstancia, transformando a los vehículos y estaciones en lugares de riesgo.

Uno de los elementos clave en la administración de la seguridad en el transporte público es la habilidad para supervisar, detectar y responder eficazmente ante circunstancias sospechosas o delictivas. No obstante, los sistemas convencionales, como la supervisión pasiva a través de cámaras, frecuentemente resultan insuficientes para evitar estos sucesos en tiempo real. Esto simplifica la realización de acciones delictivas, impactando tanto a los usuarios como al personal de transporte.

Un insuficiente sistema de seguridad no solo aumenta el peligro para los viajeros, sino que también impacta la confianza en el transporte público, lo que conduce a una reducción en su utilización. Esto, por consiguiente, impacta en la viabilidad financiera del sistema, puesto que la reducción de usuarios puede conducir a un ciclo perjudicial de menor inversión en seguridad y prestaciones.

La adopción de tecnologías vanguardistas, como los sistemas basados en inteligencia artificial, proporciona una respuesta eficaz para atenuar este problema. A través de la implementación de algoritmos sofisticados, estos sistemas tienen la capacidad de examinar en tiempo real las conductas en los vehículos y estaciones, identificar patrones vinculados a acciones delictivas y advertir al personal de seguridad. No solo

facilita la intervención oportuna, sino que también minimiza la comisión de delitos al incrementar la percepción de supervisión.

Es crucial resaltar que, a pesar de que no se puede eliminar completamente los comportamientos delictivos, un sistema basado en visión artificial, junto con un personal de seguridad formado, puede disminuir considerablemente la frecuencia de estos sucesos. Es fundamental combinar tecnologías avanzadas con estrategias preventivas y reactivas para convertir los medios de transporte público en lugares más seguros para la comunidad.

1.2 Antecedentes investigativos

Esta parte recopila estudios anteriores y antecedentes relevantes acerca del asunto "Sistema de vigilancia e identificación de conductas delictivas en medios de transporte mediante visión artificial". Principalmente se enfoca en investigaciones que utilizan la visión artificial como instrumento crucial para el estudio de comportamientos, basándose en referencias bibliográficas que proporcionan guía técnica y conceptual para el avance del proyecto. Además, se incluyen referencias de archivos académicos y bibliotecas en línea para apoyar la investigación.

En la investigación denominada "YOLO-SGC: A Dangerous Driving Behavior Detection Method With Multiscale Spatial-Channel Feature Aggregation", publicado en 2024 en el IEEE Sensors Journal, bajo la dirección de Ruijie Li, en colaboración con Changdong Yu, Xiangrong Qin, Xin An, Jinpeng Zhao, Wenhui Chuai y Baisheng Liu, elaboró un modelo sofisticado para la identificación de conductas riesgosas al manejar. El propósito principal de este estudio fue incrementar la seguridad en las vías mediante la implementación de un sistema exacto y eficaz que detecte en tiempo real patrones de conducción peligrosa. El método sugerido, conocido como YOLO-SGC, se fundamenta en el marco YOLOv8 y utiliza una técnica denominada Convolución de Reconstrucción Espacial y de Canal (SCConv) para mejorar la obtención de características multiescala, facilitando así un examen más exhaustivo de las imágenes de tráfico de vehículos. El modelo fue capacitado con grupos de datos específicos de conducción, aunque no se especifican en detalle, y se implementó un sistema con

capacidad de operación en tiempo real, consiguiendo una identificación más exacta y eficaz de conductas peligrosas en comparación con técnicas previas. Los hallazgos evidenciaron que el modelo no solo aumentó notablemente la exactitud en comparación con métodos anteriores, sino que también evidenció su eficacia en diversos contextos de vehículos, transformándolo en un instrumento alentador para la supervisión del tráfico y la seguridad en el transporte público [4].

En el estudio titulado "Moving Vehicle Detection with Real-Time Speed Estimation Using Deep Learning Techniques", se explora la detección de vehículos en movimiento y la estimación de su velocidad en tiempo real mediante el uso de técnicas de aprendizaje profundo. La investigación muestra cómo la integración de redes neuronales convolucionales mejora el análisis de imágenes en contextos dinámicos, brindando resultados instantáneos para aplicaciones de transporte inteligente. El enfoque propuesto emplea modelos avanzados de detección de objetos, como YOLOv8, que identifican y siguen vehículos en videos con mayor rapidez y precisión. La estimación de la velocidad se realiza calculando la distancia recorrida por los vehículos entre fotogramas consecutivos y dividiéndola por el tiempo transcurrido, lo que facilita una medición exacta de la velocidad en tiempo real. Los resultados obtenidos demuestran que el sistema puede operar de manera efectiva en tiempo real, detectando vehículos y calculando sus velocidades con alta precisión, lo que lo hace indispensable para aplicaciones en sistemas de transporte inteligentes y en la vigilancia del tráfico [5].

En el artículo titulado "Deep Reinforcement Learning for Intelligent Transportation Systems: A Survey", publicado en 2020 en el *IEEE Transactions on Intelligent Transportation Systems*, los autores Ammar Haydari y Yasin Yilmaz realizan una revisión detallada de las técnicas de aprendizaje profundo y aprendizaje por refuerzo aplicadas a los sistemas de transporte inteligente. El estudio analiza cómo se utilizan arquitecturas como las Redes Neuronales Convolucionales (CNN) y las Redes Neuronales Recurrentes (RNN) para la predicción y monitoreo de patrones complejos de tráfico y comportamiento humano en entornos vehiculares. Entre las aplicaciones destacadas se incluyen el control de semáforos y la conducción autónoma, donde estos modelos han mostrado avances significativos en la eficiencia y seguridad del transporte. Los resultados del estudio destacan la efectividad de estas tecnologías para

detectar eventos anómalos y mejorar la seguridad en los sistemas de transporte. Además, se abordan los desafíos y preguntas abiertas sobre el uso del aprendizaje por refuerzo profundo en estos sistemas, ofreciendo una guía útil para futuras investigaciones en este campo [6].

En el estudio titulado "Vision-Based System for Abnormal Behavior Detection and Recognition of Bus Passengers", presentado en la IEEE Intelligent Transportation Systems Conference (ITSC) en 2022, los autores Chun-Han Tseng y Huei Yung Lin proponen un sistema basado en visión por computadora para la detección y reconocimiento de comportamientos anómalos de los pasajeros en autobuses. El propósito principal de la investigación es mejorar la seguridad en el transporte público mediante la implementación de un sistema capaz de identificar comportamientos sospechosos en tiempo real, utilizando técnicas avanzadas de visión por computadora y aprendizaje profundo. Para lograr esto, desarrollaron una red neuronal convolucional 3D (3D-CNN) que analiza secuencias de video tomadas en autobuses. Los datos para entrenar y evaluar el modelo provienen de videos que representan diversas situaciones dentro de los autobuses, garantizando que el sistema pueda identificar una amplia variedad de comportamientos anómalos. El sistema procesa las secuencias en tiempo real, detectando comportamientos que se desvían de la norma, lo que permite una intervención oportuna en situaciones potencialmente peligrosas. Los resultados mostraron que el sistema es eficaz para identificar comportamientos sospechosos en tiempo real, contribuyendo a mejorar la seguridad en el transporte público con un enfoque avanzado de vigilancia y protección de los pasajeros. Este enfoque innovador utiliza visión por computadora y aprendizaje profundo para resolver los retos de seguridad en el transporte público [7].

En el estudio titulado "Vision-Based Real-time Human Malicious Behavior Detection", los autores Hajra Binte Naeem, Muhammad Usama y Muhammad Moazam Fraz presentan un sistema de vigilancia basado en visión artificial para detectar comportamientos maliciosos de los seres humanos en tiempo real. El objetivo principal de la investigación es desarrollar un sistema capaz de identificar comportamientos sospechosos o anómalos en tiempo real, mejorando así la seguridad en diversos entornos, incluido el transporte público. Para ello, los autores emplearon una arquitectura de redes neuronales convolucionales (CNN) para analizar secuencias

de video grabadas por cámaras de vigilancia. Los datos utilizados para entrenar y evaluar el modelo comprenden grabaciones que incluyen tanto actividades normales como anormales. El sistema procesa estas secuencias en tiempo real, identificando patrones de comportamiento que se alejan de la norma establecida. Los resultados obtenidos indican que este enfoque es efectivo para la detección de comportamientos maliciosos en tiempo real, lo que facilita una intervención temprana en situaciones peligrosas. Además, este sistema es particularmente útil para mejorar la seguridad en el transporte público, permitiendo una vigilancia más eficaz mediante cámaras en tiempo real y optimizando la respuesta ante comportamientos sospechosos. Este trabajo resalta el potencial de la visión artificial y el aprendizaje profundo para afrontar los desafíos de seguridad en espacios públicos, ofreciendo una herramienta avanzada para la protección y vigilancia [8].

1.3 Fundamentación teórica

Un sistema diseñado específicamente para el monitoreo y detección de comportamientos delincuenciales aplicado en medios de transporte y que utilice visión artificial, requiere como requisito principal el conocimiento profundo tanto de los sistemas de seguridad como también de las diversas técnicas de nivel avanzado del procesamiento de imágenes y de métodos de aprendizaje automático. Este sistema tiene como objetivo identificar y a la vez alertar de forma exacta y en tiempo real los distintos comportamientos sospechosos o delictivos dentro de los medios de transporte. Para el alcance de dicho objetivo, es esencial contar con un dominio de los algoritmos de visión artificial, particularmente de aquellos que se basan en Deep Learning, que facilitan el reconocimiento exacto de patrones y la detección de diversas anomalías dentro del comportamiento humano. A continuación, se detalla una síntesis de los conceptos más importantes que se relacionan directamente con estos temas. Adicionalmente a esto, dentro del Anexo A se describe cada una de las variables de investigación utilizadas para desarrollar la fundamentación teórica de este estudio.

1.3.1 Sistemas de Seguridad

Los sistemas de seguridad contemporáneos se conforman a partir de la combinación sinérgica de diversos campos tanto técnicos como materiales. Estos sistemas incluyen una gama de aparatos de vanguardia, como sistemas de vigilancia de alta resolución, dispositivos de detección de movimiento infrarrojos y alarmas de intrusión de vanguardia, con el único propósito de proteger los componentes físicos y cibernéticos [9].

a. Entre las características que mayor se destacan

- **Adopción de tecnologías emergentes:** Donde se integran herramientas como es el caso del internet de las cosas (IoT), sistemas biométricos, realidad virtual, todo esto con el propósito de lograr potenciar las capacidades relacionadas a la seguridad y la optimización de respuestas ante amenazas [9].

- **Coordinación estratégica entre seguridad TI y TO:** Resulta fundamental la fusión de la seguridad de la información (TI) y la seguridad operativa (TO) para salvaguardar los sistemas de control industrial y la infraestructura vital [9].

- **Implementación de estrictos controles físicos:** La aplicación de diversas medidas de vanguardia, como sistemas de control por biométrico, sistemas de videovigilancia de circuito cerrado (CCTV), garantiza una protección física inquebrantable [9].

1.3.2 Sistemas de seguridad en medios de transporte

Los sistemas de seguridad aplicados en medios transporte inteligentes, poseen una arquitectura de alto nivel que proporciona una descripción de la funcionalidad y los enlaces de comunicación entre los nodos y los vehículos. Todo esto se compone de un conjunto de componentes interconectados y organizados en dos ámbitos principales: Intra-vehículo e Inter-vehículo [10].

a. Dominio Intravehicular.

La cantidad de componentes electrónicos y de dispositivos integrados en los vehículos modernos va en aumento continuamente. Una infinidad de sistemas informáticos integrados que se encuentran interconectados, se denominan control electrónico (ECU), los mismos que forman una red distribuida para lograr controlar las diversas funciones del vehículo. Podemos decir que los ECU pueden comunicarse entre sí por medio de varias redes de comunicación en el vehículo: Controller Area Network (CAN), Local InterConnect Network (LIN), FlexRay, Media Oriented Sys (MOST), y Ethernet. La elección para el uso de cada uno de ellos depende de la criticidad, el costo de la implementación, el uso del ancho de banda y la temporización de las funciones deseadas por el usuario [10]. En la Figura 1 se observa la arquitectura y los componentes clave de un sistema de transporte inteligente, dentro del dominio intravehicular.

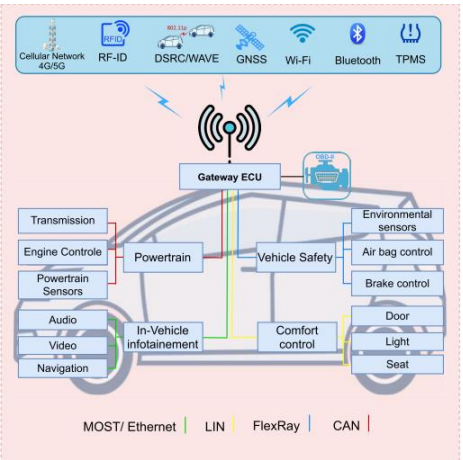


Figura 1. Dominio Intravehicular [10].

A continuación, en la Tabla 1 se detalla las tecnologías actuales de la capa física de un automóvil.

Tabla 1. Tecnologías actuales de la capa física del automóvil [10].

Nombre del bus	**Tasa de transferencia**	**Control de acceso**
Red de área de controlador	1 Mbits/s	CSMA/CA y AMD
Red de interconexión local	20 Kbits/s	Polling
FlexRay	10 Mbits/s	TDMA
Sistema de transporte orientado a los medios de comunicación	150 Mbits/s	CSMA/CD y TDM
Ethernet	100 Mbits/s	TDMA y TDD

CAN y FlexRay están diseñados específicamente para poder proporcionar una transmisión de datos rápida y de muy bajo costo, también para aplicaciones críticas, como el control de la transmisión y de la seguridad. LIN se encuentra diseñado para funciones que requieren menor velocidad de transmisión, como es el caso del control de luces, control de puertas, aire acondicionado y el control de los asientos. MOST es un bus de muy alta velocidad diseñado para las diversas aplicaciones multimedia en el entorno del automóvil [10].

b. Dominio Intervehicular.

Dentro del paradigma entre vehículos abarca la comunicación entre vehículos y sus entornos circundantes, como son el caso de peatones, bicicletas o comúnmente denominados como Usuarios Vulnerables de la Vía Pública (VRU), así como la infraestructura regional. Cada vehículo que cuente con la tecnología necesaria puede formar parte de la red y a la vez ser capaz de enviar y recibir mensajes relacionados con diversas aplicaciones direccionadas con seguridad, gestión del tráfico, etc. La comunicación entre vehículos puede referirse a V2X (vehículo a cualquier cosa), que incorpora tipos de comunicación más específica en función de las entidades destinatarias, se cuenta con la comunicación vehículo a vehículo (V2V), vehículo a infraestructura (V2I), de vehículo a peatón (V2P), y de vehículo a red (V2G). Para proporcionar soporte a estas comunicaciones (DSRC) y el acceso inalámbrico en entornos vehiculares (WAVE) son unos de los estándares inalámbricos más prometedores que se encuentran desplegados en el campo de la transmisión [10]. En la Figura 2 se observa la arquitectura y los componentes clave de un sistema de transporte inteligente, dentro del dominio intervehicular.

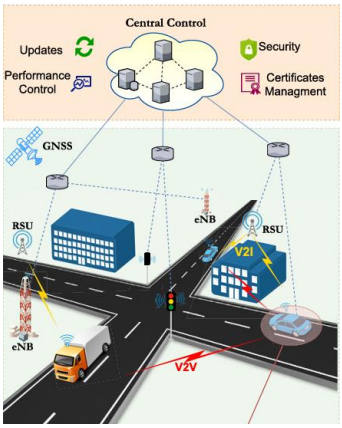


Figura 2. Dominio Intervehicular y Análisis de datos [10].

1.3.3 Sistemas de seguridad Privada

De acuerdo con la Ley Orgánica de Vigilancia y Seguridad Privada las diversas actividades que ejecutan los sistemas de Seguridad Privada son el conjunto de diversos procesos, procedimientos, métodos y técnicas, que ejecutan los distintos prestadores de servicios de seguridad privada, con el objetivo de lograr cumplir su misión societaria, que implica la protección de personas y activos, y a la vez para velar por el normal funcionamiento de las actividades de los usuarios [11].

1.3.4 Sistemas electrónicos de seguridad

Un sistema de seguridad electrónico puede ser considerado como el conjunto de equipos y de componentes fundamentales para lograr garantizar la integridad de las personas y de los bienes materiales, que se encuentran en existencia dentro de un lugar determinado. Estos sistemas se encuentran conectados por lo general con una central de alarmas, las mismas que actúan en función del tipo de situación que se encuentre. Adicional a ello, estos sistemas se clasifican en: Seguridad Estructural y Seguridad Personalizada [12].

a. Seguridad Estructural

La seguridad estructural es el conjunto fundamental de equipos destinados para lograr salvaguardar la integridad de los distintos bienes materiales de las instalaciones, entre los que más se destacan están; los Sistemas de seguridad contra incendios y Sistemas de detección de gas. En la gran mayoría de los casos es obligatorio que se lleve a cabo su instalación dentro de distintos establecimientos como son el caso de edificios públicos, hospitales, etc. [12].

Los sistemas de seguridad electrónica contra incendios tienen como finalidad principal la detección temprana de incendios, con el propósito de evitar afectaciones o pérdidas humanas y a la vez daños materiales en las instalaciones. La respuesta principal de estos sistemas es la alerta de carácter sonora y luminosa; en algunos casos también genera la activación de sistemas de extinción de incendios automatizados, mediante el corte de suministros de electricidad y de gas o puede también presentar una respuesta

de comunicación de emergencia hacia la estación de Bomberos que se encuentre más cerca [12].

Continuando con el tema de los sistemas de seguridad electrónica, tenemos también aquellos sistemas de detección de gas. Los mismos tienen por objeto; detectar acumulaciones de gases, riesgos de intoxicación por la presencia de gases tóxicos, entre otros. Por tal motivo, dentro de los criterios de instalación del sistema está presente defender la detección del tipo de gas [12].

b. Seguridad Personalizada

Dentro de esta clasificación se cuenta con el conjunto de equipos destinados para salvaguardar la integridad de las personas y de los distintos bienes materiales de la vulnerabilidad externa que pueda influir en ellas, dentro de los sistemas más comunes se cuenta con los Sistemas antirrobo y de los Sistemas de videovigilancia. Si se va a llevar a cabo la instalación de este tipo de sistemas, es importante tener en cuenta que existen normativas y regulaciones técnicas de origen nacional e internacional [12].

Estos sistemas antirrobo son definidos como un conjunto de equipos que encargan de lograr gestionar la intrusión de tipo no autorizado o el allanamiento a un determinado o perímetro específico, los robos o atracos a personas, a bienes o inmuebles, el monitoreo de la presencia de individuos dentro de un edificio y la gestión del acceso, registrando la entrada y salida de personas y vehículos a un establecimiento determinado [12].

Hablando ahora acerca de los sistemas de video vigilancia, aquellos constan de equipos que se encuentran direccionados en la visualización y el almacenamiento de imágenes; las mismas que son captadas por cámaras de seguridad dedicadas para el control y el monitoreo de las situaciones en tiempo real, todo esto dentro y fuera de una edificación. La gran parte de estos equipos se componen de cámaras, equipos de visualización (monitores), dispositivos de almacenamiento digital y de tratamiento de señales de video y de audio, conectados y controlados a la red de internet [12].

1.3.5 Sistemas de monitoreo por video

Los sistemas de monitoreo por video es un instrumento de seguridad activa conformado por distintos equipos con la capacidad de captar, visualizar, controlar y grabar audio y video para lograr vigilar, gestionar y controlar establecimientos en tiempo real, de manera interna y externa. Es importante determinar la diferencia que existe entre los términos CCTV y sistemas de monitoreo por video, dado que, aunque parecen ser similares, no lo son; si existe una diferencia entre ambos. Los circuitos cerrados de televisión tienen la capacidad de visualizar las imágenes captadas por las cámaras en tiempo real, y se transforman en sistemas de monitoreo por video con la capacidad de almacenar y lograr procesar grabaciones [12].

a. Aparatos y componentes que forman parte de los sistemas de monitoreo por video

- ***Dispositivos de captura***

Este dispositivo es uno de los más importantes dentro del sistema, por el hecho de contar con la capacidad de captar imágenes, dependiendo del lugar donde haya sido colocada la misma. Pero para que la misma realice dicha función depende de diversos elementos que logran definir sus propiedades de uso y funcionamiento, como son el caso de sensores de captación, carcasas externas, soportes, posicionadores y el resto de los accesorios complementarios necesarios para llevar las funciones de manera eficiente [12]. A continuación, en la Tabla 2 se detalla la clasificación más común de cámaras CCTV.

Tabla 2. Tipos de Dispositivos de Captura [12].

Tipo de Cámara	Descripción
Cámara de Caja	Este tipo de cámara tradicional es utilizada con una carcasa fija y su instalación se la puede llevar a cabo en interiores o exteriores.
Cámara Domo	Cámara con forma de domo y que se usa generalmente en interiores y posee un diseño discreto.
Cámara Tipo Tubo	Cámara alargada que es utilizada generalmente en exteriores y resulta ser ideal para vigilancia a larga distancia.
Cámara PTZ	Cámara motorizada que permite el control remoto para cambiar su dirección y ángulo de visión.

Dentro de las características técnicas más destacables para la selección de estos dispositivos de captura con sistemas CCTV son las dimensiones y el peso, también las características del sensor, el lente, la resolución, la sensibilidad, entre otras. Como ya se ha mencionado antes, los dispositivos de captura son el principal componente dentro de un sistema de videovigilancia, por dicho motivo los parámetros técnicos que se acaban de mencionar deben entrar en una comparativa teniendo siempre en cuenta la aplicación que se le vaya a dar [12]. En la Tabla 3 se muestran las características técnicas de los dispositivos de captura más utilizados.

Tabla 3. Características de los dispositivos de captura más utilizados [13–16].

Parámetro	**AXIS P5635-E PTZ**	**Bosch AUTODOME IP starlight 7000i**	**Dahua N52A Series**	**Sony SNC-VB770**
Dimensiones	180 mm × 131 mm × 145 mm	238 mm × 150 mm × 146 mm	161.4 mm × 69.6 mm × 69.2 mm	95 mm × 93 mm × 143 mm
Peso Aprox.	Aprox. 2.5 kg	Aprox. 3.6 kg	Aprox. 200 g	Aprox. 1.5 kg
Sensor Óptico	2 MP CMOS	1/2.8” CMOS, Starlight	2 MP CMOS	12.1 MP CMOS
Lente	4.3-129 mm motorized zoom	4.5-45 mm motorized zoom	3.6 mm	12 mm fice lens
Ángulo y Alcance de captación	Pan: 0° to 360°, Tilt: 0° to 90°, Rotation: 0° to 360°	Pan: 0° to 360°, Tilt: 0° to 90°, Rotation: 0° to 360°	Pan: 0° to 360°, Tilt: 0° to 90°, Rotation: 0° to 360°	Pan: 0° to 360°, Tilt: 0° to 90°, Rotation: 0° to 360°
Resolución	1920 (H) × 1080 (V)	1920 (H) × 1080 (V)	1920 (H) × 1080 (V)	3840 (H) × 2160 (V)
Salida de video	H.264, MJPEG	H.264, MJPEG	H.264, MJPEG	H.264, MJPEG
Entrada de Audio	Si	No	No	Si
Alimentación	12 VDC ± 25%	12 VDC ± 25%	12 VDC ± 30%	12 VDC ± 25%
Potencia de Consumo	14 W	18 W	3.1 W	10 W
Condiciones de uso	Temperatura desde -40 °C a 60 °C, Humedad máxima de 90% (Sin condensación)	Temperatura desde -40 °C a 60 °C, Humedad máxima de 95% (Sin condensación)	Temperatura desde -40 °C a 60 °C, Humedad máxima de 95% (Sin condensación)	Temperatura desde -30 °C a 50 °C, Humedad máxima de 95% (Sin condensación)
Fabricante	AXIS	Bosch	Dahua	Sony
Costo	$1,900	$2,400	$150	$2,000

- ***Sistemas de transmisión***

Es el medio por el cual se va a realizar la transmisión de las señales de audio y de video, las mismas que pueden ser cableadas mediante par trenzado, cables coaxiales o

por fibra óptica, y de manera inalámbrica con la ayuda de ondas electromagnéticas o mediante WI-FI [12].

- ***Sistemas de reproducción***

Estos sistemas son conocidos también como transductores, los mismos que cuentan con la capacidad de recuperar las señales eléctricas de las cámaras y convertirlas después en imágenes, con el objeto de ser visualizadas y monitorizadas en tiempo real. Actualmente se cuenta con equipos muy eficientes y con gran capacidad tecnológica denominados monitores. Dentro de los tipos más comunes tenemos los siguientes [12]:

- Monitores Digitales TFT
- Monitores LED
- Monitores LCD
- Monitores OLED
- Monitores Curvos
- Monitores Gaming
- Monitores Táctiles

Dentro de los parámetros más relevantes para la elección de un monitor de visualización CCTV tenemos los siguientes [12]:

- Tipos de Puertos
- Resolución
- Tamaño de la pantalla
- Ángulos de visión
- Frecuencia de actualización o refresco
- Brillo y contraste

- ***Sistemas y dispositivos de mando***

Resulta ser muy importante e infaltable un sistema de mando o control dentro de un sistema CCTV que logre gestionar los dispositivos de captura (cámaras), dentro de los sistemas y dispositivos de mando más utilizados actualmente están los sistemas de monitoreo remoto, los repartidores de video y por último las conmutadores de video [12].

Sistemas de monitoreo remoto. Sistemas con señalización electrónica para llevar a cabo el control remoto de parámetros y funciones específicas de cámaras; como son el caso del control de movimiento horizontal, vertical, zoom de imágenes, entre otros [12].

Repartidores de video. Dispositivos que permiten lograr la conmutación de varias señales de video a uno solo o a varios monitores, mediante la multiplicación y amplificación de las salidas de video [12].

Conmutadores de video. Este dispositivo permite gestionar varias entradas y varias salidas de video y audio hacia los monitores que se encuentran conectados en el sistema [12].

- ***Dispositivos de registro***

Estos dispositivos permiten almacenar la información que ha sido captada por las cámaras, además permiten reproducir este material para supervisar eventos generados durante el momento de monitorización. Dentro de la clasificación de tecnologías de grabación más comunes existen los grabadores sobre cinta de video magnética (VCR) y los grabadores digitales como son los DVR o NVR. Dentro de los parámetros más relevantes en el momento de elegir un dispositivo de grabación CCTV son los siguientes [12].

  - Capacidad de almacenamiento
  - Resolución y calidad de grabación
  - Canales y número de cámaras

- Accesibilidad remota
- Soporte de tecnología de cámaras
- Capacidad de grabación en vivo y reproducción
- Seguridad y protección de datos
- Compatibilidad con otros sistemas
- Precio

b. Arquitecturas

La arquitectura de un CCTV va en dependencia de la infraestructura en la que se vaya a realizar la implementación del sistema, todas estas arquitecturas se rigen en base a modelos estandarizados, y manteniendo los mismos elementos que ya se han detallado con anterioridad [17].

Si el sistema CCTV es analógico, la señal de video analógica es producida después de un proceso de conversión de los cambios de la intensidad de la luz en señales eléctricas. Dentro de los dispositivos de captación (cámaras) se genera un proceso de digitalización, pero antes de transmitirla de nuevo la convierte en una imagen análoga para que posteriormente sea recibida por un monitor o por un grabador de video con la ayuda de un conductor físico. La conexión de las cámaras es directamente hacia el sistema de grabación o DVR, con la ayuda del cable coaxial con malla de cobre y el conector BNC. El cable más usado para estas aplicaciones es el RG-59. El número de cámaras siempre va a estar limitada al número de canales que existan en el DVR y de la distancia de transmisión por el calibre del cable a utilizar [17].

La Figura 3 es un esquema básico CCTV Análogo. Donde se observa un DVR (Digital Video Recorder) como unidad principal, la cual se conectan cámaras de tipo analógico mediante un tipo de cableado coaxial. El DVR posibilita la visualización directa en un monitor local y se fusiona a una red Ethernet para contar con el acceso remoto desde una portátil u otros dispositivos conectados dentro de la misma red, facilitando la gestión y la reproducción de las imágenes de seguridad. Dicha configuración

representa un sistema de CCTV con la capacidad de almacenamiento local y de acceso básico en red.



Figura 3. Esquema Básico CCTV Análogo [17].

Si se va a trabajar con cámaras IP hay que tener en cuenta que el proceso es casi similar al sistema de CCTV con cámaras analógicas. Las cámaras IP captan los cambios de intensidad de la luz en señales eléctricas, posteriormente las digitaliza, realiza el proceso de la codificación sobre protocolos de redes Ethernet que en este caso son TCP/IP y las envía al sistema de grabación NVR. La alimentación de las cámaras IP son con cables de red UTP por medio de tecnología PoE, las cámaras se conectan con el cable UTP y el conector RJ-45, dicho el cableado está limitado a un máximo de 100 metros. El número de cámaras soportadas se encuentra limitada al número de canales que cuenta el NVR, una de las ventajas de trabajar con cámaras IP es la integración con otros sistemas que se encuentran conectados a la misma red [17]. La Figura 4 muestra un esquema básico de CCTV.

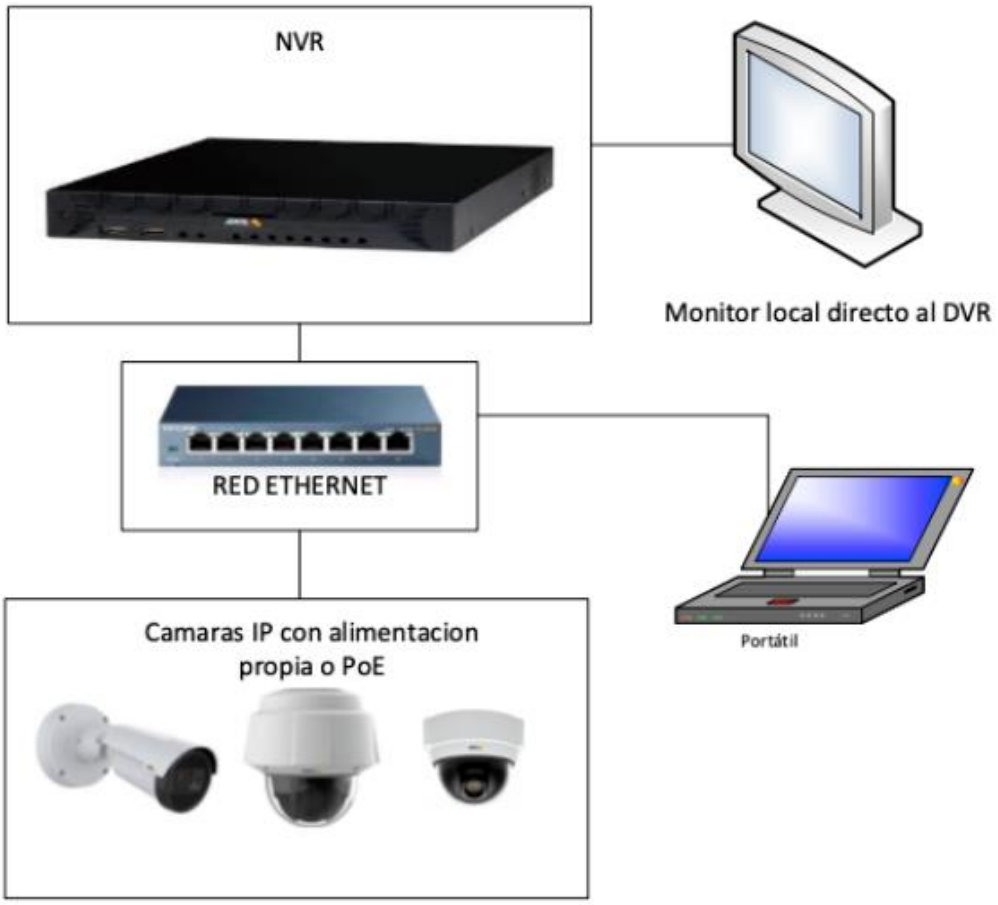


Figura 4. Esquema Básico CCTV IP [17].

c. Reglamento

De acuerdo con la Ley Orgánica de Vigilancia y Seguridad Privada, la misma que fue publicada dentro del Registro Oficial en el mes de febrero en el año 2024, establece un marco regulatorio de carácter estricto que garantiza que las empresas y las organizaciones que operan sistemas de vigilancia lo realicen dentro de los parámetros legales, precautelando tanto la seguridad de las instalaciones como también los derechos de las personas, especialmente en lo que se refiere a su privacidad y a la protección de los datos [11].

Dentro de los aspectos clave que constan dentro de la normativa es que las empresas que realicen instalaciones y operen legalmente con sistemas de videovigilancia deben encontrarse registradas y certificadas por las autoridades correspondientes. El registro garantiza que las empresas cumplan con estandarizaciones técnicas y de alta calidad para llevar a cabo instalaciones y mantenimiento de los sistemas. Adicional a ello, la

ley exige que las cámaras que se encuentran instaladas cubran solamente las áreas necesarias para la seguridad y que a la vez las mismas no invadan los espacios privados que no se encuentran autorizados para el monitoreo, por lo que es fundamental siempre respetar el derecho a la intimidad de las personas [11].

Las grabaciones de CCTV deben estar almacenadas de manera segura y por un tiempo determinado, con el objetivo de contar con dicho material solo para propósitos de seguridad; debido a que es lo que establece la ley. Las personas deben ser informadas por la organización que cuenta con el sistema de cámaras de que están siendo grabadas por medio de señalizaciones adecuadas dentro de los espacios donde las cámaras se encuentran colocadas [11].

Por último, la Ley menciona que cualquier uso indebido de imágenes o de grabaciones, para fines distintos a la seguridad, puede conllevar sanciones graves para las empresas responsables y las personas involucradas. Por lo que el manejo responsable de los sistemas de vigilancia en el Ecuador es fundamental, como siempre se debe asegurar tener un equilibrio entre la seguridad y los derechos de las personas [11].

1.3.6 Comportamiento Delincuencial en Transporte Público.

El transporte público es un entorno vulnerable a diversos tipos de delitos, como son el caso de robos, agresiones y vandalismo, los mismos que afectan a pasajeros y operadores de las unidades de transporte. La ejecución de estos eventos produce inseguridad, reduce la confianza en los sistemas de transporte y hasta cierto punto puede disminuir la utilización del transporte público, provocando un impacto negativo dentro de la movilidad urbana. El enfoque técnico de la propuesta se direcciona en la implementación de sistemas de videovigilancia en autobuses y estaciones como una solución efectiva para lograr prevenir incidentes y responder rápidamente ante la ocurrencia de estos eventos [18].

En ciudades grandes como Buenos Aires, la integración de cámaras de videovigilancia conectadas a centros de monitoreo ha reducido de forma significativa los delitos. Estas tecnologías facilitan la identificación de patrones de actividad delictiva y así optimizar la respuesta de los sistemas de seguridad. En China y en la ciudad de Moscú, por

ejemplo, han liderado iniciativas que combinan videovigilancia con reconocimiento facial para mejorar la seguridad en espacios públicos [18].

a. Buses urbanos e interprovinciales

Los buses urbanos son un medio de transporte masivo que conecta áreas dentro de las ciudades, mientras que las unidades de transporte interprovincial cubren rutas más largas entre localidades. Este medio de transporte es utilizado por grandes sectores de la población y por consiguiente es vulnerable a delitos como robos y agresiones debido a la alta densidad de pasajeros y los tiempos prolongados de viaje. Un estudio sobre los sistemas de transporte tipo Bus Rapid Transit (BRT) en Bogotá y Quito mostró que las terminales y estaciones son puntos clave tanto para la movilidad como para la interacción social, lo que las convierte en áreas críticas para implementar medidas de seguridad avanzadas [19].

b. Taxis y transporte compartido

El transporte en taxis y servicios compartidos, como es el caso de Uber o aplicaciones similares ofrecen flexibilidad y comodidad. Sin embargo, también enfrentan riesgos que se encuentran asociados con asaltos a conductores y pasajeros. Estos servicios requieren estrategias de seguridad como identificación del usuario, rastreo por GPS y cámaras de seguridad para minimizar el riesgo de incidentes [20].

1.3.7 Tipos de comportamiento delincuencial en medios de transporte público.

Los delitos como son los robos y asaltos dentro del transporte público resulta ser una gran preocupación que se mantiene constante dentro de entornos urbanos, dando como consecuencia afectaciones y molestias a los usuarios y a las autoridades. Estos delitos que implican la sustracción de pertenencias de los pasajeros por medio de actos de violencias y amenazas, todos estos malos actos son muy comunes dentro de rutas donde existe alta concentración de personas y la falta de vigilancia facilita la ejecución de estos actos ilícitos. Llevar a cabo la implementación de sistemas con visión artificial puede resultar esencial para identificar y prevenir estos delitos [21].

a. Violencia física.

La violencia física en el transporte público incluye agresiones físicas, peleas entre pasajeros, intimidaciones y otros actos de violencia directa. Este tipo de conducta delictiva puede surgir de conflictos menores que se intensifican rápidamente, y en numerosas situaciones va en aumento debido al exceso de pasajeros y la aglomeración de viajeros, falta de supervisión en zonas vitales como estaciones y paradas. Una investigación del Instituto de Transporte y Movilidad de Bogotá indicó que la violencia física en los medios de transporte público es uno de los factores más significativos de inseguridad en los medios de transporte público, transporte de grandes volúmenes, lo que demanda un estudio continuo para establecer acciones preventivas, como la implementación de cámaras de vigilancia inteligentes capaces de detectar situaciones de violencia o conflicto. Además, el seguimiento en tiempo real es esencial para poner en marcha respuestas prontas frente a estos sucesos [22].

b. Conductas Sospechosas

Las conductas sospechosas dentro del transporte público incluyen movimientos inusuales, permanencia prolongada en ciertas áreas sin justificación, y comportamientos furtivos como el hecho de evitar el contacto visual o el cambio repentino de asiento. Estos patrones pueden ser factores indicadores de intentos de robo o preparación para otro tipo de actividad delictiva. Dentro de un estudio realizado en diversas ciudades latinoamericanas, se identificó que el comportamiento sospechoso es uno de los primeros indicios de actividades ilícitas en espacios públicos como el transporte colectivo. Los sistemas de visión artificial, mediante algoritmos de detección de anomalías y seguimiento de patrones de comportamiento, pueden identificar y señalar a los operadores de seguridad aquellos comportamientos que se desvían de lo normal, ayudando a prevenir situaciones de peligro antes de que ocurran [22].

1.3.8 Machine Learning.

La inteligencia artificial (IA) forma parte de la informática que se concentra en el estudio, en el desarrollo de sistemas y de algoritmos que sean totalmente capaces de

realizar actividades que generalmente se ve involucrada la inteligencia humana. La inteligencia artificial cuenta con una amplia gama de técnicas y diferentes enfoques, entre ellos consta el aprendizaje automático y dedicado. El aprendizaje automático o machine learning (ML), se refiere a una pequeña parte de la inteligencia artificial que se concentra en el estudio, y en el desarrollo de algoritmos y de modelos computacionales que facilitan el aprendizaje a los sistemas y a mejorar su rendimiento por medio de la experiencia, sin que exista una programación exacta para cada tarea específica. Es decir; con la ayuda del Machine Learning, los seres humanos solo deben ingresar datos, así como las salidas deseadas para estos datos y de manera inmediata el algoritmo con entrenamiento previo adquiere un aprendizaje de las reglas. Y estas reglas en un futuro se pueden aplicar a datos nuevos para lograr generar salidas a datos nunca antes visualizados en entrenamiento [23].

Al hablar de las distintas disciplinas en las que se apoya el Machine Learning se puede decir que los factores que generan el crecimiento del Machine Learning son la gran abundancia de datos que se producen a diario, los procesamientos computacionales que se han convertido más potentes y el almacenamiento de los datos resulta ser más accesibles. Todo esto es posible porque el estudio de Machine Learning es alimentado de distintas disciplinas tal como se muestra en la Figura 5 [24].

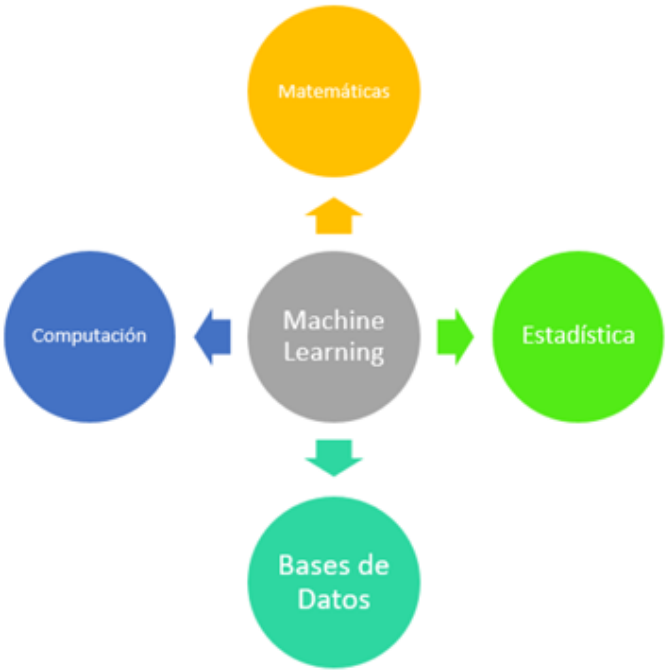


Figura 5. Disciplinas en las que se apoya Machine Learning [24].

a. Diseño de una infraestructura de datos.

En el contexto del aprendizaje automático, es fundamental contar con una infraestructura de datos muy bien diseñada para obtener éxito en cada uno de los proyectos en los que se vaya a utilizar dichos datos. A continuación, se presenta diversas prácticas importantes que establecen una infraestructura de carácter robusto y a la vez eficiente para quien vaya a realizar el manejo de datos dentro del aprendizaje automático. Dichas prácticas se basan desde la etapa de recopilación y el almacenamiento de los datos, con el único objetivo de lograr garantizar integridad, precisión y sobre todo escalabilidad en la infraestructura de los datos [23].

- ***Recopilación y almacenamiento de los datos de alta calidad.***

Entre las obligaciones del recopilador de la información, está garantizar que estos datos recopilados resulten ser pertinentes y representativos, es decir; que logren contribuir de manera específica dentro del problema que va a abordar. Adicional a ello, se debe emplear diversos sistemas de almacenamiento de datos que sean escalables y sobre todo robustos para que sean manejados de forma eficiente ante volúmenes bastante amplios de información. Estos sistemas deben contar con la capacidad de adaptación según también aumente la cantidad de datos, procurando que la información esté con total disponibilidad y accesibilidad cada que la misma sea necesaria, eso sí, sin el comprometimiento de la integridad o de la seguridad de los datos [23].

- ***Limpieza y preprocesamiento de los datos.***

Resulta ser obligatorio realizar una limpieza de los datos antes que se los vaya a utilizar; sobre todo dentro de modelos de aprendizaje automático pues se debe siempre garantizar resultados adheridos a características similares a la precisión y a la fiabilidad. Por ello todo esto conlleva un proceso de identificación y eliminación de valores atípicos, manejo de datos faltantes, corrección de errores y preprocesamiento de los datos [23].

- ***Diseño de pipelines de datos.***

Los pipelines son aquellos sistemas de automatización que permiten gestionar el flujo de datos, desde la fase de recopilación inicial hasta su procesamiento y conversión, y finalmente su suministro al modelo de aprendizaje automático. La implementación de pipelines de datos proporciona una serie de beneficios muy específicos e importantes, entre los cuales constan los siguientes: Eficiencia, consistencia, reproducibilidad, escalabilidad y por último se cuenta con mantenibilidad [23].

- ***Monitorización y mantenimiento.***

Es crucial para la monitorización y supervisión de los modelos, establecer sistemas que logren detectar problemas o degradación dentro de su desempeño. Entre las métricas que deben contar estos sistemas consta la precisión del modelo, la tasa de error, tiempo de respuesta y por último; otros indicadores importantes que permitan la evaluación del rendimiento del modelo en tiempo real; como la implementación de alertas automáticas que se accionen cuando existan anomalías o desviación significativa dentro de las métricas de rendimiento dentro del sistema [23].

b. Tipos de aprendizaje automático

Dentro del mundo digital en el que se vive actualmente, el aprendizaje automático surge como un área de estudio importante, donde las máquinas cuentan con la capacidad de aprendizaje a partir de los datos y a la mejora de rendimientos de manera autónoma, por lo que esta área abarca una variedad muy grande de técnicas y de enfoques que se basan en que las máquinas logren predecir y tomar decisiones importantes o relevantes en los datos [23].

- ***Aprendizaje supervisado.***

Este tipo de aprendizaje realiza predicciones con datos conformados por vectores de entrada, por ejemplo; $x = (x1, x2, \dots, xn)$ con sus respectivos vectores de salida $y = (y1, y2, \dots, yn)$, lo que se denomina un conjunto de datos históricos $S = (x1y1), (x2, y2), \dots, (xn, yn)$ [24].

Este conjunto de datos históricos se divide en dos grupos principales que son los siguientes: los datos de entrenamiento (70 % - 80 %) y los datos de prueba (30% - 20%), donde esta separación de los porcentajes está basada en el principio de Pareto. Donde los datos de entrenamiento son útiles como la entrada a un algoritmo de aprendizaje supervisado. Este tipo de algoritmo produce un determinado modelo de reconocimiento de patrones por medio del comportamiento o de las características que se han visualizado dentro de los datos de entrenamiento. Por lo tanto, el modelo que se genera es evaluada por medio del suministro del conjunto de datos de prueba, la cual de la misma se obtiene un determinado porcentaje de predicción [24]. En la Figura 6 se visualiza como es la manera en la que opera el aprendizaje supervisado.

Antes de llegar a la Figura 6, es importante describir un ejemplo muy puntual dentro de la vida cotidiana de aprendizaje supervisado, en un grupo de alumnos de una determinada asignatura, el docente encargado dicta la clase sobre un tema con ejemplos, les proporciona ejercicios y los alumnos hacen todo lo posible de lograr reproducir lo que aprendieron para llegar al mismo resultado [24].

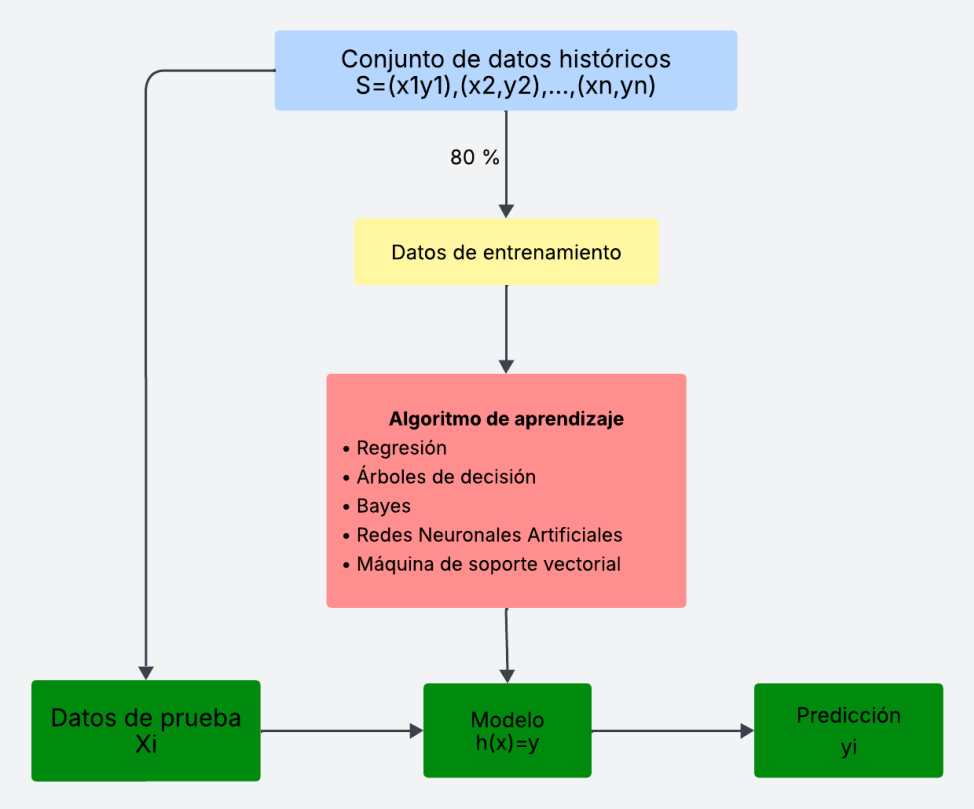


Figura 6. Aprendizaje supervisado. Fuente: Elaboración propia.

- ***Aprendizaje no supervisado.***

Este tipo de aprendizaje opera con datos conformados únicamente por el vector de entrada $x = (x1, x2, \dots, xn)$, aquí no existe un vector de salida y, por lo que no se cuenta con una agrupación de datos dedicados al entrenamiento y por lo tanto no existe un porcentaje de predicción [24]. Tal como se muestra en la Figura 7.

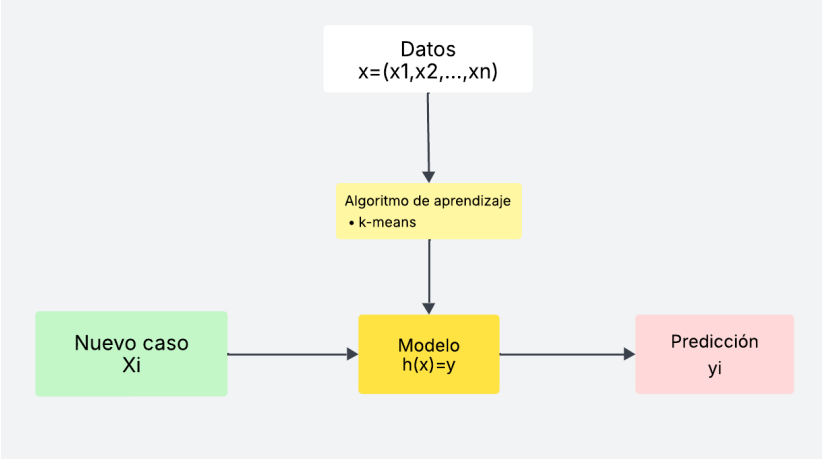


Figura 7. Aprendizaje no supervisado. Fuente: Elaboración propia.

El aprendizaje no supervisado es demasiado útil para realizar tareas de agrupamiento (clustering).

Clustering. Cuando se va a trabajar con este tipo de problemas, hay que proporcionarle a la máquina un conjunto de datos y no se le da ninguna instrucción específica de qué hacer con esos datos ni se le indica el punto de partida [24]. En la Figura 8 se visualiza un conjunto de datos sin valores asignados.

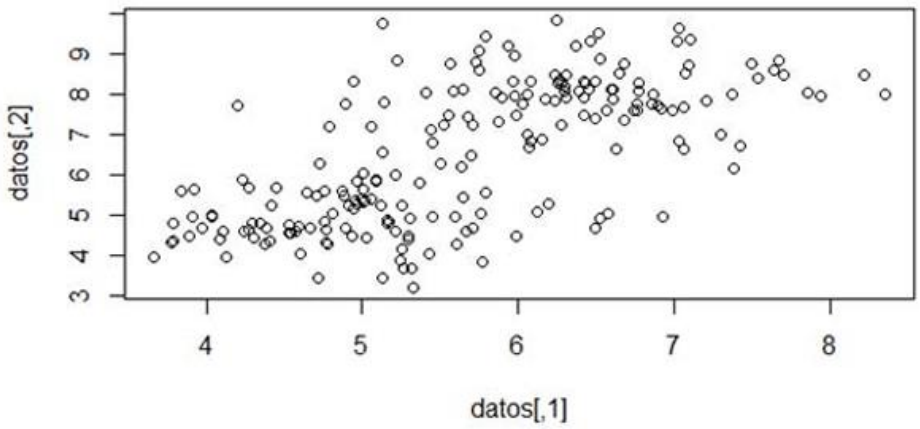


Figura 8. Conjunto de datos sin valores asignados [20].

Con los datos que se mostraron en la Figura 8 y con la ayuda de un algoritmo no supervisado se procede a dividir los datos en cuatro grupos; a esta acción se le conoce como problema de agrupamiento; mismo que se puede visualizar en la Figura 9. Los grupos se distinguen entre sí porque los datos dentro de cada grupo comparten características comunes, mientras que son distintos a los datos de otros grupos [24].

Un ejemplo muy claro para explicar sobre el agrupamiento en Google News. Donde tiene por objeto buscar nuevas historias dentro de una infinidad de datos; los mismos que son generados constantemente en Internet. Cada una de las noticias encontradas son agrupadas de forma automática dentro de historias cohesivas por medio de palabras clave, las mismas que sirven como un patrón para el entrenamiento y de cierta manera generar bloques de noticias que sean del mismo tema.

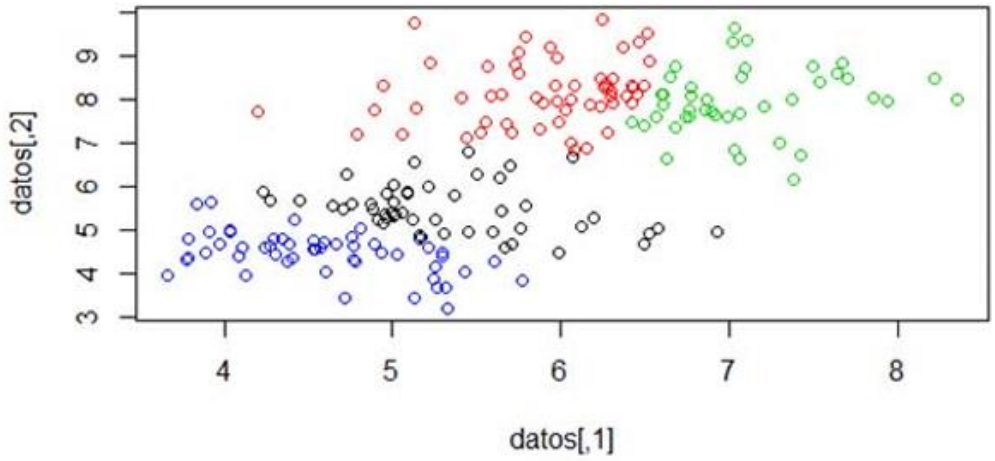


Figura 9. Conjunto de datos divididos en cuatro grupos [20].

- ***Aprendizaje por refuerzo***

Este tipo de aprendizaje resulta ser crucial dentro de la creación de sistemas inteligentes con total capacidad de poder tomar decisiones de manera autónoma y que logren adaptarse a los entornos cambiantes. Este tipo de aprendizaje es utilizado en una amplia variedad de aplicaciones como son dentro de la robótica y de los videojuegos. Los algoritmos más comunes dentro del aprendizaje por refuerzo está el algoritmo Q-learning y el Deep Q-networks (DQN); algoritmos capaces de aprender políticas de decisión complejas dentro de entornos dinámicos logrando resultados

notables en una serie amplia de tareas [23]. Figura 10 muestra los tipos de aprendizaje automático



Figura 10. Tipos de aprendizaje automático [23].

c. Metodología básica para Machine Learning.

A continuación, se describe la metodología básica para realizar una implementación de un proyecto general con Machine Learning, la misma se encuentra compuesta por una serie de pasos que inicia desde la recolección de los datos hasta la etapa en la que el creador del proyecto logra generar conocimiento a partir de la ejecución de la práctica [24]. Tal como se observa en la Figura 11 que en palabras muy simples explica la metodología para Machine Learning.

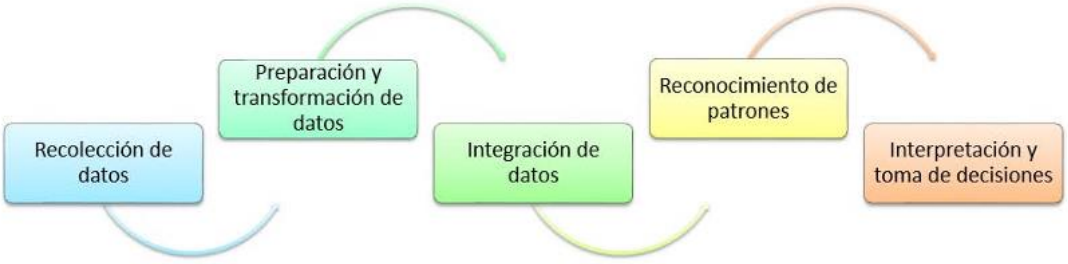


Figura 11. Metodología para Machine Learning [20].

- ***Recolección de datos.***

Dentro de esta etapa se hace la identificación de las fuentes de información que cumplan con cada uno de los requerimientos que solicita el problema. Posteriormente se hace la extracción de los datos más importantes desde una o varias fuentes [24].

- ***Preparación y transformación de datos.***

En esta etapa se realiza el tratamiento a los datos faltantes, a los datos que presentan diversas anomalías o que simplemente son inconsistentes al momento de aplicar técnicas de análisis de datos, las mismas que se componen de medidas estadísticas y de gráficas que son utilizadas para lograr visualizar el comportamiento y posteriormente la mejora de la calidad de los datos. Los datos son sometidos a una transformación para agilizar el uso de algoritmos de aprendizaje, específicamente ya sea reemplazar valores continuos por valores discretos y viceversa [24].

- ***Integración de datos.***

Se sabe que los datos provienen de distintas fuentes y que con gran frecuencia son combinados dentro de un almacén de datos (Data Warehouse); el motivo de ello es para que todos los datos tengan un formato en común. A pesar de que no es imprescindible. Debido a que en algunos casos se procede a trabajar con los datos originales en hojas de cálculo [24].

- ***Reconocimiento de patrones.***

Cumplir con esta etapa es muy importante porque es donde se toma una decisión con respecto al tipo de tarea de aprendizaje que va a realizar, ya sea: regresión, clasificación o puede ser también agrupamiento. Así como el algoritmo de aprendizaje automático o Machine Learning más adecuado para realizar la elaboración del modelo [24].

- ***Interpretación y toma de decisiones.***

Una vez que se obtiene un modelo óptimo para la resolución del problema se asciende a la etapa de la interpretación. Dentro de esta etapa se ocupa la estadística descriptiva para proceder a la presentación de los datos de manera gráfica. Permitiendo que el análisis del conocimiento que se ha producido sea puesto al nivel de usuario [24].

d. Algoritmos

Esta sección incluye una revisión de los algoritmos de Machine Learning más comunes y un análisis exhaustivo del proceso mediante el cual se aprende de los datos y se

generan predicciones. También se trata el tema de la complejidad computacional en relación con estos algoritmos, ofreciendo una visión general sobre la creación de los modelos que les corresponden a cada uno [25].

- ***k-Nearest Neighbors (KNN).***

Este algoritmo es probablemente el algoritmo de aprendizaje automático más simple o sencillo debido a que construir el modelo consiste solo en almacenar el conjunto de datos de entrenamiento. Con el fin de realizar una predicción para un nuevo punto de datos, el algoritmo tiene la capacidad de encontrar los puntos de datos más cercanos en el conjunto de datos de entrenamiento, es decir; los vecinos más cercanos. A partir de lo mencionado se puede concluir que este tipo de algoritmo en su versión más sencilla solo tiene en cuenta un vecino más cercano, que es el punto de datos de entrenamiento más próximo al punto para el que queremos hacer una predicción [25]. En la Figura 12 se visualiza un ejemplo de predicciones del modelo del vecino más próximo.

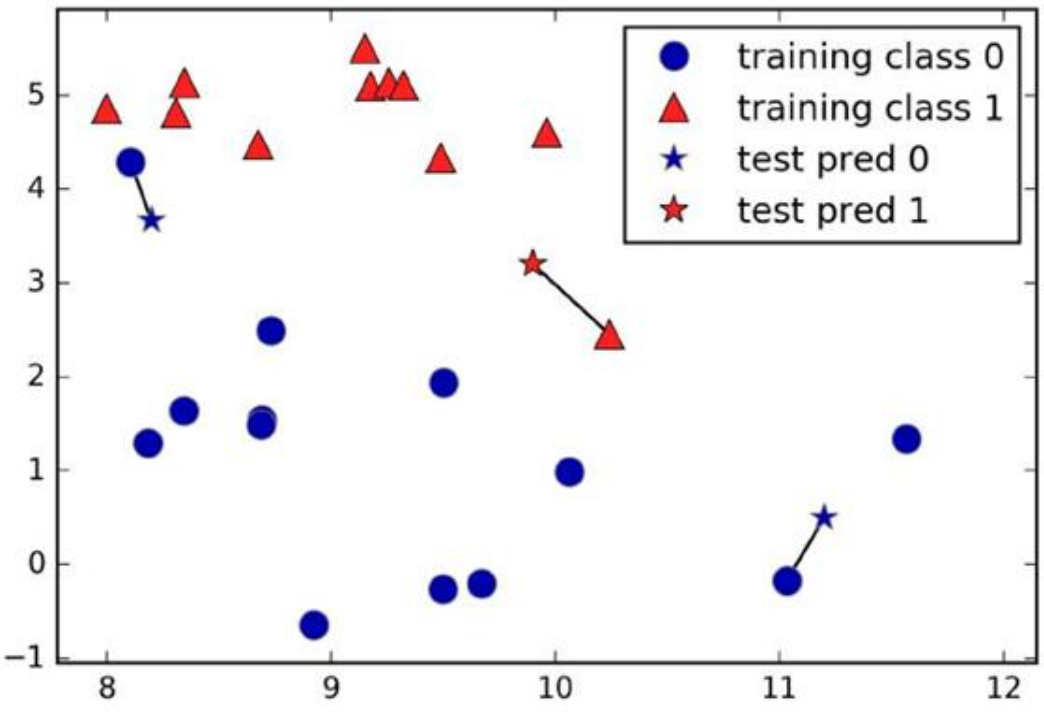


Figura 12. Predicciones del modelo del vecino más próximo en el conjunto de datos de forja [25].

- ***Modelos Lineales.***

Este tipo de modelos son muy utilizados dentro de la práctica y a la vez muy estudiados dentro de las últimas décadas, pero con raíces que se remontan a hace más de cien años. Estos modelos hacen una predicción por medio de una función lineal que dependen directamente de las características de entrada [25]. La fórmula de predicción general para regresión, dentro de un modelo lineal es la siguiente:

$$y = w[0] * x[0] + w[1] * x[1] + w[0] * x[0] + \cdots + w[p] * x[p] + b \qquad (1)$$

Aquí $x[0]$ hasta $x[p]$ son las características, $w[0]$ resulta ser la pendiente y por último b es el desplazamiento que experimenta el eje y. Para realizar más funciones, w cuenta con varias o múltiples pendientes a lo largo del eje de característica. En la Figura 13 se muestra un claro ejemplo de regresión lineal [25].

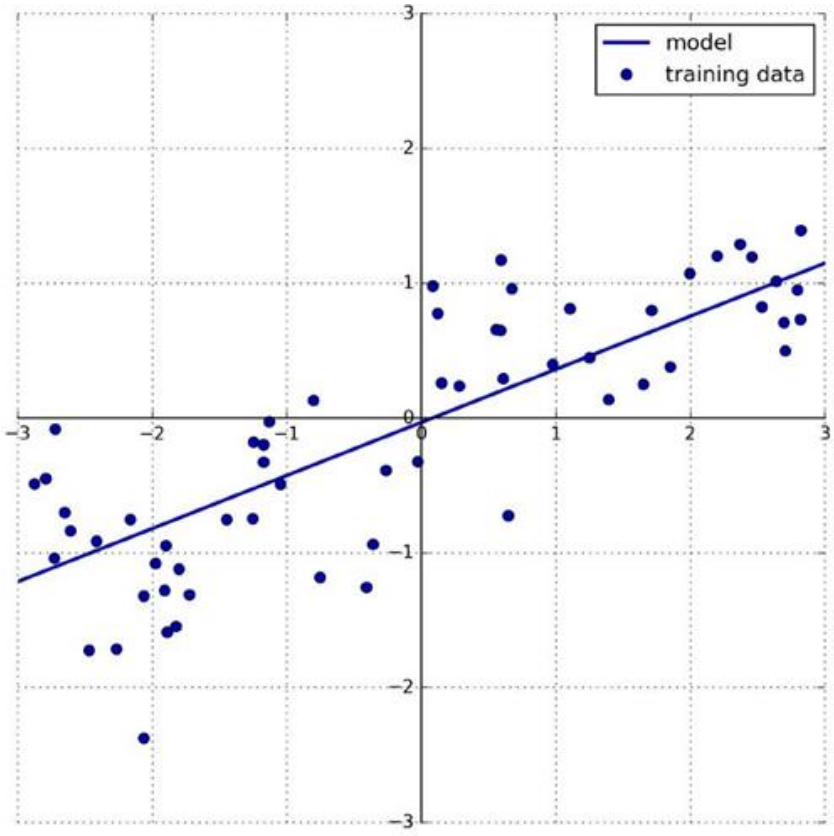


Figura 13. Ejemplo de un modelo de regresión lineal [25].

- ***Clasificadores Naive Bayes***

Los clasificadores de Naive Bayes pertenecen a una familia de clasificadores que son muy similares a los modelos lineales. Pero a diferencia de ellos, éstos tienden a ser incluso mucho más efímeros en la etapa de entrenamiento. Esto se debe a que aprenden parámetros por medio de la extracción de características individuales y recopilaciones estadísticas simples por clase de cada característica. Aquellos modelos generalmente proporcionan un determinado rendimiento de generalización ligeramente inferior que el de los clasificadores lineales como son la Regresión Logística o SVC lineal [25].

- ***Árboles de decisiones.***

Los algoritmos de árboles de decisión son modelos muy comunes para realizar tareas de clasificaciones y regresiones. Principalmente, realizan un aprendizaje de jerarquía de preguntas “si” y “si no” que conllevan a una decisión. Aquellas preguntas son muy similares a las que pueden ser planteadas dentro de un juego que cuente con 20 preguntas. Imagine que quiere realizar una clasificación de los cuatro animales siguientes: osos, halcones, pingüinos y, por último; delfines. El objetivo es llegar a la respuesta correcta mediante la formulación del menor número posible de preguntas si/entonces. Se puede empezar preguntando si el animal tiene plumas, por ejemplo. La misma es una pregunta que reduce los animales posibles a solo dos posibilidades u opciones. Si la respuesta resulta ser afirmativa, se puede realizar otra pregunta que ayude a distinguir entre halcones y pingüinos. Por ejemplo, se puede plantear la siguiente pregunta: ¿si el animal puede volar?, en cambio si el animal no posee plumas, las opciones de respuesta posibles son los delfines y los osos, y se debe plantear una pregunta para distinguir entre los dos animales, por ejemplo, realizando una pregunta relacionada a si el animal tiene aletas [25]. Esta serie de preguntas puede expresarse en forma de árbol de decisión, como se muestra en la Figura 14.

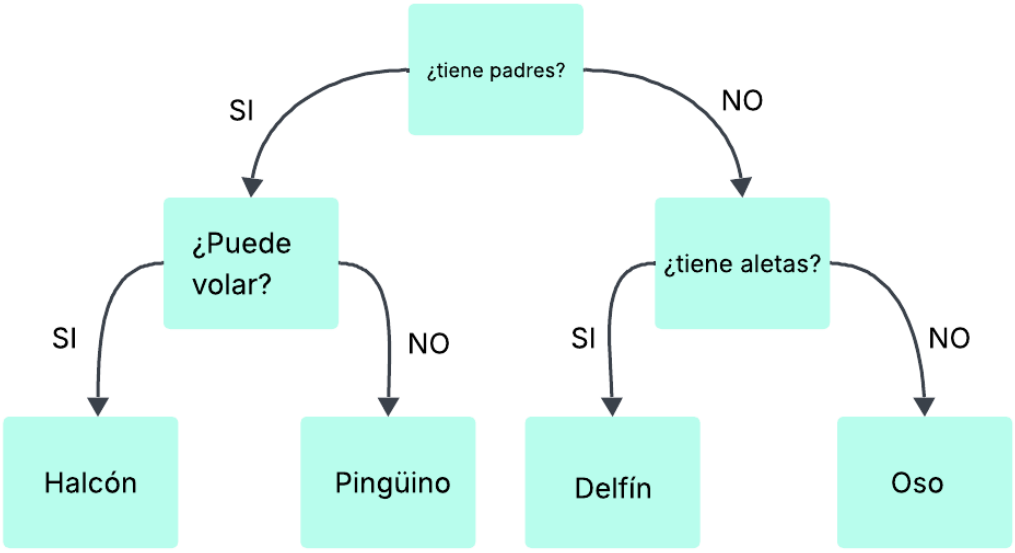


Figura 14. Un árbol de decisión para distinguir entre varios animales [25].

- ***Support Vector Machine (SVM)***

Este tipo de algoritmo es una extensión de los modelos lineales dedicados a la clasificación de modelos más complejos que no se encuentran definidos solamente por hiperplanos en el espacio de entrada. En la etapa de entrenamiento, SVM realiza un aprendizaje sobre cuán importante es cada uno de los distintos puntos de datos de entrenamiento para lograr presentar el límite de decisión entre dos o más clases. Casi siempre solo un subconjunto de los puntos que forman parte del entrenamiento importa para la definición del límite de decisión y son denominados como vectores de soporte [25].

Para la ejecución de una predicción para un nuevo punto se toma en cuenta una decisión de clasificación que sea en función de las distancias al vector de soporte, y la importancia de los vectores de soporte que fueron aprendidos en la etapa del entrenamiento. La distancia entre los puntos de datos se lo hace por medio del núcleo gaussiano (rbf) [25].

$$K_{rbf}(X_1, X_2) = e^{\gamma\, ||(x1-x2)||^2} \qquad (2)$$

En este caso x1 y x2 son puntos de datos, $||(x1 - x2)||$ es la distancia euclidiana y por último gama (γ) es un parámetro que hace un control del ancho del núcleo gaussiano. En la Figura 15 se muestra de manera gráfica el funcionamiento de SVM [25].

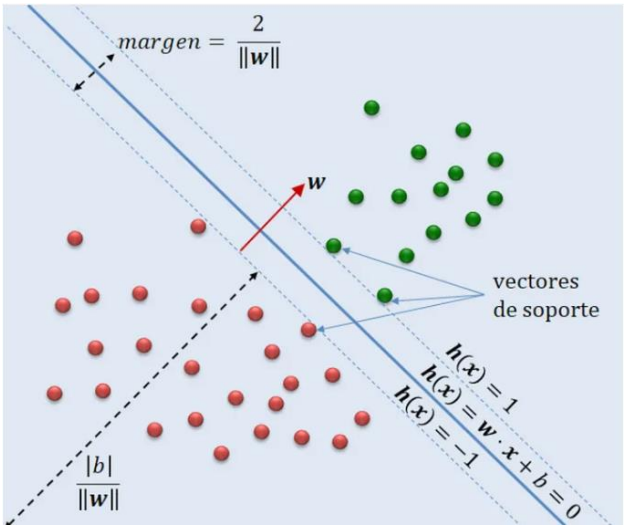


Figura 15. Máquina de soporte vectorial [26].

Para empezar con el entrenamiento de un modelo SVM se debe definir un parámetro C de regularización, idéntico al que se utiliza en los modelos lineales. Por medio de la limitación de cada punto. Dentro de la Figura 16 se plasma un ejemplo de la clasificación de 2 clases por medio de SVM compuesto de distintos parámetros [25].

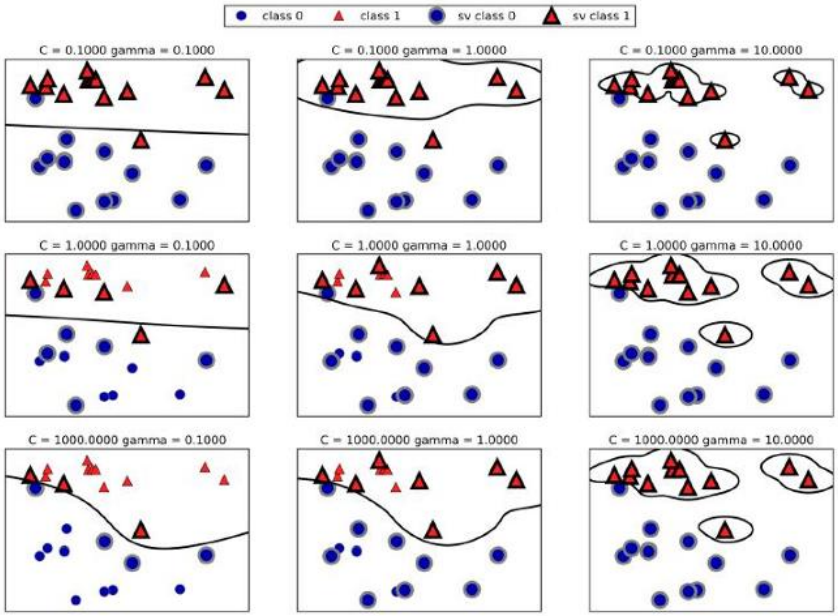


Figura 16. Clasificación binaria con SVM [25].

- ***Redes Neuronales (Deep Learning).***

Las redes neuronales han asumido una popularidad importante dentro del campo del Deep Learning. Los perceptrones multicapa son utilizados dentro de la clasificación y regresión, conforman una base para métodos que son más complejos. Los perceptrones multicapa son también denominados redes neuronales de retroalimentación que han desempeñado un papel significativo dentro del desarrollo de la IA. El perceptrón puede interpretarse como extensiones de modelos lineales que llevan a cabo diversas fases de procesamiento para alcanzar una resolución. La salida de la función lineal es una sumatoria ponderada de las características de entrada $x[0]$ a $x[p]$ valorada por los coeficientes adquiridos de $w[0]$ a $w[p]$ [25]. La Figura 17 muestra de manera gráfica esta idea.

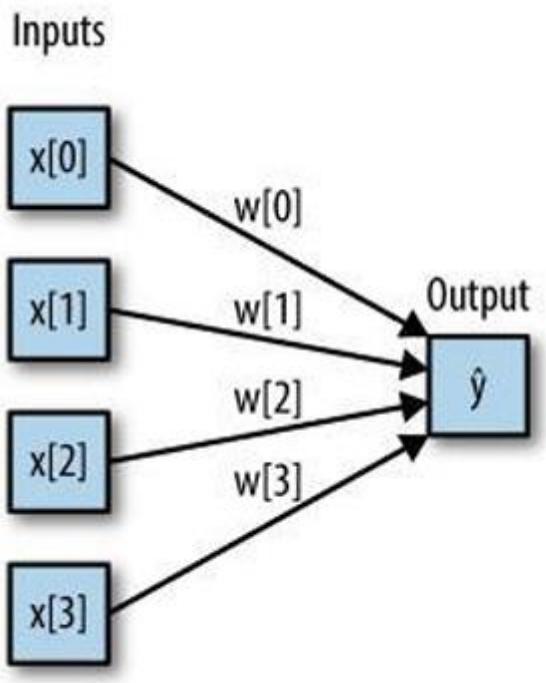


Figura 17. Esquema Perceptrón básico [25].

Dentro de la capa de "Input" cada nodo simboliza una característica de entrada. Los coeficientes aprendidos son simbolizados dentro del nodo "Output", exactamente donde la salida es una suma ponderada de las entradas. Dentro de un perceptrón multicapa, dicho procedimiento de determinar sumas ponderadas se realiza algunas veces. Al inicio se realiza el cálculo de unidades ocultas que simbolizan un paso intermedio de procesamiento, las mismas que son nuevamente unidas o fusionadas por

medio de sumas ponderadas para la obtención del resultado final [25]. La Figura 18 plasma un perceptrón multicapa con una sola capa oculta.

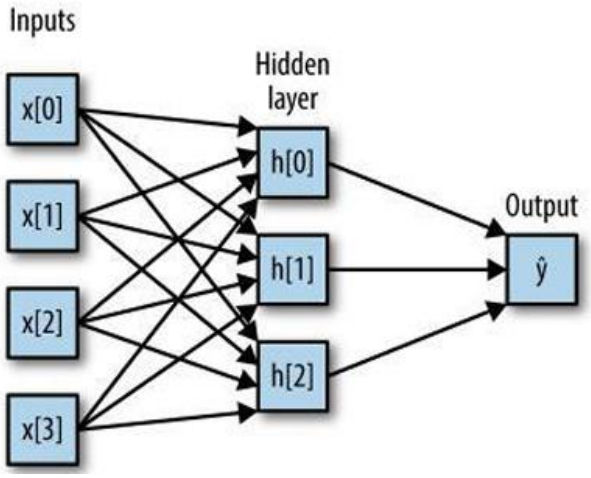


Figura 18. Perceptrón multicapa con una sola capa oculta [25].

e. Plataformas Cloud

Las plataformas cloud dedicadas para entrenamiento de Machine Learning son plataformas que se encuentran basadas en la nube y que ofrecen la infraestructura, herramientas y, por último; servicios indispensables para lograr desarrollar, entrenar y finalmente implementar modelos de Machine Learning. Estos entornos les permiten a los usuarios contar con el acceso a recursos de computación avanzados (como son el caso de GPUs y TPUs) y llevar a cabo el almacenamiento de grandes volúmenes de datos sin la necesidad de mantener infraestructura local costosa. Estas plataformas proporcionan acceso a entornos colaborativos, almacenamiento de datos seguro y herramientas automatizadas que permiten facilitar absolutamente todo el ciclo de vida de un proyecto de Machine Learning [27].

- ***Google Colab.***

Google Colab es una plataforma totalmente gratuita y ofrecida por Google. Esta plataforma proporciona un entorno de cuaderno Jupyter que se encuentra alojado en la nube la cual permite llevar a cabo la ejecución de código Python. Adicional a ello, se puede visualizar los resultados de forma directa dentro del navegador web, sin la necesidad de realizar algún tipo de configuración o instalación extra. Dicha plataforma fue desarrollada para facilitar el trabajo de equipos integrados; permite compartir y a

la vez colaborar de manera muy sencilla con otras personas. Esto es muy importante para los analistas y para los desarrolladores juniors que requieren de revisiones profundas y apoyo dedicado [28].

Uno de los retos tecnológicos más significativos, particularmente para aquellos que laboran en proyectos de aprendizaje automático y similares, es el manejo de datos. Por lo tanto, Google Colab se asemeja a una Ferrari. Una de las principales ventajas de la herramienta es justamente el acceso sin costo a recursos de computación, tales como CPU, GPU y TPU. Esto facilita la ejecución de código de gran tamaño que demanda un procesamiento considerable sin cefaleas ni gastos extra [28]. En la Figura 19 se ejemplifica un gráfico de la función seno generado en Google Colab, con etiquetado y leyenda añadidos para mayor claridad.

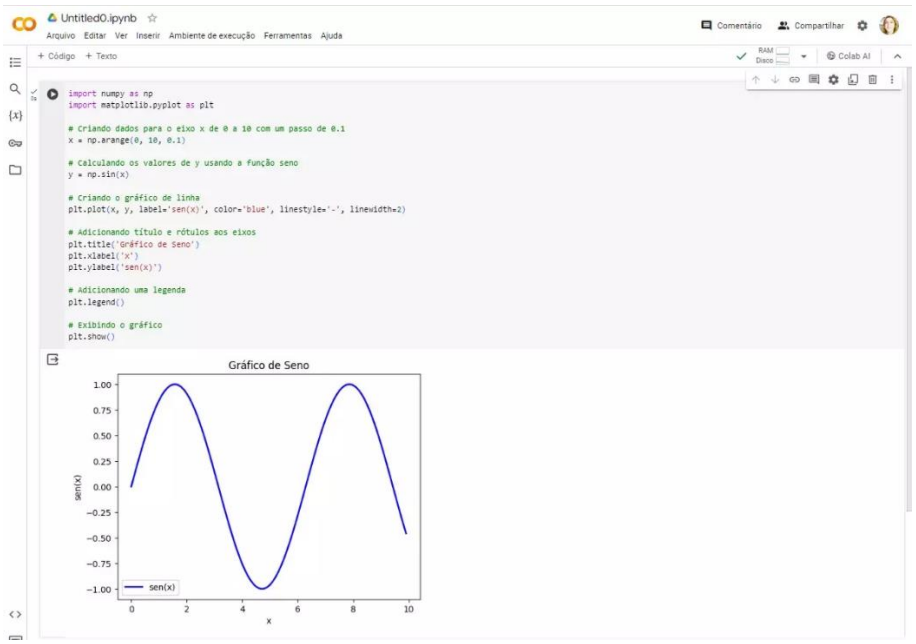


Figura 19. Gráfico de la función seno generado en Google Colab, con etiquetado y leyenda añadidos para mayor claridad [28].

- ***Amazon Web Services (AWS) – SageMaker***

Es una de las plataformas líderes de Cloud Computing a nivel global, proporcionando más de 200 servicios completos a escala global, que van desde tecnologías de infraestructura como computación, almacenamiento y bases de datos, hasta tecnologías en auge como el aprendizaje automático e inteligencia artificial. Amazon

SageMaker es uno de los productos que AWS ofrece para usos de aprendizaje automático [29].

Este es un servicio completamente gestionado que incluye una extensa gama de herramientas de alto rendimiento y asequibles para el desarrollo de soluciones de aprendizaje automático. Puede desarrollar, capacitar e instaurar modelos a escala utilizando herramientas como Notebooks, depuradores, generadores de perfiles, canalizaciones, MLOP y más, todo ello en un ambiente de desarrollo integrado (IDE). SageMaker proporciona acceso a cientos de modelos previamente entrenados, incluyendo máquinas virtuales accesibles de manera pública, que se pueden implementar de manera sencilla. La versión gratuita de SageMaker brinda una prueba gratuita de 2 meses que otorga 250 horas de entrenamiento. al mes para el uso de cuadernos t2.medium o t3.medium, 50 horas al mes para entrenamiento en m4.xlarge o m5.xlarge y 125 horas al mes para hosting en m4.xlarge o m5.xlarge [30]. Figura 20 muestra la creación de entornos seguros de aprendizaje automático con Amazon SageMake

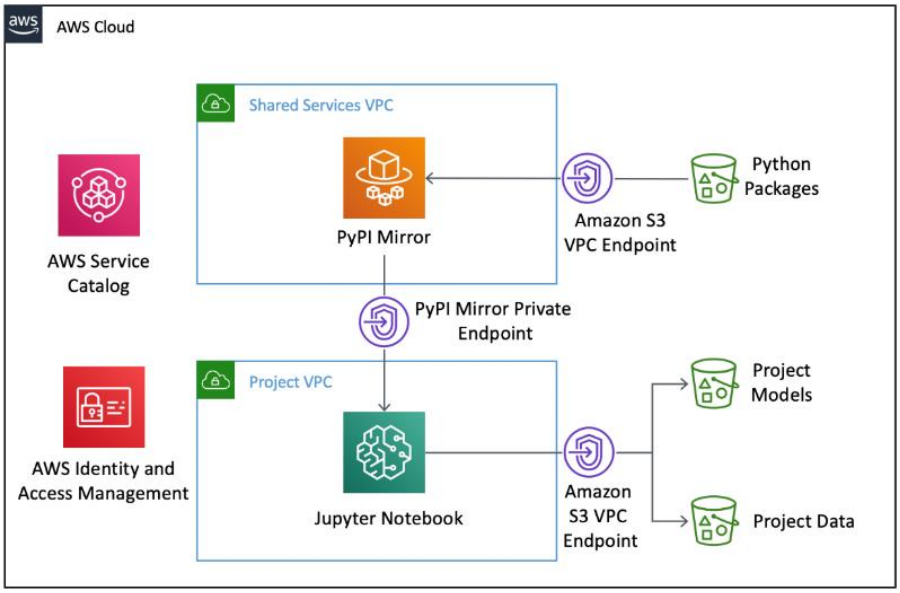


Figura 20. Creación de entornos seguros de aprendizaje automático con Amazon SageMake [31].

- ***Microsoft Azure***

Azure Machine Learning ofrece herramientas que simplifican el proceso de entrenamiento, implementación y administración de proyectos de Machine Learning en la nube. Azure también ofrece servicios que son diseñados para facilitar tareas de desarrollo y escalamiento de cualquier aplicación como servicio (laaS, PaaS y SaaS). Azure también se centra en proporcionar ayuda a las empresas a construir soluciones de IT de forma eficiente, generando recursos como máquinas virtuales, bases de datos, gestores de almacenamiento y redes [32]. En la Figura 21 se encuentra una integración de Microsoft Azure Machine Learning.

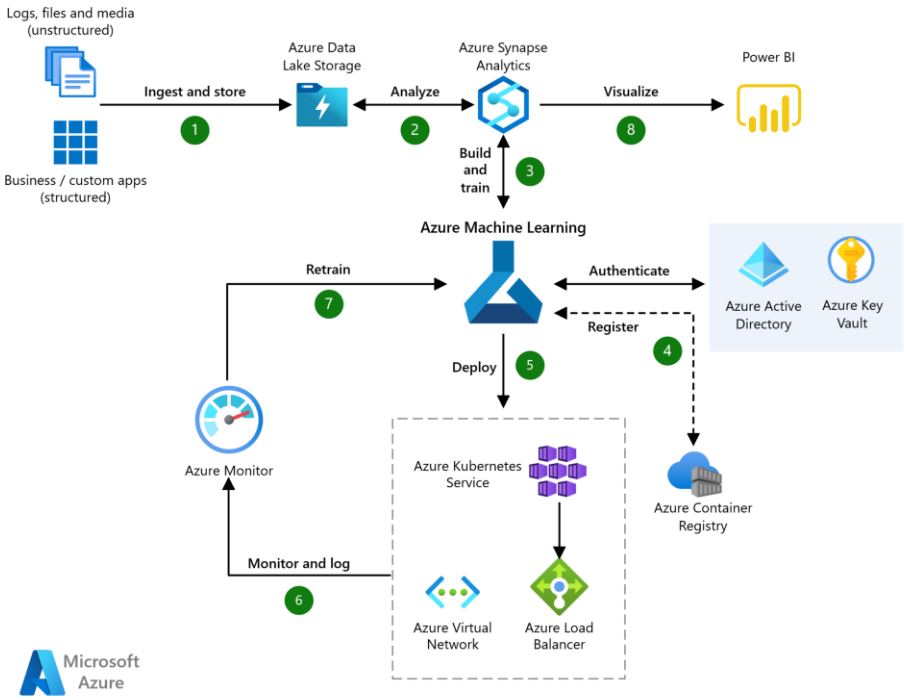


Figura 21. Integración de Microsoft Azure Machine Learning [33].

En la Tabla 4 se muestra las características técnicas de varias plataformas cloud de entrenamiento para machine Learning.

Tabla 4. Características técnicas de varias plataformas cloud de entrenamiento para machine Learning [27–33].

Criterio	Google Colab	Amazon Web Services (AWS) – SageMaker	Microsoft Azure
Tipo	Notebooks en la nube	Notebooks, competencias, datasets, modelos y foros.	Notebooks en la nube, Plataformas y Servicios completos de ML
Costo	Gratuito con límites	Plataforma completa de ML	Pago según uso
Colaboración	Muy buena (compartir fácilmente)	Amplio (compatible con la mayoría)	Buena (colaboración en equipos de proyectos)
Soporte para Bibliotecas	Amplio (TensorFlow, PyTorch, etc.)	GPU/CPU escalable según necesidad	Muy amplio (TensorFlow, PyTorch, etc.)
Recursos de cómputo	GPU/TPU gratuitos (con limitaciones)	Pago según uso	GPU y CPU escalables según necesidad
Facilidad de uso	Intuitivo y accesible para principiantes	Intuitivo, orientado a competiciones	Intuitivo, orientado a empresas y desarrolladores
Almacenamiento	Moderada (uso centrado en proyectos)	Moderado (uso centrado en proyectos)	Almacenamiento en la nube, integración con S3
Despliegue de modelos	Limitado (requiere más trabajo)	Requiere conocimientos avanzados	Requiere más trabajo (requiere conocimientos en DevOps)
Usuarios Objetivo	Estudiantes, investigadores, desarrolladores	Científicos de datos, competidores	Empresas, desarrolladores, científicos de datos
Funciones adicionales	Acceso a herramientas de Google (Drive, Sheets, etc.)	Competencias y herramientas integradas con servicios de AWS	Acceso a herramientas avanzadas para ajuste de modelos

f. Estrategias en proyectos de Machine Learning

Para la creación de soluciones de Aprendizaje Automático o Machine Learning, hay diversas técnicas que facilitan la realización de estas implementaciones, desde el entendimiento del ambiente hasta la implementación y evaluación del modelo. Dentro de las más destacadas se incluyen CRISP-DM, ML-Canvas y KDD, a continuación, se explican detalladamente cada una.

- ***Cross-Industry Standard Process for Data Mining (CRISP-DM)***

Esta estrategia o método es utilizado para llevar a cabo la orientación de trabajos de minería de datos. Como un enfoque metodológico, CRISP-DM incluye una descripción de las etapas habituales de un proyecto, las actividades requeridas en cada

etapa y una explicación de las conexiones entre las tareas. Como modelo de proceso, proporciona un resumen del ciclo de vida de la minería de datos y el aprendizaje automático o Machine Learning [34].

El ciclo de vida del modelo consta de seis etapas con flechas que señalan las relaciones más relevantes y habituales entre cada una de las fases. La disposición de las etapas no es rigurosa. En realidad, la mayoría de los proyectos progresan y retornan entre etapas si se requiere. es versátil y se pueden ajustar con facilidad. CRISP-DM facilita la creación de un modelo de minería de datos que se ajuste a sus requerimientos específicos [34]. La Figura 22 ilustra el flujo de trabajo propuesto por esta metodología.

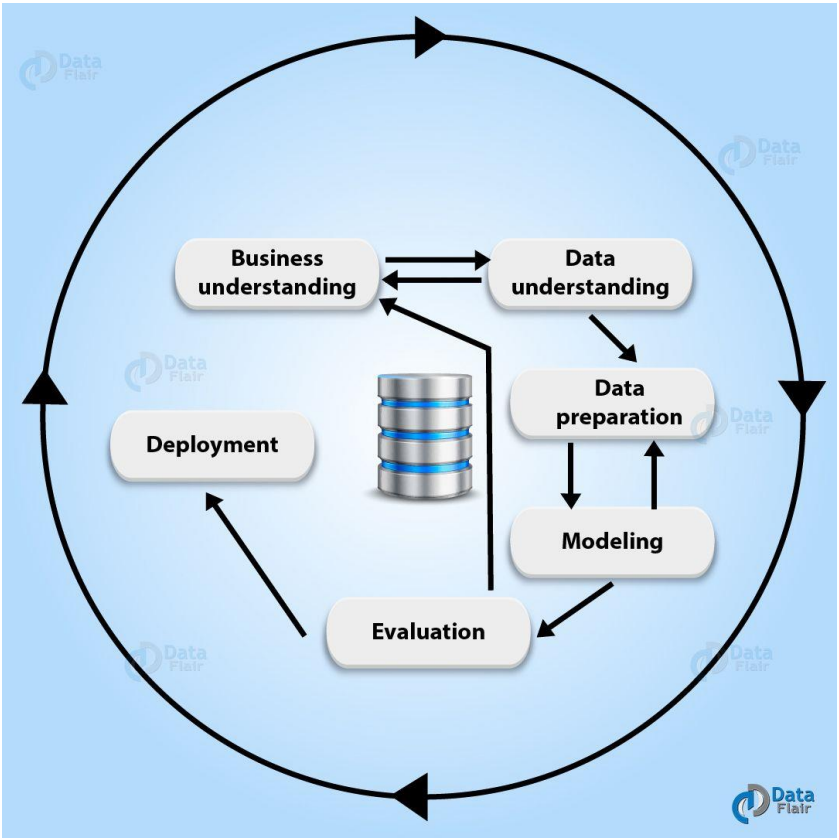


Figura 22. Flujo de trabajo propuesto por la metodología CRISP-DM [35].

Bussiness Understanding. Esta fase intenta comprender los propósitos del proyecto de manera efectiva como las necesidades desde una perspectiva empresarial y transformar dichas necesidades en un plan de trabajo para el aprendizaje automático [36].

Data Understanding. Inicialmente, esta etapa inicia con la recolección de datos preliminares. En esta fase, se requiere de la recopilación de la información de las fuentes existentes para conocerlas mejor. Para llevar a cabo la recolección de información, se deben ejecutar diversas tareas, tales como la carga de datos e integración de estos. A continuación, es necesario analizar y comunicar las características "gruesas" o "superficiales" de los datos recolectados. Después, es imprescindible analizar las demandas de los datos y responder a las interrogantes de minería de datos, las cuales pueden ser abordadas a través de consultas, informes y visualización. Finalmente, es necesario analizar la calidad de los datos para responder a algunas cuestiones relevantes, tales como: "¿Están los datos obtenidos en su totalidad?" y "¿Existen valores ausentes en los datos obtenidos?" [35].

Data Preparation. En este proceso de recolección de datos, se dedica el 90% del tiempo, puesto que el mismo requiere una gran cantidad de datos para poder generar un modelo de reconocimiento adecuado. Además, su resultado final se convierte en el conjunto de datos final. Después de identificar las fuentes de datos, tenemos que escoger, limpiar, construir y darle el formato que se desee. Durante esta etapa, es necesario llevar a cabo una tarea de exploración de datos más detallada para detectar patrones, que deben fundamentarse en el entendimiento del negocio [35].

Modeling. En primer lugar, se necesita elegir las técnicas de modelado que emplearemos para el conjunto de datos ya preparado. Después, se requiere crear un contexto de prueba para confirmar la calidad y la validez del modelo. A continuación, empleando los instrumentos de modelado, tenemos que generar uno o varios modelos basados en el conjunto de datos. Finalmente, es vital incluir a los interesados para valorar los modelos desarrollados, garantizando que estos se ajusten a las iniciativas empresariales [35].

Evaluation. Específicamente, en esta situación, es necesario valorar el resultado dentro del marco del propósito empresarial. En esta etapa, a causa de la emergencia de nuevos patrones, emergen nuevos requerimientos corporativos. Esos patrones deben ser identificados en los resultados del modelo o mediante otros elementos. Comprender el negocio es un procedimiento iterativo en la extracción de datos. La elección de

seguir o no debe realizarse en esta etapa para progresar hacia la etapa de implementación [35].

Deployment. La información adquirida mediante el procedimiento de minería de datos debe ser expuesta de tal forma que los interesados puedan hacer uso de ella cuando lo requieran. De acuerdo con las necesidades del negocio, la etapa de implementación puede implicar la elaboración de un reporte. Además, el procedimiento puede ser tan intrincado como un proceso de extracción de datos repetitivo a lo largo de toda la entidad. En este plan de ejecución, es necesario desarrollar tácticas para el mantenimiento y el respaldo futuro. Desde la perspectiva del proyecto, el informe final debe sintetizar las vivencias del proyecto y evaluar lo ejecutado para detectar áreas de mejora y enseñanzas adquiridas. El CRISP-DM proporciona un esquema homogéneo para documentar la experiencia y las directrices. Adicionalmente, CRISP-DM puede implementarse en varias industrias con distintos tipos de datos [35].

- ***Machine Learning Canvas (ML-Canvas)***

Este esquema de trabajo fue desarrollado por Louis Dorard en 2015 y se utiliza para conceptualizar y organizar proyectos de aprendizaje automático. Este canvas cuenta con 10 cuadros para considerar. En términos generales, se pueden distinguir como cuatro secciones principales: primero establecer el propósito del sistema, determinar el modo en que el modelo aprende, cómo se llevan a cabo las predicciones y, finalmente, la forma de la evaluación [37]. A continuación, se presenta el flujo de trabajo de esta metodología.

Goal. Se trata de la propuesta de valor, es decir, dar respuesta a las interrogantes. ¿Qué es lo que se pretende realizar? ¿Qué relevancia tiene? ¿A quién se pretende favorecer? [38].

Learn. Aquí se establece la fuente y la recopilación de datos, además se determina la entrada del modelo basándose en los datos en bruto. De igual forma, en estas secciones se establecen los periodos de generación de modelos y su actualización con nuevos datos de entrenamiento [38].

Predict. En esta parte se establece la tarea de aprendizaje automático a emplear, además se define el método para procesar las predicciones y se evalúa el modelo utilizando métricas de evaluación antes de su implementación [38].

Evaluate. Esta fase supervisa y evalúa el modelo en directo [38].

- ***Sampling, Exploring, Modifying, Modeling, and Assessing (SEMMA)***

SEMMA, establecida por el Instituto SAS en 2012, es una metodología orientada a la Minería de Datos que define un proceso que comprende esas cinco etapas. El flujo de trabajo de esta metodología se ilustra en la Figura 23 [38].

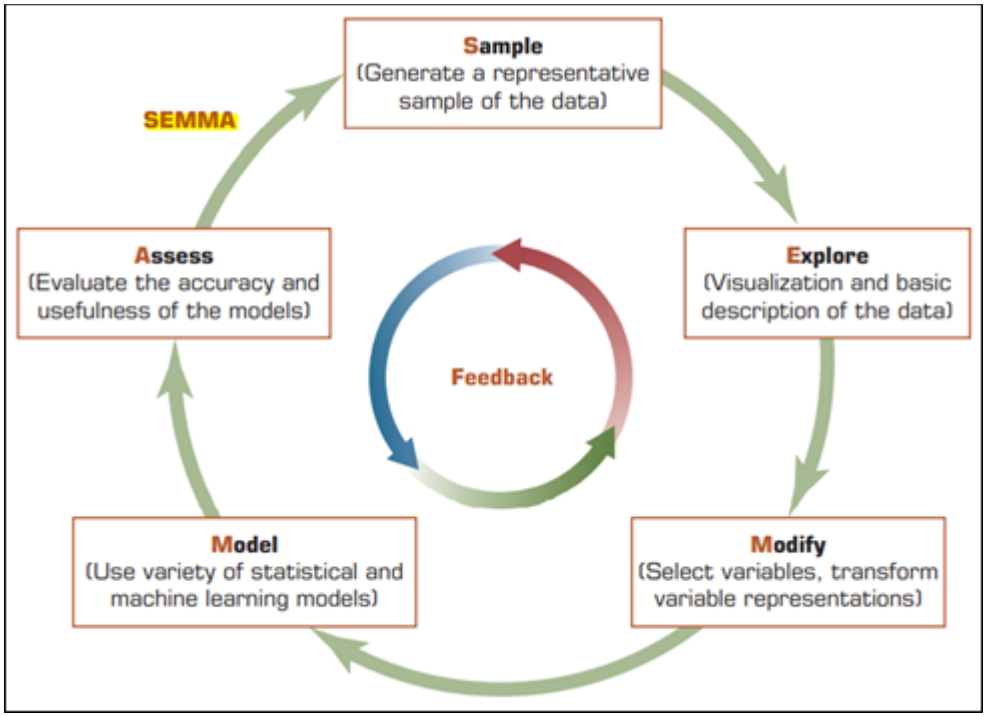


Figura 23. Flujo de trabajo de la metodología SEMMA [39].

Sample. Esta etapa implica la recopilación de datos, enfocándose en el tamaño de las muestras, de tal forma que el conjunto de datos sea lo bastante amplio para ser representativo, pero al mismo tiempo lo suficientemente reducido para facilitar su procesamiento [38].

Explore. En esta etapa, el objetivo es comprender la esencia de los datos a través de la visualización, agrupación y otros métodos, permitiendo la observación de relaciones,

tendencias y cualquier otra información que proporcione un panorama general de los datos y la conexión entre estos [38].

Modify. Esta etapa tiene como objetivo preprocesar los datos para que queden preparados para el futuro modelado a través de transformaciones, escogiendo variables e incluso variables obteniéndolas de datos en bruto [38].

Model. Esta fase establece el modelo más apropiado para satisfacer las necesidades del proyecto, escogiendo entre las familias existentes y ajustando el modelo basándose en los datos y las conclusiones necesarias [38].

Asses. Analizar el desempeño del modelo aplicado en términos de confiabilidad, utilidad, entre otros aspectos [38].

Una vez que se ha descrito cada una de estas tres estrategias o metodologías para Machine Learning, la Tabla 5 compara estas metodologías con criterios técnicos propios de cada una de ellas.

Tabla 5. Análisis comparativo de enfoques para proyectos de Machine Learning [36–39].

Criterio	**CRISP-DM**	**ML-Canvas**	**SEMMA**
Tipo	Metodología estructurada	Plantilla visual orientada	Enfoque flexible y dinámico
Enfoque	Secuencial e iterativo	Enfoque colaborativo y visual	Adaptado a la manipulación de datos
Fases o etapas	Entendimiento del negocio, Análisis de datos, Preparación de datos, Modelado, Evaluación, Implementación.	Objetivos, Aprendizaje, Predicción, Evaluación	Muestra, Exploración, Modificación, Modelado, Evaluación
Flexibilidad	Moderada en la fase de datos y modelado	Alta en visualización y ajustes	Alta, con enfoque en procesos de modelado
Público objetivo	Científicos de datos y analistas	Equipos de trabajo multidisciplinarios	Profesionales técnicos en análisis de datos
Uso de documentación	Detallada y estructurada, centrada en procesos	Documentación intermedia, enfoque visual	Adaptable y técnica
Iteración	Posible pero menos frecuente	Muy alta, favorece la iteración	Alta, con enfoque explícito en ajustes
Facilidad de uso	Técnica pero accesible	Fácil de usar, visualmente accesible	Moderada, requiere conocimientos técnicos
Énfasis en la evaluación	Fuerte enfoque en la evaluación	Moderado, prioriza la interacción	Evaluación centrada en la exploración de datos

g. Criterios de evaluación de modelos

Al elegir una métrica de evaluación, siempre debe tenerse en cuenta el propósito final de la aplicación, no solo enfocándose en realizar predicciones exactas, sino también en emplear estas predicciones en un proceso de toma de decisiones más extenso. Se describen a continuación las métricas de evaluación más empleadas en modelos de aprendizaje automático en actividades como la clasificación y la regresión [25].

- ***Accuracy o exactitud.***

La exactitud es el número de predicciones acertadas dividido por el número total de muestras, es decir, todas las entradas de la matriz de confusión sumadas [25]. Puede manifestarse o expresarse de la siguiente forma:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (3)$$

En este caso:

TP: Verdaderos positivos

TN: Verdaderos Negativos

FP: Falsos Positivos

FN: Falsos Negativos

- ***Precisión***

La precisión determina la cantidad de muestras que se predicen como positivas son en realidad positivas, se emplea como un indicador de rendimiento cuando el propósito es reducir la cantidad de falsos positivos. También se denomina valor predictivo a la precisión vigente (VPP) [25]. Es posible calcularlo de la siguiente forma:

$$Precisión = \frac{TP}{TP + FP} \quad (4)$$

- ***Matriz de confusión***

Dentro de las formas más detalladas de representar el resultado de la evaluación de la clasificación se lleva a cabo mediante matrices de confusión, donde las filas están vinculadas con las clases verdaderas y las columnas con las clases proyectadas. Cada anotación registra la frecuencia con la que una muestra que pertenece a la clase correspondiente a la fila se categorizó como la clase correspondiente a la columna [25]. El diagrama de confusión de un modelo de clasificación binaria se muestra en la Figura 24.

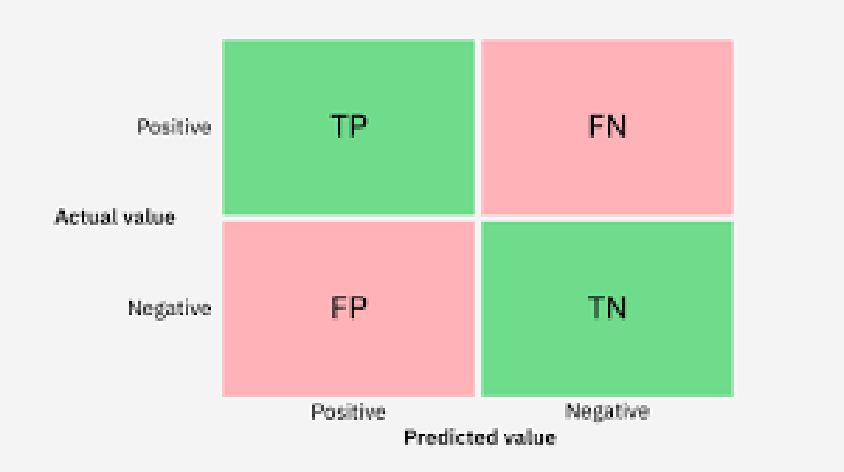


Figura 24. Diagrama de confusión de un modelo de clasificación binaria [40].

En el escenario de clasificación multiclase, es precisamente la misma con variación en la dimensión de la matriz. Considerando el número de clases que el modelo puede clasificar, se busca que la mayor concentración de predicciones se encuentre en la diagonal central de la misma [25].

- ***Recall***

La recuperación o "recall" determina la cantidad de muestras positivas que son capturadas por las predicciones positivas, se emplea como indicador de rendimiento cuando es necesario reconocer todas las muestras positivas; en otras palabras, cuando

es crucial prevenir los falsos negativos. Además, se le denomina sensibilidad, tasa de aciertos o tasa de positivos verdaderos (TPR) [25]. Es posible calcularlo de la forma siguiente:

$$Recall = \frac{TP}{TP + FN} \quad (5)$$

- ***F-Score***

La precisión y la recuperación son medidas de gran relevancia, aunque miden puntos de vista particulares del modelo. Una manera de unificar estas métricas es la medida f, también denominada "F-score, que representa la media armónica de la exactitud y la recuperación. "F1 puntuación" es una de las alternativas más empleadas [25]. Es posible calcular de la forma siguiente:

$$f1\ score = 2 * \frac{Precisión * Recall}{Precisión + Recall} \quad (6)$$

1.3.9 Deep Learning

El Deep Learning es un conjunto de métodos o técnicas de aprendizaje que intentan modelar datos con arquitecturas complejas con la combinación de diferentes transformaciones no lineales. Los bloques básicos elementales del Deep Learning son las redes neuronales, que se combinan para formar las redes neuronales profundas [41].

Estas metodologías han facilitado progresos importantes en los ámbitos del sonido y el tratamiento de imágenes, que incluyen el reconocimiento facial, el reconocimiento del habla, la visión por computadora, el tratamiento automatizado del lenguaje y la categorización de textos (como el reconocimiento de correo no deseado) [41].

Existen varios tipos de arquitecturas para las redes neuronales:

***Los perceptrones multicapa**; l*os más antiguos y sencillos.

Las redes neuronales convolucionales (CNN); adaptadas al tratamiento de imágenes.

Las redes neuronales recurrentes; utilizadas para datos secuenciales como texto o series temporales.

Se basan en una profunda cascada de capas. Necesitan algoritmos inteligentes de optimización estocástica e inicialización, así como una elección inteligente de la estructura. Sus resultados son impresionantes, aunque hasta ahora se dispone de muy pocos fundamentos teóricos [41].

a. Red neuronal convolucional.

Las CNN por sus siglas en inglés Convolutional Neuronal Network tienen como objetivo actuar en función de las imágenes de forma muy parecida a la corteza primaria visual del cerebro humano. Los humanos clasifican y distinguen de manera perfecta entre varios objetos, gracias a su habilidad para identificar diferentes características, tales como el color, los bordes, las curvas, las sombras; en otras palabras, se concentran en todo lo que les permite diferenciar o categorizar cada objeto de forma individual. Por lo tanto, gran cantidad de estos sistemas buscan replicar el funcionamiento de las neuronas de la corteza visual [42].

Las CNN operan tanto en imágenes 2D como en 3D y se componen de tres capas: capa de convolución, capa de reducción (max pooling) y capas densas (fully connected); como se observa en la Figura 25. De estas capas, las dos primeras realizan una etapa de extracción de características, mientras que la última (fully connected) es la responsable de llevar a cabo la etapa de clasificación [42].

Las primeras capas de una CNN facilitan la extracción de características, un procedimiento que replica cómo el cerebro humano reconoce las características fundamentales de un objeto. Durante la etapa de clasificación o regresión, que sucede en las capas densas, se vinculan todas las propiedades previamente obtenidas para conseguir la clasificación definitiva. Por ejemplo, si se forma una CNN para identificar perros, en la fase inicial (convolución y max-pooling) se obtienen atributos fundamentales del animal, tales como su pelaje, color y forma. Después, en la segunda

etapa (fully connected), se emplean estas propiedades para establecer si la imagen posee o no analogías con un perro [42].

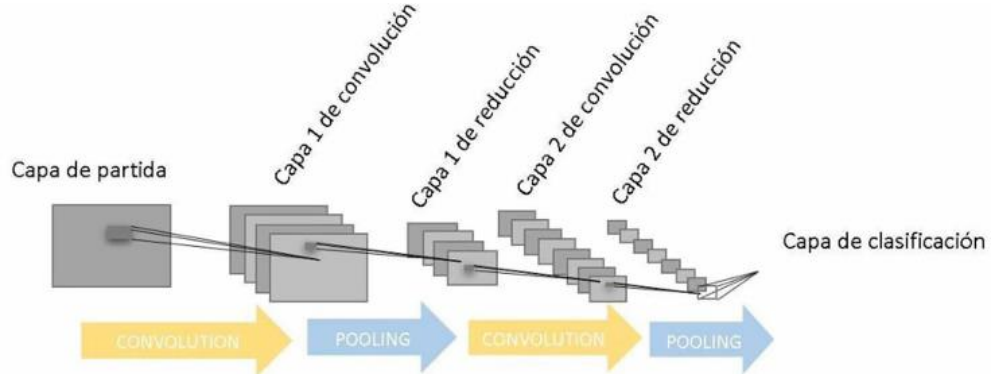


Figura 25. Desarrollo de una red neuronal convolucional. Capa de partida (imagen inicial, o entrada) [42].

- ***Conceptos claves de las redes neuronales***

Pixel. Proviene del inglés picture element, "elemento de imagen", es un punto único o un cuadrado pequeño que posee color; puede ser en este caso blanco, negro o matrices de grises en el caso de las imágenes de diagnóstico y constituye una imagen digital [42]. En la Figura 26 se visualiza el esquema de un pixel.

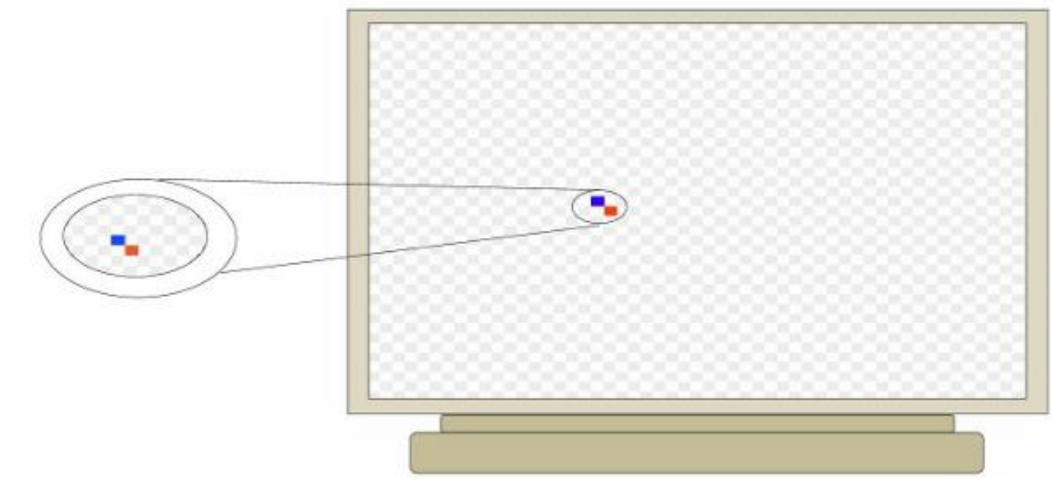


Figura 26. Esquema de un pixel [42].

Kernel. Es conocido como feature detector, filter o en español como "detector de propiedades", la misma es una matriz compuesta por números que son utilizados para

enfocar, desenfocar, para realizar grabaciones, para la detección de bordes y para determinar características de la imagen en cuestión [42].

Procesamiento. Una imagen digital es una matriz que se guarda en el ordenador, en la que, en el caso de imágenes en escala de grises (imágenes radiológicas), cada uno de los pixeles se representa con un valor que oscila entre 0 (tonalidades oscuras) y 255 (tonalidades claras o luminosas). La normalización de la información se refiere a la reducción del valor del pixel en 255. Esta transformación atribuye valores de pixeles entre 0 y 1, lo que facilita la reducción de la carga informática y la creación de una red neuronal más eficiente en el aprendizaje, lo que refleja positivamente en las proyecciones de las imágenes [42]. En la Figura 27 se cuenta con el preprocesamiento de una imagen digital.

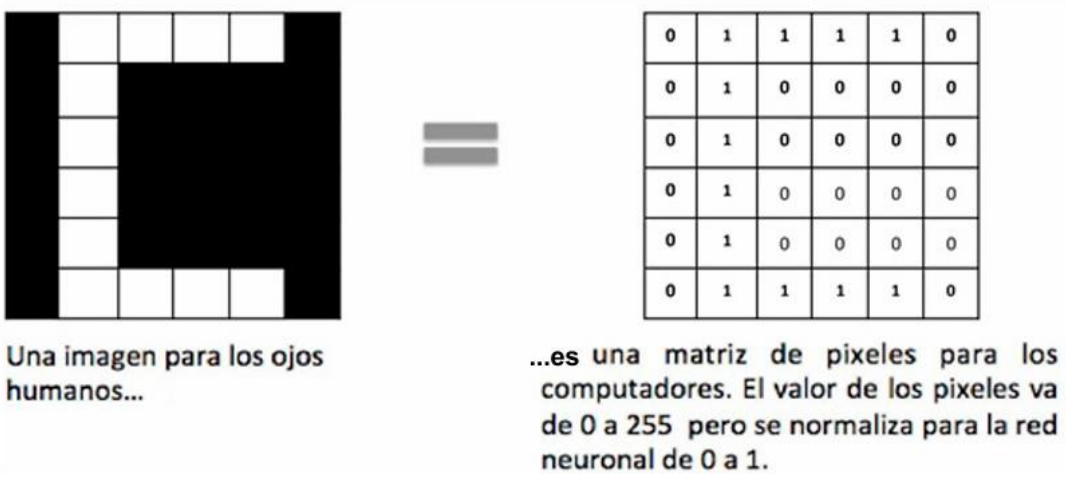


Figura 27. Preprocesamiento [42].

- ***Funcionamiento de la CNN***

Capas de convolución. La función principal de la convolución es que con la ayuda de un filtro (kernel), el mismo que se aplica a la imagen, extrae determinadas características. Entonces, si se cuenta con un filtro que detecta bordes dentro de la imagen, lo que se observa en la convolución es una imagen como la Figura 28. Por lo general de la convolución es la aplicación de un patrón a una imagen, con el fin de extraer características específicas o patrones dentro de aquella imagen [42].



Figura 28. Convolución [42].

El proceso de convolución matemática en redes neuronales se lleva a cabo mediante la aplicación de un filtro o núcleo, de tamaño 3×3, sobre una imagen de 6×6 píxeles. Este filtro recorre la imagen y lleva a cabo una serie de operaciones, creando un mapa de atributos relevantes. En cada ciclo, el filtro multiplica sus coeficientes por los píxeles equivalentes de la imagen, y posteriormente agrupa los resultados para producir un nuevo píxel en la imagen resultante. El filtro se mueve por la imagen de izquierda a derecha, y al alcanzar el extremo derecho, baja una posición y vuelve a realizar el procedimiento hasta abarcar toda la imagen. Finalmente, se logra la imagen obtenida, que posteriormente se traslada a la capa de reducción [42]. Todo lo descrito se lo visualiza en la Figura 29.

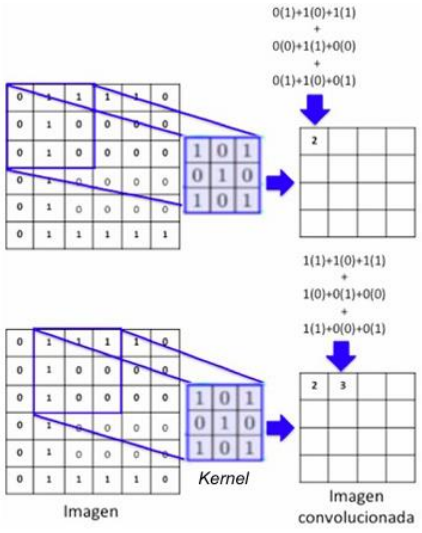


Figura 29. Función del Kernel: La imagen que se genera se reduce con rasgos que predominan en comparación con la imagen de origen [42].

Capa max pooling o capa de reducción. En una CNN, la capa de reducción disminuye el flujo de datos producido por las capas de convolución, obteniendo las propiedades más relevantes de la imagen y suprimiendo el ruido. Esto contribuye a reducir la carga informática y facilita que la red se entrene y segmente de manera más eficiente. A pesar de que la capa de max pooling disminuye el ruido, su empleo excesivo puede suprimir elementos significativos de la imagen. La meta principal de esta capa es disminuir la dimensión de la imagen manteniendo a la vez sus características fundamentales. Tras la convolución y la disminución, la imagen final es más reducida y limpia, manteniendo las activaciones más relevantes. Por ejemplo, una ilustración de 4×4 podría ser disminuida a 2×2 [42]. En la Figura 30 se cuenta con una imagen convolucionada, en base al ejemplo recientemente descrito.

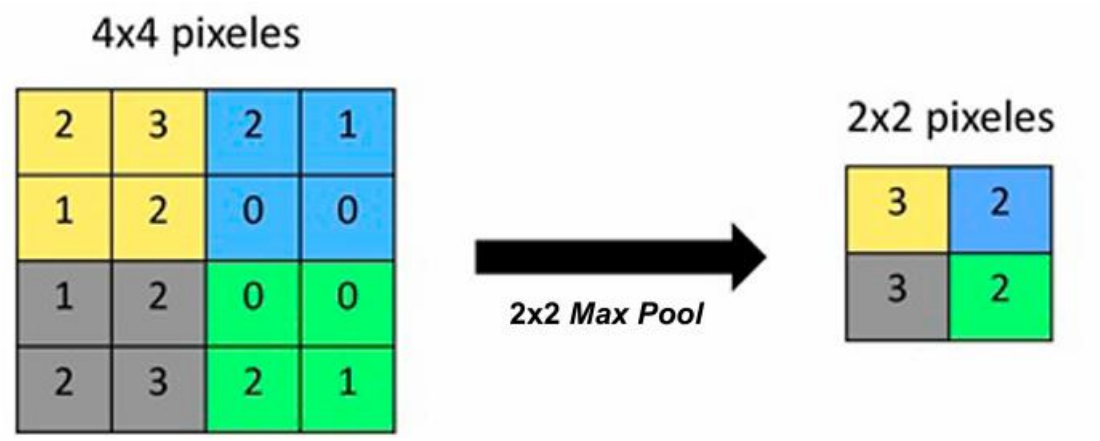


Figura 30. Imagen convolucionada del ejemplo anterior con un tamaño de pool o filtro [42].

A continuación, en la Figura 31 se aprecia el esquema de entrenamiento de una red neuronal convolucional.

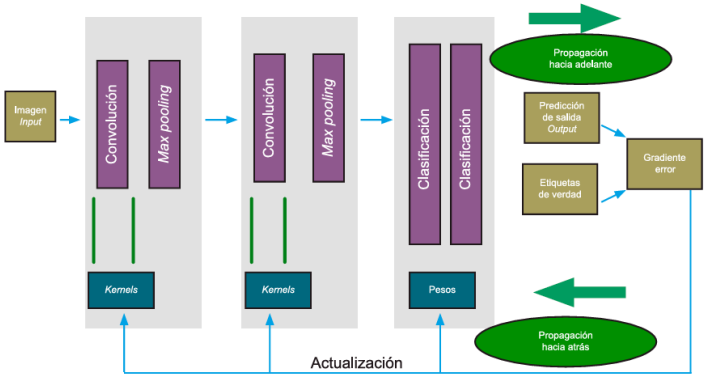


Figura 31. Esquema de entrenamiento de una red neuronal convolucional [42].

Capas densas o capas de clasificación (red neuronal multicapa o fully connected). Las capas totalmente vinculadas determinan la puntuación de cada tipo de características obtenidas en las capas de convolución y máxima agrupación, y las vinculan para lograr la clasificación final. A pesar de que no existe un número específico de capas totalmente conectadas, la literatura indica la utilización de entre 2 y 4 capas, tal como sucede en LeNet, VGG Net y AlexNet. En estas capas, cada neurona desempeña una función no lineal, cuya selección se basa en la tarea que debe llevar a cabo [42].

- ***Entrenamiento de una CNN.***

El proceso de entrenamiento de una red neuronal tiene como objetivo modificar los kernels en las capas de convolución y los pesos en las capas de clasificación para reducir la discrepancia entre las predicciones de salida de la CNN y las etiquetas verdaderas (respuestas acertadas). En campos como el de la radiología, esto implica lograr diagnósticos precisos de las imágenes. El proceso de aprendizaje en una CNN es iterativo, iniciando con la propagación hacia adelante (forward propagation), en la que la información se desplaza por las capas para producir una proyección, que posteriormente se contrasta con la etiqueta de verdad. Basándose en esto, se determina el gradiente de error, que modifica los parámetros (kernels y pesos) a través de la propagación retrospectiva (back propagation). Este procedimiento de ajuste se realiza de manera iterativa hasta rectificar los fallos de la red. El entrenamiento se lleva a cabo en diversas etapas, en las que se procesan los datos de la base de datos varias veces para modificar los parámetros de la red [42]. La Figura 32 plasma un esquema general y resumido del funcionamiento de un modelo CNN con tres capas convolucionales.

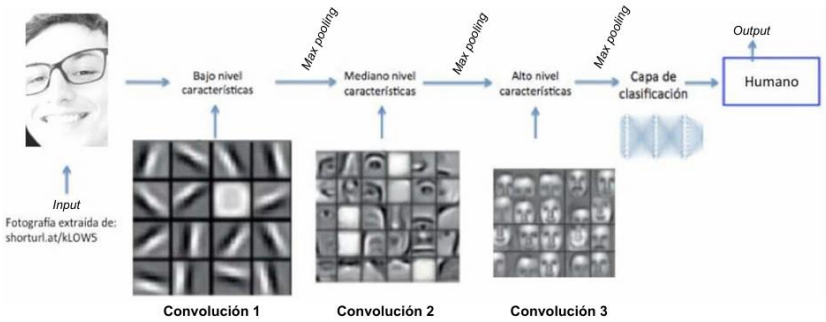


Figura 32. Esquema general y resumido del funcionamiento de un modelo CNN con tres capas convolucionales [42].

1.3.10 Visión Artificial.

A la visión artificial también se la denomina Visión por computador, y puede ser definida como un proceso en el que se extrae información proveniente del mundo físico a partir de imágenes. Desde el punto de vista ingenieril, un sistema de visión artificial es un sistema autónomo que ejecuta varias de las tareas que el sistema de visión humano realiza. La visión artificial facilita la realización de funciones que abarcan desde la identificación de objetos sencillos hasta la interpretación de escenas complejas en 3D. Esta tecnología es esencial para definir el vínculo entre el mundo tridimensional y sus representaciones bidimensionales capturadas por aparatos de visión [43]. La Figura 33 muestra las etapas a considerar dentro de un proceso de visión artificial.

La estructuración recientemente presentada es muy completa pero no quiere decir que cualquier proceso de visión artificial tenga necesariamente que pasar por cada una de aquellas. Tenemos por ejemplo, dentro de un sistema estéreo o de seguimiento visual, etapas de segmentación, la extracción de características y reconocimiento no suelen ser indispensables, mientras que si se necesita de la implantación de otras determinadas o específicas técnicas [43].

A continuación, se detalla cada una de las etapas de la Figura 33.



Figura 33. Etapas a considerar dentro de un proceso de visión artificial [43].

Las etapas del tratamiento digital de imágenes son las siguientes: Adquisición de la imagen, que tiene por objeto representar o mostrar el mundo real tridimensional dentro de una imagen digital; Preprocesamiento, que incluye diversas operaciones destinadas a preparar la imagen para etapas posteriores, como la eliminación de ruidos y la calibración de realce; Detección de bordes, que es importante en muchos procesos visuales porque permite extraer los bordes que componen los objetos de la imagen; Segmentación, que tiene por objeto identificar las zonas dentro de la imagen en las que los píxeles comparten algún tipo de atributo; Extracción de características, que da lugar a una representación matemática de los objetos previamente segmentados; Reconocimiento: los elementos que aquí se clasifican como pertenecientes a la clase o prototipo cuyas características coinciden con las del objeto; la localización es el proceso de ubicar el objeto dentro del espacio 3D, e implica el uso de técnicas de triangulación (que pueden ser activas o pasivas) o de restricción espacial basadas en el conocimiento de la escena; por último, la interpretación es el proceso de interpretar la escena utilizando la información recopilada en las etapas anteriores, teniendo en cuenta la relación entre los objetos simples que ya han sido identificados y localizados [43].

a. Detección de objetos

La detección de objetos representa una de las labores más esenciales y complicadas en el ámbito de la visión artificial, con usos en sectores como la robótica, la conducción autónoma, la seguridad, la supervisión y la realidad aumentada. En términos básicos, la detección de objetos alude a la habilidad de un dispositivo para reconocer y ubicar ejemplos concretos de objetos en una imagen o secuencia de video. Históricamente, este problema se afrontaba a través de métodos de procesamiento de imágenes como los clasificadores en serie y la utilización de características diseñadas manualmente. No obstante, los progresos en el ámbito del aprendizaje profundo han transformado la identificación de objetos, facilitando la creación de modelos más exactos y eficaces [44].

Uno de los métodos más impactantes en la identificación de objetos es el sugerido por Girshick y colaboradores en 2014, en su estudio "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation". Este estudio presentó el algoritmo de redes neuronales convolucionales basadas en regiones (R-CNN), que

fusiona las neuronas convolucionales con una jerarquía de atributos ricos para incrementar la exactitud en la identificación y segmentación de objetos. La arquitectura R-CNN posibilita obtener atributos de las áreas sugeridas en las imágenes, lo que facilita la identificación y categorización de objetos en escenas complejas de forma más eficaz que los métodos previos [44].

R-CNN ha influido notablemente en el progreso de la visión computacional, debido a su método revolucionario y su alto desempeño en benchmarks de detección de objetos, como el conjunto de datos PASCAL VOC. Desde su lanzamiento, ha sido el fundamento para otros progresos, como Fast R-CNN y Faster R-CNN, que han perfeccionado y acelerado el proceso de detección, aproximando aún más a la implementación en tiempo real[44].

- ***Yolo***

Yolo es un sistema que hace uso de una red neuronal convolucional dedicado a la detección de objetos en tiempo real, este sistema aplica una sola red neuronal a la imagen completa y por este motivo es muy rápido. Aquella red se basa en dividir la imagen en regiones y posteriormente en predecir múltiples cajas delimitadoras, que en inglés se conoce como “bounding box”, y la probabilidad de detección de las clases de entrenamiento para cada caja delimitadora. Finalmente, Yolo utiliza un método de supresión de no máximos para lograr suprimir múltiples detecciones del mismo objeto. Dentro del resultado que presenta este sistema está la impresión de los objetos detectados, la confidencia y el tiempo de ejecución [45]. En la Figura 34 se presenta la ejecución del sistema aplicado sobre una imagen de entrada.

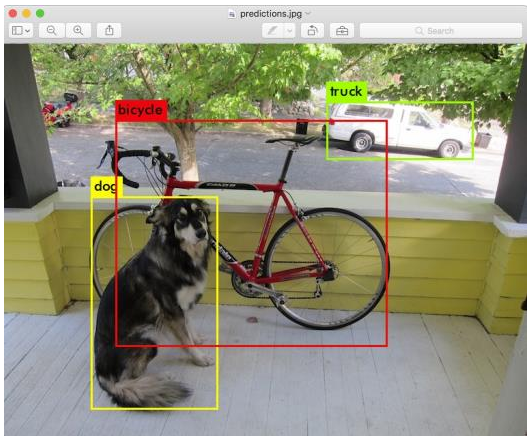


Figura 34. Detección de objetos con YOLO [45].

Yolo (You Only Look Once) hace uso de Deep Learning y CNN para llevar a cabo la detección de objetos, y se diferencia de sus “competidores” porque es el más rápido de todos, aunque sacrifica un poco de exactitud, por lo cual esta rapidez facilita la detección de objetos en tiempo real con hasta 30 FPS [46].

Para la ejecución de la detección de objetos, primeramente, se divide la imagen en una cuadrícula de SxS (tal como se ve en la imagen de la izquierda de la figura 35), posteriormente en cada una de las celdas se hace una predicción de N posibles “bounding boxes” y se realiza el cálculo del nivel de certidumbre o de probabilidad de cada una de ellas (tal como se ve en la imagen del centro de la figura 35), es decir, se realiza el cálculo mediante la multiplicación de SxSxN diferentes cajas, la gran parte de ellas con un nivel de certidumbre muy bajo. Una vez obtenidas estas proyecciones, se procede a suprimir las cajas que se encuentren por debajo de un límite establecido. Se aplica un procedimiento de "no máxima supresión" a las cajas restantes, que tiene como objetivo eliminar posibles objetos que se detectaron en duplicado, dejando solo el más preciso de ellos (imagen de la derecha de la Figura 35) [46].

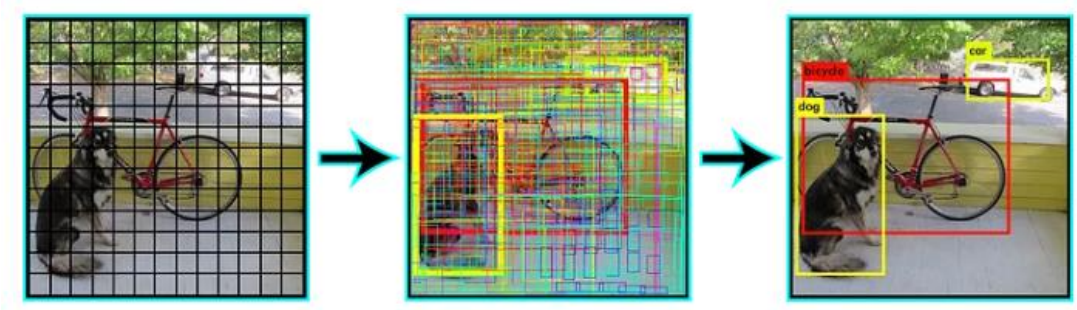


Figura 35. El funcionamiento de YOLO para la identificación de objetos [46].

Dentro de la documentación de Ultralytics se ofrece soporte hacia una gran variedad de modelos o versiones de Yolo, cada una de estas versiones se encuentran adaptados a tareas específicas como es el caso de la detección de objetos, clasificación de imágenes, segmentación de instancias, entre otras. En la Figura 36 se muestra una gráfica de evaluación comparativa sobre varios modelos de detección de objetos basados en Yolo, regidos en términos de dos métricas: la primera mAP (mean Average Precision) y la segunda por medio de la latencia (ms por imagen). Dentro del eje vertical se ubica el mAP con el conjunto de datos COCO que realiza mediciones de precisión promedio de los modelos con respecto a la detección de objetos. Y

finalmente, dentro del eje horizontal aparece la latencia de los modelos y se mide en milisegundos por imagen que se procesa [47].

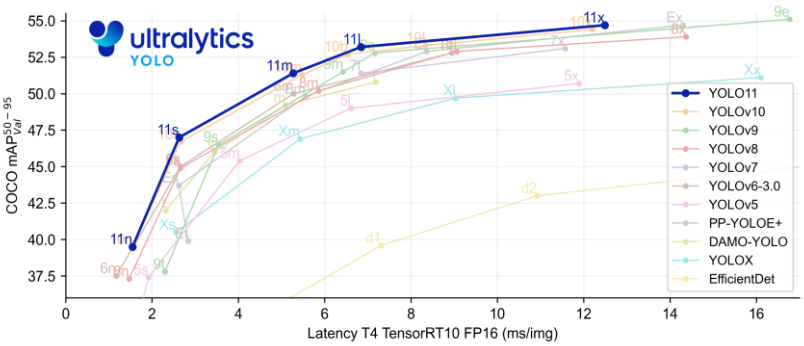


Figura 36. Comparación del rendimiento de diferentes versiones de YOLO en términos de mAP (precisión promedio) y latencia en milisegundos por imagen en el conjunto de datos COCO [47].

La Tabla 6 realiza una pequeña descripción de lo más importante o destacable de cada una de las versiones de Yolo.

Tabla 6. Descripciones de las versiones de Yolo más importantes [47].

Versión de Yolo	Descripción
YOLOv3	La tercera versión de la serie de modelos YOLO, creada por Joseph Redmon, reconocida por su eficiente capacidad de detectar objetos en tiempo real.
YOLOv4	Una mejora sobre la arquitectura de YOLOv3, publicada por Alexey Bochkovskiy en 2020.
YOLOv5	Una versión optimizada de la arquitectura YOLO desarrollada por Ultralytics, que proporciona mejores rendimientos y una mejor relación entre velocidad y precisión en comparación con versiones anteriores.
YOLOv6	Presentado por Meituan en 2022, utilizado en sus robots de entrega autónomos.
YOLOv7	Modelos actualizados de YOLO lanzados en 2022 por los creadores de YOLOv4.
YOLOv8	Un modelo avanzado con mejoras como la segmentación de instancias, la estimación de poses y puntos clave, además de la clasificación.
YOLOv9	Un modelo experimental basado en el código de YOLOv5 de Ultralytics que incorpora la Información de Gradiente Programable (PGI).
YOLOv10	Desarrollado por la Universidad de Tsinghua, este modelo se entrena sin NMS y cuenta con una arquitectura optimizada para eficiencia y precisión, ofreciendo un rendimiento y latencia de alta calidad.
YOLO11	Los modelos más recientes de YOLO de Ultralytics ofrecen un rendimiento de vanguardia (SOTA) en diversas tareas como detección, segmentación, estimación de poses, seguimiento y clasificación.
SAM	Un modelo original de Meta, conocido como Segment Anything.
SAM2	La nueva generación del modelo Segment Anything de Meta, que incluye soporte para vídeos e imágenes.
MobileSAM	La versión móvil del modelo SAM, desarrollada por la Universidad Kyung Hee para aplicaciones móviles.

- ***Faster R-CNN: Un modelo eficiente para la detección de objetos en tiempo real.***

Este sistema diseñado para la detección de objetos está conformado por dos módulos: el primero es una red completamente convolucional profunda que propone regiones, el segundo módulo es exactamente el detector Fast R-CNN que hace uso de las regiones propuestas. El sistema en su totalidad es una red unificada direccionada para la detección de objetos, tal como se observa en la Figura 37 [48].

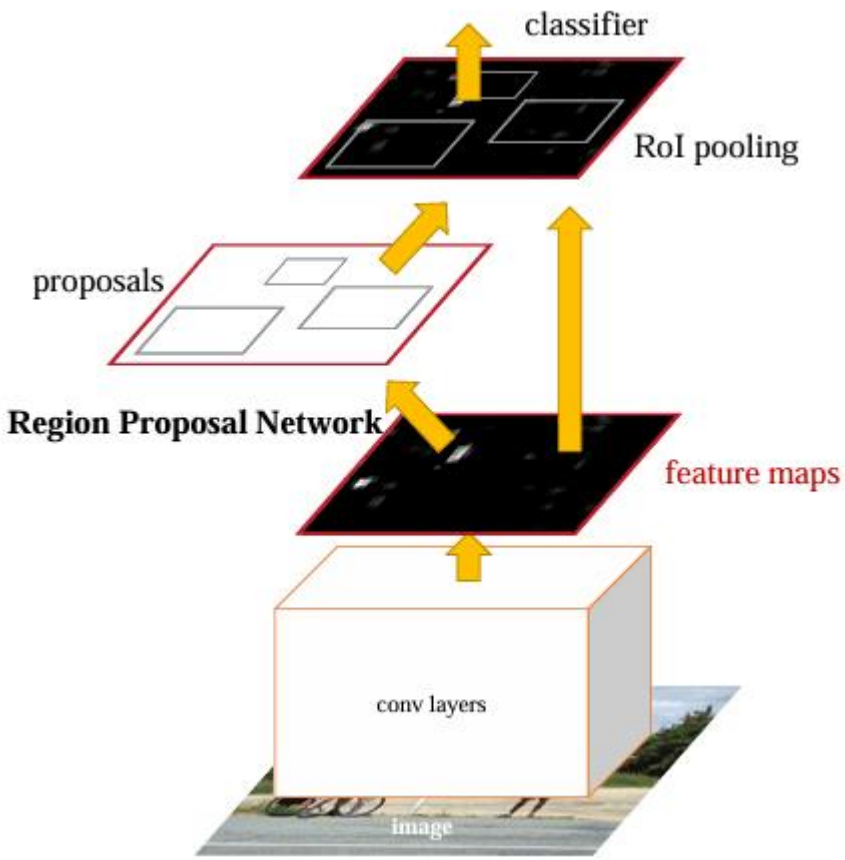


Figura 37. Esquema de una Red de Propuestas de Regiones (RPN) para detección de objetos [48].

Redes de Propuestas de Regiones. Una Red de Propuestas de Regiones (RPN) utiliza una imagen (de cualquier tamaño) como fuente y produce un grupo de propuestas de objetos de forma rectangular, cada una con una calificación de "objetividad". Este procedimiento se representa con una red en su totalidad convolucional, intercambiando capas convolucionales con una red de detección de objetos Fast R-CNN. En los experimentos, se emplearon dos modelos: el modelo de Zeiler y Fergus (ZF) que

comparte 5 capas convolucionales, y el modelo de Simonyan y Zisserman (VGG-16) que comparte 13 capas. Para elaborar las sugerencias de región, se despliega una pequeña red sobre el mapa de propiedades convolucionales producido por la capa convolucional compartida más reciente. Esta red captura una ventana espacial de dimensiones n x n de la imagen inicial y la agrupa a una característica de menor tamaño, que posteriormente se traslada a dos niveles completamente vinculados: una capa de regresión de cajas (reg) y una capa de clasificación de cajas (cls). En esta investigación, se emplea n = 3, notándose que el campo de recepción efectivo en la imagen de entrada es amplio (171 y 228 píxeles para ZF y VGG, respectivamente). La mini-red que opera en una ventana deslizante se establece con una capa convolucional de dimensiones n x n, seguida de dos capas convolucionales de dimensiones 1x1 para reg y cls respectivamente [48]. La Figura 38 muestra el proceso de generación de propuestas de regiones con RPN.

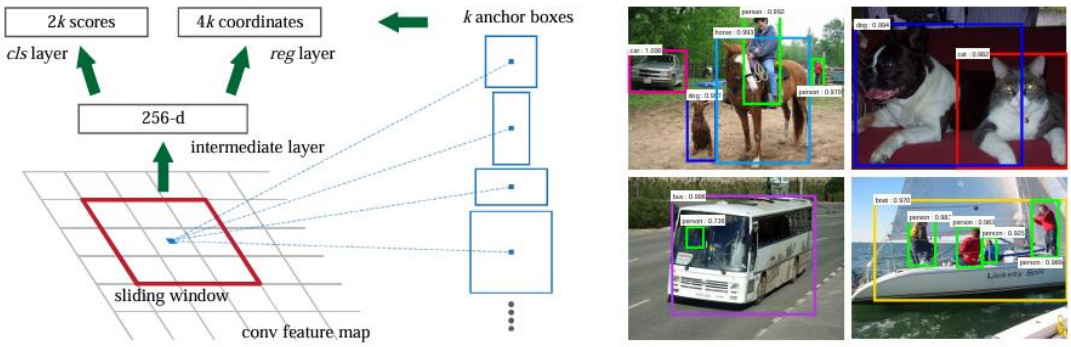


Figura 38. Proceso de generación de propuestas de regiones con RPN [48].

- ***Single Shot MultiBox Detector (SSD)***

El modelo SSD discretiza el espacio de salida de las tablas delimitadoras empleando tablas predefinidas con distintas relaciones de aspecto y escala. En el proceso de predicción, la red crea calificaciones para cada tipo de objeto en cada panel y modifica los paneles para que se alineen con la forma del objeto. Además, fusiona proyecciones de diversos mapas de características con resoluciones variadas para gestionar objetos de variados tamaños. SSD es más sencillo que otros procedimientos que necesitan propuestas de objetos, puesto que suprime la creación de propuestas y las fases de muestreo repetitivo, llevando a cabo todo el cálculo en una única red [49].

Durante el entrenamiento, el procesamiento de SSD solo necesita una imagen de entrada y cuadros de verdad del terreno para cada objeto. Mediante un procedimiento convolucional, se analizan cuadros preestablecidos con variadas relaciones de aspecto en cada localización de diversos mapas de características de diferentes escalas, tales como 8 × 8 y 4 × 4. Para cada cuadro preestablecido, se anticipan los cambios de forma y las confianzas para todas las clases de objetos. Este procedimiento está estructurado en la Figura 39 [49].

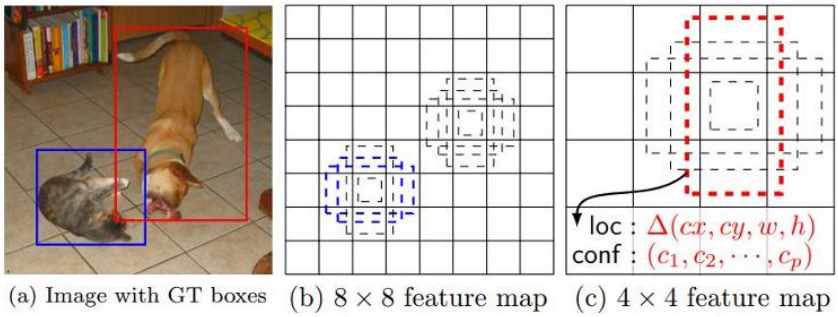


Figura 39. Predicción de cuadros en SSD: (a) Imagen con GT, (b) Mapa 8×8, (c) Mapa 4×4 [49].

La Tabla 7 presenta el análisis comparativo de las arquitecturas detalladas para la identificación de objetos, valorando factores como la rapidez, exactitud, requerimientos de hardware, entre otros.

Tabla 7. Sistemas de identificación de objetos [46–48].

Criterio	**YOLO**	**Faster R-CNN**	**SSD**
Estilo de Arquitectura	Detección en tiempo real	Detección con dos etapas	Detección rápida en tiempo real
Velocidad	45 a 155 FPS	5 a 20 FPS	30 a 60 FPS
Precisión	Buena, con bajo rendimiento en objetos pequeños	Alta, muy precisa	Buena, con rendimiento aceptable en objetos pequeños
Estructura	Modelo único	Dos etapas: propuestas de región y clasificación	Modelo único
Uso de Características	Captura de información global	Propuestas de región de interés	Múltiples capas de características combinadas
Tamaño del Modelo	Compacto	Grande	Moderado
Facilidad de Entrenamiento	Moderada	Requiere recursos altos	Relativamente fácil
Requisitos de Hardware	Bajo	Alto	Moderado
Rendimiento en Videos e Imágenes	Videos e imágenes rápidas	Videos e imágenes moderadas	Videos e imágenes con buena fluidez

b. Unidades de procesamiento de datos

Hay numerosas unidades o módulos de procesamiento que pueden correr modelos de aprendizaje automático para aplicaciones de visión artificial. Sin lugar a duda, el parámetro más relevante es la unidad de procesamiento de gráficos o GPU, que facilita el funcionamiento fluido de modelos más complejos. Se describen a continuación las unidades de procesamiento más habituales en las aplicaciones de visión artificial.

- ***Rasberry Pi 4.***

El Raspberry Pi 4 Modelo B es una tarjeta de alta eficiencia que dispone de un procesador de cuatro núcleos de 64 bits, admite doble pantalla con resoluciones de hasta 4K mediante puertos micro HDMI y posibilita la decodificación de vídeo en 4K a 60 FPS. Proporciona hasta 8 GB de RAM, doble banda de conectividad inalámbrica (2.4/5 GHz), Bluetooth 5.0, Ethernet Gigabit y puertos USB 3.0. Adicionalmente, cuenta con capacidad PoE (Potencia sobre Ethernet) a través de un complemento PoE HAT [50].

El desempeño de este modelo se asemeja al de sistemas de computación de nivel elemental. Conserva la compatibilidad con la versión previa Raspberry Pi 3 Model B+, aunque mejora la rapidez del procesador, el desempeño multimedia, la memoria y la conectividad. La certificación modular de su LAN inalámbrica y Bluetooth disminuye las pruebas de conformidad, lo que optimiza los gastos y tiempos de venta al diseñar productos finales [50]. En la Figura 40 se visualiza una placa electrónica Rasberry Pi 4.



Figura 40. Placa electrónica Rasberry Pi 4 [50].

- ***Banana Pi M2 Zero***

La Banana Pi M2 Zero es una tarjeta gráfica de gran tamaño, con una dimensión de 65x30 mm. Emplea un procesador Cortex A7 Allwinner H3/H2 con 512 MB de memoria RAM. Es perfecta para usos en sistemas con restricciones de espacio y peso, y es apropiada para labores parecidas a las realizadas por la Raspberry Pi Zero W. Apoya Linux y Android, y proporciona conexión WiFi (AP6212) y Bluetooth. Además, posee un puerto mini-HDMI y 40 pines GPIO [51]. En la Figura 41 se visualiza una placa electrónica Banana Pi M2 Zero.



Figura 41. Placa electrónica Banana Pi M2 Zero [52].

- ***NVIDIA Jetson Nano***

El Kit de Desarrollo Jetson Nano de NVIDIA es un ordenador compacto y potente que facilita la ejecución de varias redes neuronales simultáneamente, perfecto para usos como la clasificación de imágenes, la detección de objetos, la segmentación y el procesamiento de voz, todo ello con un consumo energético reducido de apenas 5 vatios. El Jetson Nano, dotado con una CPU ARM de 64 bits de cuatro núcleos, una GPU NVIDIA Maxwell de 128 núcleos y 4 GB de memoria LPDDR4, ofrece 472 GFLOPS para ejecutar algoritmos de Inteligencia Artificial de forma veloz y eficaz, con la capacidad de manejar varios sensores de alta resolución al mismo tiempo [53]. En la Figura 42 se visualiza una placa electrónica NVIDIA Jetson Nano.



Figura 42. Placa electrónica NVIDIA Jetson Nano [53].

La Tabla 8 presenta el análisis comparativo de las distintas unidades de procesamiento de datos presentadas.

Tabla 8. Análisis comparativo de las distintas unidades de procesamiento de datos presentadas [50], [51], [53].

Parámetro Técnico	**Raspberry Pi 4**	**Banana Pi M2 Zero**	**NVIDIA Jetson Nano**
CPU	Quad-core Cortex-A72 @ 1.5 GHz	Quad-core Cortex-A7 @ 1.2 GHz	Quad-core ARM Cortex-A57 @ 1.43 GHz
GPU	Broadcom VideoCore VI	Mali-400 MP2	128-core Maxwell GPU
Memoria RAM	2 GB, 4 GB o 8 GB LPDDR4	512 MB LPDDR2	4 GB LPDDR4
Almacenamiento	MicroSD card slot	MicroSD card slot	MicroSD card slot
Conectividad	2x USB 3.0, 2x USB 2.0, Gigabit Ethernet, Wi-Fi 802.11ac	1x USB 2.0, 10/100 Ethernet, Wi-Fi 802.11b/g/n	Gigabit Ethernet, 4x USB 3.0, 1x USB 2.0 Micro-B
Puertos GPIO	40 pines GPIO	40 pines GPIO	40 pines GPIO
Salida de Video	2x Micro HDMI (4Kp60)	HDMI, Composite Video	HDMI, DisplayPort
Cámara	1x Camera Serial Interface (CSI)	1x Camera Interface (CSI)	2x MIPI CSI-2 DPHY
Consumo de Energía	5V/3A (15W)	5V/2A (10W)	5V/4A (5W)
Sistemas Operativos	Linux (Raspbian, Ubuntu, otros)	Linux (Armbian, Debian)	Linux (Ubuntu, JetPack SDK)
Dimensiones	88mm x 58mm x 19.5mm	60mm x 30mm x 8.8mm	100mm x 80mm x 29mm
Precio Aproximado	$35 (2GB), $55 (4GB), $75 (8GB)	$25-$30	$200 (Jetson Nano 4GB)

1.3.11 Control de la información

Un sistema que cuenta con la capacidad de controlar la información es un software que gestiona tanto el procesamiento de datos dentro de la etapa del backend como su visualización dentro del frontend de un proyecto. Además, se encarga de almacenar esta información para que pueda ser analizada y visualizada en cualquier momento. A continuación, se presentan varias plataformas de desarrollo utilizados para crear este tipo de sistemas.

a. Plataformas de desarrollo para el control de la información.

- ***FastAPI***

FastAPI es framework de Python, compuesto de un conjunto de herramientas, bibliotecas y convenciones predefinidas, mismas que los desarrolladores utilizan para construir aplicaciones web de manera más eficiente y organizada, altamente eficaz para la creación de APIs. Se fundamenta en las sugerencias de tipo de Python, accesibles desde la versión 3.6. Es uno de los marcos de codificación más veloces de Python. Cuando se utilizan las bibliotecas Starlette y Pydantic, su desempeño es superior, equiparable al de NodeJS y Go. Aparte de su excelente rendimiento, FastAPI proporciona un desarrollo ágil, disminuye los fallos humanos en el código, es sencillo de aprender y está totalmente preparado para la producción. FastAPI cumple completamente con estándares de APIs reconocidos, como OpenAPI y JSON Schema [54].

- ***CherryPy***

CherryPy es uno de los frameworks web más antiguos para Python, a pesar de que numerosas personas desconocen su presencia. Uno de los motivos de esto es que CherryPy no es un stack integral que brinde soporte a arquitecturas de múltiples capas. No ofrece beneficios para el frontend ni te instruye sobre cómo interactuar con tu almacenamiento. Por otro lado, CherryPy posibilita que el programador haga esas elecciones. Esta perspectiva contrasta con otros marcos de referencia más reconocidos. CherryPy ofrece una interfaz ordenada y se esfuerza por no perturbar tu labor, al mismo tiempo que ofrece una base sólida desde la que puedes edificar. Los usos

habituales de CherryPy abarcan desde aplicaciones web convencionales con interfaces de usuario hasta aplicaciones web frecuentes con interfaces de usuario [55].

- ***Flask***

Flask es un marco de trabajo de desarrollo web asequible y versátil, creado para simplificar el comienzo rápido y la expansión hacia aplicaciones más sofisticadas. Emplea instrumentos esenciales como Werkzeug (que introduce WSGI, la interfaz estándar de Python entre aplicaciones y servidores), Jinja (un motor de plantillas) y MarkupSafe (para prevenir ataques de inyección). Además, integra Itsdangerous para garantizar la integridad de la información y salvaguardar las cookies de acceso [56].

La Tabla 9 presenta el análisis comparativo de las distintas Plataformas de desarrollo para el control de la información presentadas.

Tabla 9. Análisis comparativo de las distintas Plataformas de desarrollo para el control de la información [54–56].

Criterio Técnico	**FastAPI**	**CherryPy**	**Flask**
Velocidad	Muy rápido debido a su soporte para **async** y optimización automática	Rápido, pero no tiene soporte nativo para **async**	Moderado, no tan rápido como FastAPI
Soporte para ASGI/WSGI	**ASGI** (Asynchronous Server Gateway Interface)	**WSGI** (Web Server Gateway Interface)	WSGI
Facilidad de uso	Fácil de usar, pero más adecuado para desarrollos modernos (APIs)	Interfaz limpia, pero requiere más configuración manual	Muy fácil de usar, especialmente para principiantes
Escalabilidad	Muy alta, especialmente para aplicaciones asíncronas	Escalable, pero no optimizado para aplicaciones muy grandes	Moderada, adecuada para aplicaciones medianas
Tamaño del marco	Relativamente pequeño y eficiente	Framework ligero, pero más básico	Pequeño y ligero, ideal para proyectos pequeños
Soporte de Plantillas	Usa **Jinja2** para plantillas HTML	No tiene soporte nativo para plantillas, depende del desarrollador	Usa **Jinja2** para plantillas HTML
Manejo de Bases de Datos	No incluye un ORM, pero se integra fácilmente con bases de datos y ORMs como SQLAlchemy	No tiene un ORM, se puede integrar con otros	No incluye un ORM, pero se integra bien con SQLAlchemy

1.3.12 Módulos GPS para Sistemas de Navegación y Geolocalización

Los módulos GPS son aparatos electrónicos que facilitan la determinación de la localización geográfica de un objeto, basándose en señales provenientes de satélites. Estos módulos emplean el sistema de posicionamiento global (GPS), que es una red de satélites que orbitan la Tierra y transmiten señales que el módulo recibe para determinar su ubicación en función de latitud, longitud y, en ciertas situaciones, altitud [57].

a. Tipos de módulos GPS para sistemas de navegación y geolocalización.

- ***Módulo GPS u-blox 6***

El NEO-6M es un módulo GPS que utiliza el chip u-blox NEO-6, empleado en una variedad de aplicaciones como sistemas de navegación, seguimiento y supervisión en tiempo real. Este módulo se distingue por su precio reducido, su tamaño reducido y su sencilla integración con plataformas como Arduino, ESP32 y Raspberry Pi. El NEO-6M brinda una exactitud de hasta 2.5 metros en situaciones ideales y posee una frecuencia de actualización de 1 Hz, suministrando información de localización cada segundo. Se adhiere al protocolo NMEA 0183, lo que simplifica su incorporación con otros sistemas que necesitan datos estándar como latitud, longitud, altitud y tiempo. Funciona con una tensión de 3.3V a 5V, y demanda aproximadamente 50 mA durante su operación, lo que lo convierte en idóneo para usos con batería. Posee una antena cerámica incorporada para optimizar la recepción, y en ciertos modelos es posible vincular una antena externa para situaciones de señal más rigurosas. La interacción con otros aparatos se lleva a cabo mediante UART (TX/RX), y ciertos módulos también facilitan el uso de I2C [58]. En la Figura 43 se visualiza un Módulo GPS u-blox 6.



Figura 43. Módulo GPS u-blox 6 [59].

- ***Ublox ZED-F9P (Módulo GPS RTK)***

El Ublox ZED-F9P es un módulo GPS de vanguardia que integra la tecnología RTK (Real-Time Kinematic), que posibilita alcanzar una exactitud de pocos centímetros en las mediciones de ubicación. Esto lo hace una excelente alternativa para aplicaciones que demandan alta precisión, como la topografía, la agricultura de precisión y la navegación autónoma. Este módulo soporta varias constelaciones satelitales, entre las que se incluyen GPS, GLONASS, Galileo, Beidou y QZSS, incrementando notablemente la confiabilidad y exactitud del sistema de posicionamiento, incluso en situaciones difíciles o cuando existen ciertas restricciones al cielo [60].

El ZED-F9P ofrece una frecuencia de actualización de hasta 20 Hz, lo que facilita la adquisición de hasta 20 mediciones de la posición cada segundo, lo que resulta idóneo para aplicaciones de alta velocidad y en tiempo real. Esto lo convierte en perfecto para sistemas autónomos de navegación, drones, vehículos sin tripulación (UGV) y otros aparatos que necesitan actualizaciones rápidas de su ubicación [60].

Este módulo está diseñado para trabajar con interfaces de comunicación como UART, I2C y SPI, lo que simplifica su incorporación a microcontroladores y sistemas embebidos, como Arduino o Raspberry Pi. Adicionalmente, el ZED-F9P puede recibir correcciones RTK de estaciones base o servicios de corrección diferencial para incrementar la exactitud de las mediciones en términos de centímetros. Estas correcciones RTK pueden ser ofrecidas por una estación base próxima o mediante servicios de corrección GNSS de entidades externas [60].

Con un consumo energético reducido, el Ublox ZED-F9P es idóneo para usos móviles y sistemas portátiles, dado que solo requiere aproximadamente 65 mA a 3.3 V en el modo estándar de operación. Este módulo ha sido creado para brindar un rendimiento superior incluso en contextos de interferencia electromagnética y condiciones de señal desfavorables, lo que lo convierte en extremadamente confiable para aplicaciones críticas donde la exactitud en tiempo real es esencial [60]. En la Figura 44 se visualiza un Módulo Ublox ZED-F9P.



Figura 44. Módulo Ublox ZED-F9P [61].

- ***ZED-F9P-01B GPS Module***

El ZED-F9P-01B es un módulo GNSS de u-blox de alta precisión, que ofrece soluciones de posicionamiento con precisión centimétrica a través de la tecnología RTK. Este módulo resulta perfecto para aplicaciones en tiempo real que demandan una actualización y exactitud rápidas, como la navegación autónoma de vehículos, drones y sistemas industriales. Además, respalda varias constelaciones satelitales como GPS, GLONASS, Galileo y BeiDou, lo cual mejora la disponibilidad y exactitud del servicio en zonas con señales restringidas [62].

Este módulo tiene la capacidad de ofrecer un rendimiento superior con una frecuencia de actualización de hasta 20 Hz en modo RTK, lo que lo convierte en idóneo para aplicaciones dinámicas y de alto desempeño. Su habilidad para recibir múltiples bandas le facilita el uso de correcciones de RTK para lograr mediciones más exactas, lo que lo hace una elección ideal para aplicaciones industriales que demandan precisión en centímetros [62].

El ZED-F9P-01B proporciona un alto rendimiento energético y puede ser incorporado con facilidad en una diversidad de sistemas mediante el uso de interfaces estándar como UART, I2C y SPI. Además, se ajusta a protocolos de corrección como RTCM y SPARTN, lo que simplifica su aplicación en sistemas de navegación sofisticados y en aplicaciones de agricultura de precisión, geodesia y seguimiento de vehículos. El ZED-F9P-01B, por su capacidad para corrección diferencial en tiempo real y su habilidad para manejar ambientes complejos, se ajusta a aplicaciones que demandan una exactitud constante y fiabilidad. Además, su habilidad para manejar diversas constelaciones de satélites y su desempeño en situaciones difíciles lo hacen una

alternativa sólida y adaptable para proyectos que demandan una precisión elevada y baja latencia en la ubicación [62].

La Tabla 10 presenta el análisis comparativo de los distintas Tipos de módulos GPS para sistemas de navegación y geolocalización presentadas.

Tabla 10. Análisis comparativo de los distintos tipos de módulos GPS para sistemas de navegación y geolocalización [58], [60], [62].

Parámetro Técnico	**Módulo GPS u-blox 6**	**Ublox ZED-F9P (Módulo GPS RTK)**	**ZED-F9P-01B GPS Module**
Tipo de módulo	GPS	GNSS (GPS, GLONASS, Galileo, Beidou, QZSS)	GNSS (GPS, GLONASS, Galileo, Beidou, QZSS)
Precisión	Hasta 2.5 metros	Centimétrica con RTK	Centimétrica con RTK
Frecuencia de actualización	1 Hz	Hasta 20 Hz (con RTK)	Hasta 20 Hz (con RTK)
Constelaciones soportadas	Solo GPS	GPS, GLONASS, Galileo, Beidou, QZSS	GPS, GLONASS, Galileo, Beidou, QZSS
Interfaces	UART, I2C, SPI	UART, I2C, SPI	UART, I2C, SPI
Consumo energético	Bajo (50 mA)	65 mA típico	65 mA típico
Tiempo de arranque frío	~32 segundos	40 segundos	40 segundos
Soporte para RTK	No	Sí	Sí
Frecuencia de corrección RTK	N/A	1 Hz a 10 Hz	1 Hz a 10 Hz
Precisión de posicionamiento	2.5 metros	Centimétrica (con RTK)	Centimétrica (con RTK)
Aplicaciones típicas	Navegación básica, rastreo en tiempo real	Navegación avanzada, vehículos autónomos, drones, agricultura de precisión	Navegación avanzada, vehículos autónomos, drones, agricultura de precisión
Características adicionales	No soporta corrección diferencial, solo GPS	Soporta corrección RTK, alta sensibilidad, fácil integración	Soporta corrección RTK, alta sensibilidad, fácil integración
Dimensiones	16.0 x 12.2 mm	25 x 23 x 6 mm	25 x 23 x 6 mm
Voltaje de operación	3.3V a 5V	3.3V a 5V	3.3V a 5V
Ideal para	Proyectos de bajo costo y aplicaciones simples	Proyectos que requieren alta precisión en tiempo real	Proyectos que requieren alta precisión en tiempo real

1.4 Objetivos

1.4.1 Objetivo general

Implementar un sistema de monitoreo y detección de comportamiento delincuencial aplicado en medios de transporte utilizando visión artificial.

1.4.2 Objetivos específicos

- Realizar un análisis de los sistemas de seguridad actuales en medios de transporte, identificando sus deficiencias y puntos vulnerables para establecer una base en el desarrollo del sistema de monitoreo y detección de comportamientos delincuenciales.
- Diseñar y desarrollar un sistema de visión artificial que utilice algoritmos avanzados para el monitoreo y detección precisa de eventos delictivos en medios de transporte.
- Implementar un sistema de gestión de datos eficiente que permita respuesta rápida y efectiva en caso de detección de eventos delictivos en medios de transporte.

CAPÍTULO II. METODOLOGÍA

2.1 Materiales

Para la investigación, se emplearon herramientas de hardware y software especializadas en visión artificial para desarrollar un sistema de detección de delitos en el transporte público. Este sistema está diseñado para identificar y alertar sobre actividades sospechosas o delictivas en tiempo real.

Se seleccionó una webcam HD (720p) de Sinovision con micrófono integrado. Esta resolución fue elegida por ofrecer un equilibrio óptimo entre la calidad de imagen necesaria para la detección de posturas y objetos, y la eficiencia computacional. Resoluciones mayores habrían sobrecargado la Nvidia Jetson Nano, aumentando la latencia sin una mejora significativa en la precisión. Así, se garantiza un procesamiento en tiempo real eficiente y un menor consumo de recursos, crucial para un sistema embebido.

El sistema incorporó diversas librerías y modelos. Se utilizó OpenCV para el preprocesamiento del video y MediaPipe Pose Landmark para la estimación de posturas amenazantes. Además, se implementó un modelo YOLO entrenado con un conjunto de datos de objetos sospechosos mediante Transfer Learning, y un modelo SVM para la identificación de posturas.

Para el geoposicionamiento, se integró un módulo GPS Neo 6M que envía coordenadas en tiempo real al detectar situaciones clave. Todas las detecciones y coordenadas de geoposicionamiento se almacenaron en una base de datos SQLite, gestionada mediante el lenguaje de programación Python.

En la etapa de visualización y gestión de la información, se desarrolló una interfaz de usuario intuitiva utilizando el entorno FLASK. Adicionalmente, se creó un canal de difusión en Slack para enviar notificaciones automáticas sobre personas con actitudes sospechosas y delitos flagrantes, garantizando una comunicación rápida y efectiva.

2.2 Métodos

2.2.1 Modalidad de la investigación

Dentro de esta sección se detalla los tipos de investigación que fueron utilizados en el desarrollo de esta investigación, así también como el proceso que implicó este fin.

El proyecto se clasifica como una investigación aplicada, debido a que utilizó herramientas de hardware y software existentes, así como los conocimientos previos sobre visión artificial. Esto que se aplicó el conocimiento disponible para resolver problemas prácticos.

La investigación también fue de tipo bibliográfica, dado que se empleó algoritmos, técnicas y modelos previamente publicados en trabajos de investigación, artículos científicos, y repositorios como GitHub, entre otros. Además, la base teórica de la investigación se sustentó con conocimiento disponible en libros y revistas científicas.

Finalmente, este proyecto también fue una investigación de campo, debido a que se requirió la recopilación de datos sobre los sistemas de seguridad existentes, con el objetivo de establecer puntos de partida y parámetros evaluables en la implementación del sistema propuesto. Este proceso se llevó a cabo mediante entrevistas con el personal de seguridad y la observación directa de los sistemas en funcionamiento.

2.2.2 Población y muestra

Dentro de esta investigación, se incluye la implementación de un prototipo libre, por lo tanto, se puede llevar a cabo en cualquier medio de transporte, y no requiere población y muestra.

2.2.3 Recolección de información

La recolección de datos para este proyecto se llevó a cabo mediante entrevistas al personal de seguridad en medios de transporte público y observaciones directas de los sistemas de monitoreo y vigilancia existentes. Esta información permitió identificar

los componentes clave, como las arquitecturas de los sistemas, el hardware y software requeridos, y los protocolos de seguridad actualmente utilizados. A través de la observación en las entidades de transporte, se definió los parámetros de control que se incorporaron en la solución propuesta. Además, se necesitó construir un dataset representativo para el entrenamiento del sistema de visión artificial. Este dataset se desarrolló a partir de datasets existentes en repositorios especializados, que fueron preprocesados y adaptados para reflejar de manera precisa los comportamientos y escenarios típicos de los entornos de transporte público. La Tabla 11 muestra los criterios de recolección de información.

Tabla 11. Criterios de recolección de información.

Criterio	Descripción
¿Para qué?	Monitorear y detectar comportamientos delincuenciales aplicado en medios de transporte
¿De qué personas u objetos?	Para medios de transporte
¿Como?	A través del uso de Visión Artificial
¿Con que?	Con un sistema de monitoreo y detección de comportamientos delincuenciales
¿Cuándo?	En el periodo de marzo 2025 a julio 2025
¿Dónde?	En medios de transporte

2.2.4 Procesamiento y análisis de datos

La información recopilada dentro de esta investigación fue procesada de la siguiente forma:

- Se llevó a cabo una revisión bibliográfica de proyectos previos relacionados con sistemas de seguridad similares al que se propone.
- Se compararon las metodologías y tecnologías empleadas en estas implementaciones, identificando las más adecuadas y eficientes.
- Se creó un dataset “depurado”, lo cual implica su preprocesamiento antes de ser utilizado en el entrenamiento del modelo.

- Se analizó el rendimiento del modelo, comparando sus resultados con la información obtenida en las entrevistas y observaciones realizadas antes de la implementación.

- Se redactó un informe detallado que describe la metodología empleada durante el desarrollo del proyecto, así como los resultados obtenidos.

CAPÍTULO III. RESULTADOS Y DISCUSIÓN

El estudio se basó en la metodología o estrategia CRISP-DM, la cual fue seleccionada a partir de la Tabla 5 del Capítulo I. La primera etapa, Estado Actual, esquematiza el hardware y software esenciales para un sistema de detección delictiva. A esta le sigue la etapa de Datos e Información Disponible, que describe el procedimiento de recolección de datos, y luego la fase de Conjunto de Datos, donde se detalla la limpieza y el etiquetado de la información.

Posteriormente, en la etapa de Modelado, se presentan los resultados del entrenamiento y el sistema de gestión y visualización de información en tiempo real. La etapa de Evaluación del Sistema especifica las pruebas y métricas de validación realizadas. Finalmente, la fase de Despliegue detalla tanto el esquema funcional como los gastos asociados a la implementación del sistema.

3.1 Estado Actual

De acuerdo con el sondeo efectuado, los sistemas de seguridad más aplicados en los medios de transporte incluyen: Cámaras de vigilancia por video, con una significativa respuesta del 65.0%, lo que indica una elevada prevalencia de estos sistemas. Siguen los botones antipánicos, empleados por un 20.0% de los participantes en la encuesta. En menor proporción, los sistemas de alertas ocupan el 10.0%, seguidos por los sistemas GPS, que constituyen el 5.0% de las respuestas.

Esta preponderancia de las cámaras de videovigilancia subraya la relevancia de la supervisión visual como estrategia primordial en la percepción y administración de la seguridad en el transporte. Sin embargo, una adopción reducida de sistemas como el GPS y las alarmas podría señalar una deficiencia en la capacidad de respuesta rápida o seguimiento post-incidente, a pesar de su capacidad para desalentar o simplificar la intervención de las autoridades. Es vital evaluar si la puesta en marcha actual de estos sistemas basta para satisfacer todas las demandas de seguridad, particularmente en la identificación proactiva de conductas inusuales. La incorporación de tecnologías que potencien la vigilancia por video, como la inteligencia artificial para el estudio de

comportamientos, podría robustecer de manera considerable la infraestructura de seguridad actual. La Figura 45 muestra esta distribución.

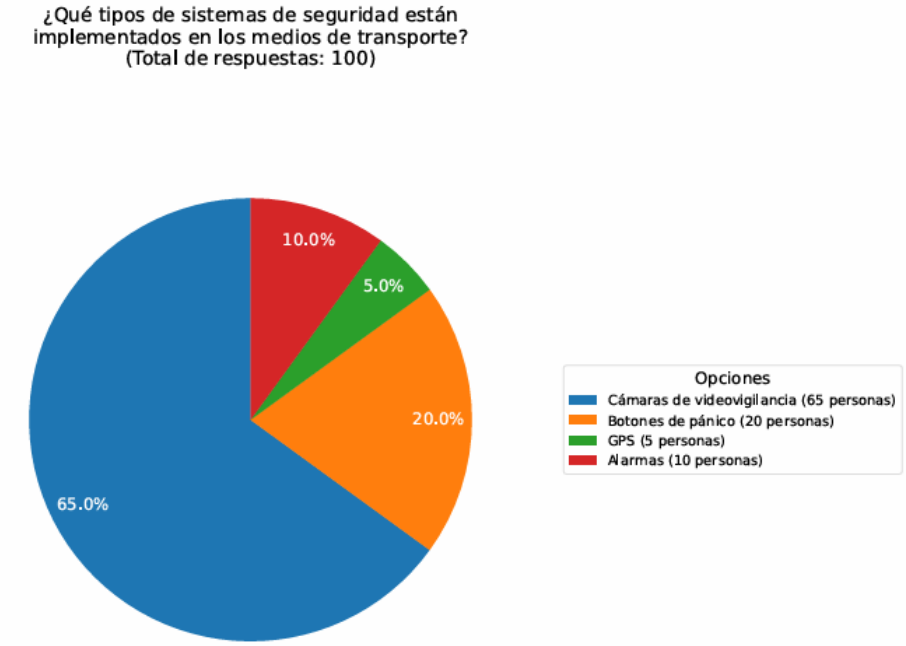


Figura 45. Sistemas de seguridad más utilizados entre los encuestados.

El estudio de la frecuencia de sucesos delictivos en medios de transporte muestra que el 60.0% de los participantes en la encuesta experimenta o sabe de estos sucesos "Frecuentemente". Un 20.0% de los participantes señala que esto sucede "De vez en cuando", mientras que el 20.0% restante percibe estos sucesos como algo "Con frecuencia" vivenciado. La Figura 46 muestra esta distribución de percepciones.

El elevado número de individuos que reportan sucesos delictivos a menudo resalta una inquietud considerable sobre la seguridad de los medios de transporte. Esta percepción común de riesgo sugiere que las medidas de seguridad vigentes podrían no ser adecuadas o no estar funcionando con la eficiencia esperada para desalentar la actividad delictiva. La continuidad de incidentes, incluso esporádicos, afecta de manera directa la confianza y la serenidad de los clientes. Estos descubrimientos subrayan la importancia crucial de instaurar sistemas de vigilancia y detección de conductas delictivas más sólidas y proactivas, como los fundamentados en visión artificial, con el fin de incrementar la seguridad y restablecer la confianza de la población.

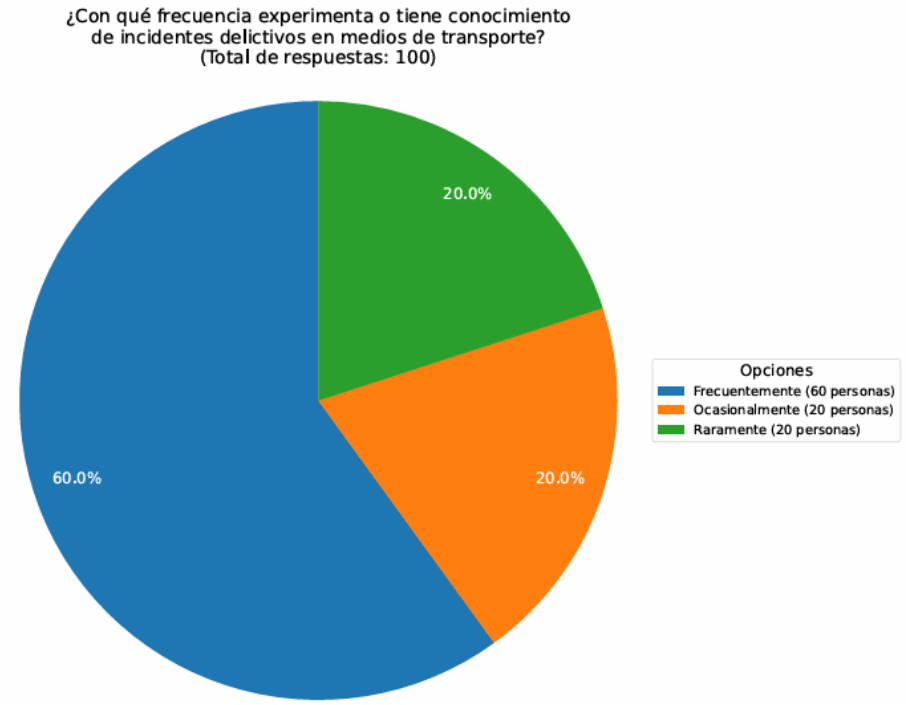


Figura 46. Frecuencia de Incidentes Delictivos en Medios de Transporte.

Respecto a las principales carencias o vulnerabilidades de los sistemas de seguridad vigentes en el sector del transporte, el sondeo muestra que la ausencia de cobertura es considerada el problema más grave, con un 55.0% de las respuestas. El Tiempo de respuesta lento le sigue de cerca, representando el 30.0% de las menciones. Finalmente, un 15.0% de los participantes en la encuesta reconoce la baja resolución de cámaras como una carencia.

La importancia de la "Ausencia de cobertura" indica que, pese a la presencia de sistemas, existen áreas o instantes en los que la supervisión es insuficiente, generando "puntos ciegos" que los criminales podrían aprovechar. El "Tiempo de respuesta lento" es igual de alarmante, puesto que la efectividad de cualquier sistema de seguridad no se basa únicamente en la identificación, sino en la habilidad para responder con rapidez. Una resolución reducida de cámaras, a pesar de ser menos citada, puede poner en riesgo la identificación de individuos o sucesos. La Figura 47 presenta los resultados de los Principales Deficiencias o Puntos Vulnerables de los Sistemas de Seguridad Actuales

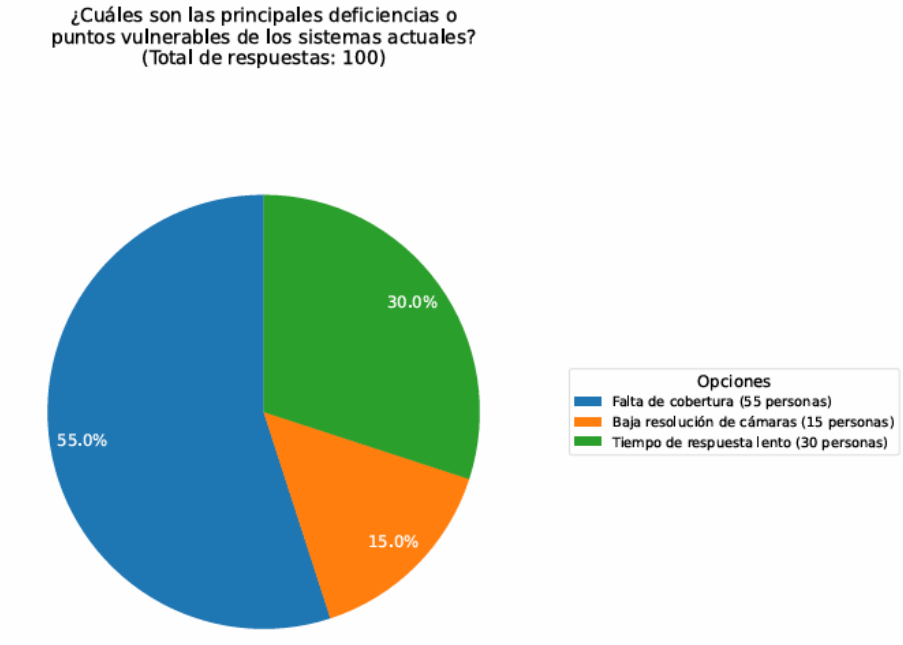


Figura 47. Principales Deficiencias o Puntos Vulnerables de los Sistemas de Seguridad Actuales.

Respecto a la percepción acerca de la eficacia de los sistemas de seguridad actuales para prevenir e identificar actos delictivos en los medios de transporte, el sondeo muestra una diversidad de puntos de vista. Un 40.0% de los participantes en la encuesta califica la eficacia con un 3 en una escala de 1 a 5 (en la que 1 significa "Muy ineficaz" y 5 significa "Muy eficaz"), lo que señala una percepción de eficacia moderada. Un 25.0% la califica con una calificación de 2, lo que indica una efectividad reducida, y un 20.0% la califica como "Muy ineficaz" (puntuación de 1). Solo un 15.0% le concede un puntaje de 4, y ninguno (0%) la califica como "Muy eficaz" (puntuación de 5). Esta distribución de respuestas se muestra en la Figura 48.

Estos hallazgos indican que un porcentaje considerable de los participantes en la encuesta no considera los sistemas de seguridad actuales como sumamente eficaces. El hecho de que las respuestas se concentren en las calificaciones 1, 2 y 3, que representan un 85% del total, señala que hay un considerable margen para mejorar. La falta de votos en la calificación de 5 resalta la carencia de sistemas evaluados como óptimas operaciones. Esta sensación de ineficacia es un elemento crucial que respalda la importancia de explorar e instaurar soluciones innovadoras, como un sistema de vigilancia basado en visión artificial, capaz de tratar las carencias presentes y potenciar la confianza y protección de los usuarios.

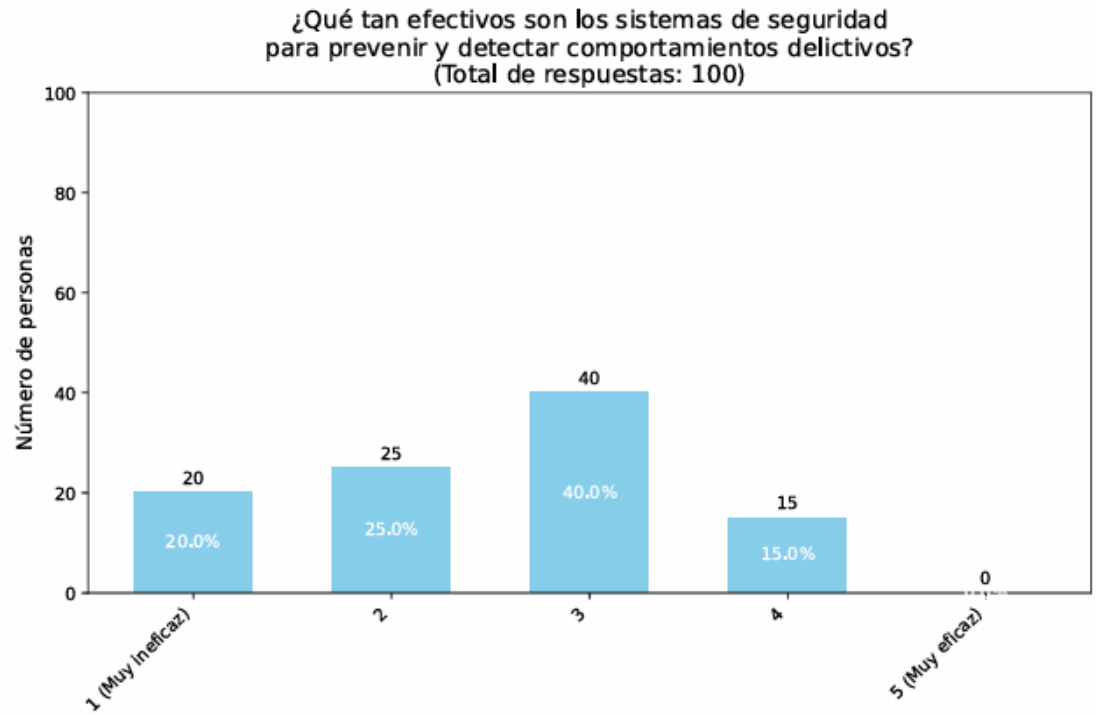


Figura 48. Componentes del Sistema y Análisis de Efectividad.

Respecto a la voluntad de los participantes en la encuesta de instaurar un sistema de seguimiento basado en visión artificial en los medios de transporte, los hallazgos son reveladores: un enorme 88.0% de los individuos encuestados expresa "Sí" dispuestos a hacerlo, mientras que un 12.0% se inclina por "No".

Este elevado índice de aceptación señala una intensa demanda y una considerable disposición del público hacia la adopción de tecnologías de vanguardia para potenciar la seguridad. La actitud positiva indica que los usuarios ven la visión artificial como una posible respuesta para tratar las carencias e inquietudes de seguridad presentes. Este apoyo mayoritario confirma la relevancia y el posible efecto beneficioso del desarrollo de un sistema de vigilancia e identificación de conductas delictivas mediante visión artificial, ofreciendo una base firme para su fundamentación e implementación en el marco de este proyecto de investigación. La Figura 49 muestra los resultados sobre la Propuesta y Aceptación de un Sistema de Monitoreo Basado en Visión Artificial.

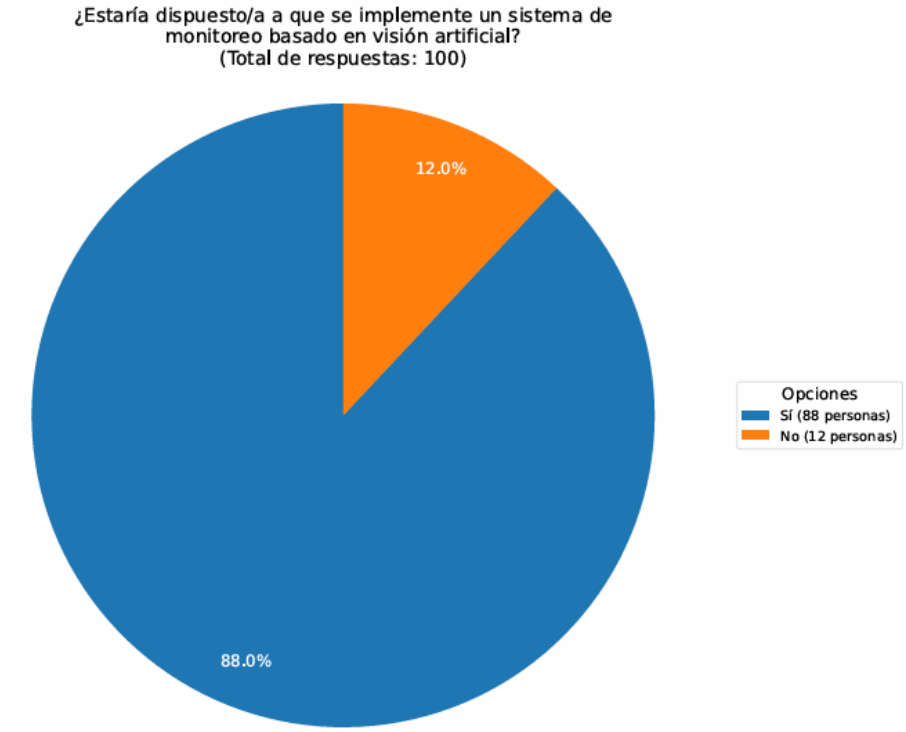


Figura 49. Propuesta y Aceptación de un Sistema de Monitoreo Basado en Visión Artificial.

3.1.1 Estructura general

La estructura general ofrece una perspectiva global del flujo de información que presenta el sistema de detección delictiva en medios de transporte, se compone de distintas etapas, las mismas son fundamentales e imperdibles dentro de un sistema de visión artificial y fueron expuestas dentro del capítulo I. La Figura 50 indica la estructura general y las etapas más importantes que fueron utilizadas dentro del sistema implementado en este estudio.

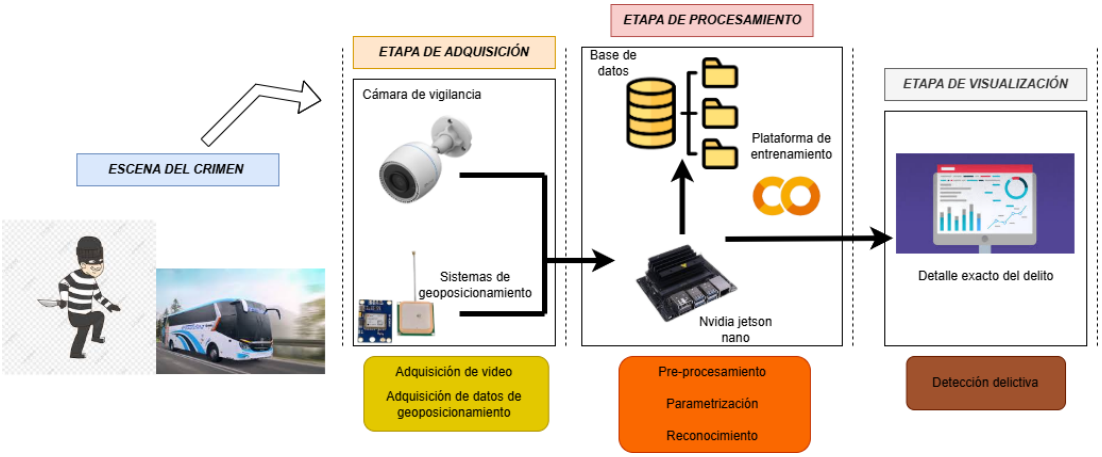


Figura 50. Estructura general del sistema propuesto.

a. Etapa de Adquisición

- ***Adquisición de video***

Esta fase se ocupa de capturar imágenes en tiempo real mediante las cámaras de vigilancia instaladas en el ambiente de estudio. Esta información es transformada en señales eléctricas que son transmitidas a la siguiente fase, donde se encuentra el dispositivo de procesamiento.

- ***Adquisición de datos de geoposicionamiento***

La obtención de datos de geoposicionamiento implica la recolección de datos exactos acerca de la localización geográfica de un objeto, individuo o vehículo en un sistema de coordenadas terrestres (como latitud y longitud). Esta información es esencial para diversas aplicaciones en el sector de las telecomunicaciones, navegación, supervisión, logística, etc.

b. Etapa de Procesamiento

- ***Preprocesamiento***

Recopila totalmente las señales eléctricas que se originan en la fase de captura de video y las convierte en fotogramas digitales. Luego, realiza un filtrado para disminuir el

ruido en las imágenes y genera secuencias estables de fotogramas para su envío a la siguiente fase.

- ***Parametrización***

Dentro de esta etapa, los fotogramas que se han procesado en la fase previa son parametrizados para ajustarlos a las dimensiones que admite la capa de entrada del modelo durante la fase de reconocimiento.

- ***Reconocimiento***

La detección de los eventos delictivos se los realiza en base o en función a los datos que se parametrizan dentro de la etapa anterior, aquí se hace un manejo de enfoque multimodal; debido a que se cuenta con el procesamiento de la detección de objetos sospechosos que realiza la identificación de objetos sospechosos para el cometimiento de los actos ilícitos y el procesamiento de posturas amenazantes que proporciona una predicción de amenaza por medio de un modelo de detección de posturas corporales.

c. Etapa de Visualización

- ***Detección delictiva***

Esta fase tiene como objetivo mostrar dentro del gestor de información; el video en tiempo real, así como los datos geográficos (como latitud y longitud), que son recolectados por la fase de Adquisición de Datos de Video y geoposicionamiento.

3.1.2 Selección del hardware

Para la selección del hardware, se consideró cuidadosamente los requerimientos del sistema. Esta decisión fue fundamental, ya que determina directamente las capacidades técnicas del sistema de inspección automatizada. El proceso de selección se basó en una comparativa de dispositivos disponibles en el mercado, con el objetivo de elegir aquellos que mejor se adaptaran a cada una de las etapas principales del sistema.

La implementación de un sistema de visión artificial para la detección de asaltos dentro de vehículos requiere una selección técnica adecuada del sensor de imagen. Esta

elección debe basarse en parámetros físicos y operativos que aseguren la calidad de la captura bajo condiciones variables de luz, espacio limitado y necesidad de respuesta en tiempo real. A continuación, se detallan y justifican los principales parámetros a considerar.

a. Parámetros Críticos para la Selección de la Cámara

- ***Resolución del Sensor***

La resolución determina el número total de píxeles que conforman la imagen. Para identificar rostros y objetos con claridad dentro de un vehículo, se requiere una resolución mínima de 1280x720 píxeles. Sin embargo, una resolución Full HD (1920x1080) proporciona mayor detalle, lo cual es deseable para aplicaciones de seguridad.

$$Tamaño\ objeto\ detectable \approx Tamaño\ del\ FOV\ /\ N°\ píxeles\ dimensión \quad (7)$$

- ***Campo de Visión (FOV)***

El FOV debe ser suficiente para cubrir todo el interior del vehículo desde una ubicación óptima. Este campo depende de la distancia de trabajo, el tamaño del sensor y la distancia focal de la lente. Un FOV mayor permite cubrir más espacio, pero reduce la resolución efectiva sobre objetos pequeños.

$$FOV = (Tamaño\ del\ sensor \times Distancia\ de\ trabajo)\ /\ Distancia\ focal \quad (8)$$

Utilizan los siguientes parámetros típicos de una cámara compacta de visión artificial adecuada para espacios pequeños como el interior de un vehículo:

• Ancho del sensor: 4.8 mm

• Distancia focal: 2.8 mm

• Distancia de trabajo: 500 mm

Aplicando la fórmula del FOV:

$$FOV = (Tamaño\ del\ sensor \times Distancia\ de\ trabajo) / Distancia\ focal.$$

$$FOV = (4.8\ mm \times 500\ mm) / 2.8\ mm = 857.1\ mm.$$

Esto significa que, a 500 mm de distancia, la cámara puede observar un área de aproximadamente 857 mm de ancho, como se muestra en la Figura 51.

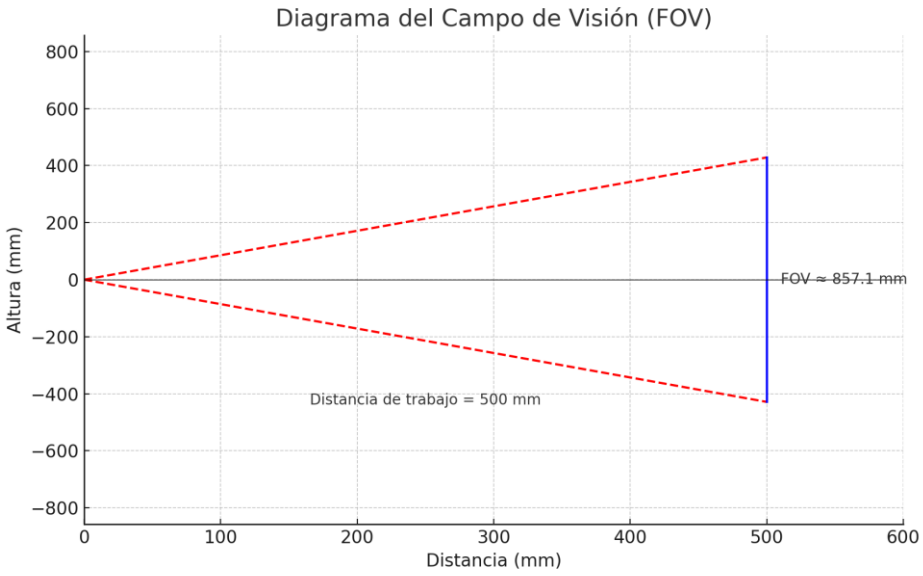


Figura 51. Diagrama de campo de visión focal

- ***Tamaño del Sensor y del Píxel***

Sensores con mayor superficie y píxeles grandes captan más luz, lo que mejora el rendimiento en condiciones de baja iluminación y reduce el ruido digital. Esto es crucial en escenarios con iluminación limitada, como el interior de vehículos en horas nocturnas.

$$Cantidad\ de\ luz\ captada \propto Tamaño\ del\ píxel^2 \quad (9)$$

- ***Framerate (FPS)***

El número de imágenes por segundo (frames per second, FPS) debe permitir registrar con fluidez el movimiento humano. Un mínimo de 30 fps asegura que no se pierdan eventos críticos por falta de muestreo temporal.

$$FPS = 1 / Tiempo\ entre\ cuadros \tag{10}$$

- ***Obturador: Global vs. Rolling***

El obturador determina cómo el sensor expone los píxeles a la luz. El obturador rolling es más económico y suficiente si el vehículo está en reposo. Sin embargo, si hay movimiento importante o vibraciones, es preferible un global shutter.

$$Rolling: fila\ por\ fila\ /\ Global: exposición\ simultánea \tag{11}$$

- ***Visión Nocturna e Iluminación***

Para garantizar operación continua en condiciones de baja luz, es indispensable contar con sensores sensibles al infrarrojo o con soporte para iluminación IR (invisible al ojo humano). Las cámaras deben tener bajo umbral de lux (<0.1 lux).

- ***Lente y Distancia Focal***

La distancia focal afecta directamente el FOV. En vehículos, se recomiendan lentes de corta distancia focal (2.8 mm – 4 mm) para obtener un ángulo amplio. Esto permite observar a todos los ocupantes con una sola cámara.

$$FOV \approx 2 \times arctan\left((tamaño\ del\ sensor) / (2 \times distancia\ focal)\right) \tag{12}$$

b. Selección de Cámaras Comerciales

Con base en los requerimientos anteriormente detallados, se presentan ejemplos de cámaras que cumplen con los parámetros expuestos, siendo aptas para integración en sistemas de detección de eventos en tiempo real en vehículos. La Tabla 12 expresa una comparativa entre distintas Cámaras Comerciales:

• ***Basler daA1920-30uc:*** Cámara industrial USB 3.0, 1920x1080, 30 fps, óptima para IA embebida.

• ***Vivotek MD9584-H:*** Cámara PoE, visión IR, carcasa antivandálica, ideal para transporte público.

• ***Arducam USB 3.0 Global Shutter:*** Compacta, Full HD, para sistemas embebidos con IA en borde.

Tabla 12. Comparativa entre distintas Cámaras Comerciales

Tipo de proyecto	**Cámara recomendada**
Integración con IA embebida (Jetson)	Basler o Arducam USB 3.0 Global Shutter
Seguridad en transporte público/taxi	Vivotek MD9584-H o similar (PoE, IR)
Sistema personalizado con software propio	Basler ace 2 o Dahua Starlight OEM

La selección adecuada de una cámara para visión artificial en interiores de vehículos requiere equilibrio entre calidad de imagen, sensibilidad, resolución y ángulo de visión. El dimensionamiento preciso, garantiza el desempeño óptimo del sistema bajo diferentes condiciones operativas.

c. Etapa de captura de imágenes

Para el sistema de inspección, se consideró una cámara Sinovision HD 720p como la opción principal debido a sus capacidades. Además, se analizaron otros dos modelos de cámaras USB de 720p que presentaban características técnicas inferiores.

La primera cámara evaluada fue la Microsoft LifeCam HD-3000 y es presentada en la Figura 52. Este modelo, aunque de una marca reconocida, opera a 720p y se caracteriza por utilizar un lente de plástico y un enfoque fijo optimizado para distancias de videoconferencia, lo que limita su capacidad para la inspección de objetos cercanos. Su velocidad de captura es de 30 fotogramas por segundo (FPS), pero la calidad de

imagen en condiciones de iluminación no ideales tiende a ser inferior, mostrando más ruido digital. Su campo de visión (FOV) es moderado, pero la corrección automática de luz es básica y puede no ser suficiente para entornos de inspección variables.



Figura 52. Cámara Microsoft LifeCam HD-3000

Un segundo modelo considerado fue la Logitech C270, una webcam de 720p de marca reconocida y amplia disponibilidad. Si bien es fiable y opera a 30 FPS, utiliza una lente de plástico, lo que puede afectar la nitidez en comparación con lentes de vidrio o híbridas. En la Figura 53 se muestra esta cámara. Su enfoque es fijo, optimizado para distancias de videoconferencia, lo que limita su versatilidad para diferentes distancias de objeto. La sensibilidad lumínica es aceptable, aunque no sobresaliente en entornos con poca luz. Uno de sus principales puntos de debilidad para aplicaciones de inspección es su campo de visión estrecho, que limita el área de cobertura efectiva.



Figura 53. Cámara Webcam Logitech C270.

La Tabla 13 muestra las especificaciones técnicas de los tres tipos de cámaras analizadas.

Tabla 13. Especificaciones técnicas de los tres tipos de cámaras analizadas.

Característica Técnica	**Sinovision HD 720p (Referencia)**	**Microsoft LifeCam HD-3000 (720p)**	**Logitech C270 (720p)**
Resolución Máxima	1280x720p (HD)	1280x720p (HD)	1280x720p (HD
Velocidad de Captura	30 FPS a 720p	30 FPS a 720p (en condiciones ideales)	30 FPS a 720p
Tipo de Lente	Probablemente vidrio o híbrida	Plástico	Plástico
Enfoque	Foco fijo o Autoenfoque básico	Foco fijo	Foco fijo
Sensibilidad Lumínica	Buena para su rango (ej. 0.5 - 1 Lux)	Aceptable, pero con más ruido en poca luz	Aceptable
Campo de Visión (FOV)	Moderado (ej. 60°-70°)	Moderado (68°)	Estrecho (60°)
Corrección de Luz	Automática básica	Tecnología TrueColor (básica)	RightLight™ 2 (básica)
Interfaz	USB 3.0	USB 2.0	USB 2.0
Micrófono	Integrado, calidad decente	Integrado, con cancelación de ruido	Mono con reducción de ruido
Compatibilidad OS	Compatible con UVC	Compatible con UVC	Compatible con UVC
Construcción	Plástico de calidad media, clip universal	Plástico duradero, base flexible	Plástico duradero, clip universal

d. Equipos microcontroladores.

Cuando se selecciona un microcontrolador para un proyecto específico, la decisión se basa en equilibrar las necesidades del sistema con las capacidades del hardware disponible en el mercado. En el caso del sistema de inspección, se consideró tanto el Arduino Uno como el ESP32, dos plataformas ampliamente utilizadas, pero con enfoques y prestaciones distintas.

El Arduino Uno es una plataforma de desarrollo basada en el microcontrolador ATmega328P. Se le reconoce por su simplicidad, robustez y una extensa comunidad de soporte, lo que facilita su uso para principiantes y proyectos con requisitos de hardware sencillos. Su arquitectura se enfoca en tareas de control y adquisición de datos que no demandan alta velocidad de procesamiento ni conectividad avanzada. Por ejemplo, para la adquisición de datos de geoposicionamiento a través de un módulo como el NEO-6M, el Arduino Uno ofrece suficientes puertos serie (UART) para comunicarse con el GPS y la capacidad de procesamiento necesaria para leer y retransmitir los datos NMEA. Su consumo de energía es moderado y su tamaño es adecuado para integraciones donde el espacio no es extremadamente limitado.

Por otro lado, el ESP32 es un microcontrolador de doble núcleo mucho más potente, que integra conectividad Wi-Fi y Bluetooth de forma nativa. Su arquitectura está diseñada para aplicaciones que requieren capacidades avanzadas de procesamiento, comunicación inalámbrica (IoT) y un menor consumo de energía en modos de bajo consumo. Dispone de más pines GPIO, múltiples interfaces serie (UART, SPI, I2C) y una mayor velocidad de reloj. A pesar de estas ventajas, su mayor complejidad puede implicar una curva de aprendizaje ligeramente más pronunciada y un consumo energético mayor cuando sus capacidades inalámbricas están activas.

La elección del Arduino Uno en el presente proyecto se fundamentó en su idoneidad específica para la tarea de adquisición de datos GPS del módulo NEO-6M. Dado que esta etapa solo requería una comunicación serie estable y el procesamiento básico de datos de coordenadas, las capacidades del Arduino Uno eran completamente suficientes. La Tabla 14 muestra las especificaciones técnicas de los dos microcontroladores analizados.

Tabla 14. Especificaciones técnicas de los dos microcontroladores analizados.

Característica Técnica	**Arduino Uno**	**ESP32**
Microcontrolador	ATmega328P	ESP32 (generalmente ESP32-WROOM-32, doble núcleo)
Velocidad de Reloj	16 MHz	160/240 MHz
Memoria Flash	32 KB (para código)	4 MB (o más, para código y datos)
SRAM	2 KB	520 KB
Pines GPIO	14 (6 con PWM)	~30-36 (depende del módulo, muchos con PWM y otras funciones)
Interfaces Serie (UART)	1	3
Conectividad Inalámbrica	N/A	Wi-Fi (802.11 b/g/n), Bluetooth (v4.2 BR/EDR y BLE)
ADC (Resolución)	6 canales (10 bits)	18 canales (12 bits)
DAC (Resolución)	N/A	2 canales (8 bits)
Consumo de Energía	Moderado (depende del uso, sin modos de bajo consumo avanzados)	Bajo (con modos de sueño profundo y ligero) a Moderado (Wi-Fi/BT activos)
Complejidad de Uso	Muy sencillo, ideal para principiantes	Moderada, más opciones y curva de aprendizaje más pronunciada
Coste Típico	Bajo	Bajo a Moderado (ligeramente superior al Uno por sus funciones)

e. Unidades de procesamiento

Cuando se trata de elegir una plataforma para sistemas de inspección automatizada que requieren procesamiento de imágenes avanzado, visión por computadora o inferencia de inteligencia artificial en el borde, la decisión entre una Jetson Orin Nano y una Raspberry Pi es crucial. Ambas son computadoras de placa única (SBC), pero están diseñadas para propósitos y niveles de rendimiento muy diferentes.

La Raspberry Pi es reconocida por ser una plataforma versátil, de bajo costo y fácil de usar, ideal para una amplia gama de proyectos de electrónica, IoT, y servidores ligeros. Sus modelos más recientes, como la Raspberry Pi 4, ofrecen capacidades de procesamiento decentes con una CPU ARM, memoria RAM considerable y puertos GPIO. Es excelente para tareas generales de computación, automatización del hogar, pequeños servidores web o como centro de medios. Sin embargo, su capacidad para ejecutar cargas de trabajo intensivas de visión por computadora o inferencia de redes neuronales es limitada, ya que carece de una unidad de procesamiento gráfico (GPU) dedicada y optimizada para estas tareas.

Por otro lado, la Jetson Orin Nano de NVIDIA es una plataforma diseñada específicamente para la inteligencia artificial y la robótica en el borde. Su característica distintiva es la inclusión de una GPU NVIDIA basada en la arquitectura Ampere con núcleos Tensor. Esto le otorga una capacidad de cómputo paralelo masiva, indispensable para acelerar aplicaciones de visión por computadora, procesamiento de imágenes en tiempo real y la ejecución eficiente de modelos de aprendizaje profundo. Aunque su costo es significativamente mayor que el de una Raspberry Pi, y su consumo de energía puede ser más alto bajo carga pesada, su rendimiento en tareas de IA es incomparable dentro de su segmento, lo que la hace ideal para sistemas de inspección que demandan alta precisión y velocidad en el análisis visual. La Figura 54 exhibe la Unidad de procesamiento NVIDIA Jetson Nano utilizada.



Figura 54. Unidad de Procesamiento NVIDIA Jetson Orin Nano.

La elección de la Jetson Orin Nano para el presente proyecto se justifica plenamente por sus capacidades superiores en procesamiento de IA y visión por computadora. Para un sistema de inspección automatizada, la habilidad de procesar grandes volúmenes de datos de imagen rápidamente, ejecutar algoritmos complejos de detección de defectos o clasificación de objetos en tiempo real, y realizar inferencia de modelos de aprendizaje profundo con baja latencia, son factores críticos. Estas tareas son inherentemente intensivas en GPU, y la arquitectura de la Jetson Orin Nano está optimizada precisamente para ellas, ofreciendo un rendimiento que una Raspberry Pi simplemente no puede igualar en este ámbito especializado. La Tabla 15 muestra las especificaciones técnicas de las dos Unidades de procesamiento analizadas.

Tabla 15. Especificaciones técnicas de las Unidades de procesamiento analizadas

Característica Técnica	Jetson Orin Nano (8GB)	Raspberry Pi 4 (8GB)
CPU	6-core ARM Cortex-A78AE v8.2 64-bit CPU (hasta 1.5 GHz)	Broadcom BCM2711, Quad-core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5 GHz
GPU	NVIDIA Ampere architecture with 1024 CUDA cores and 32 Tensor Cores	VideoCore VI (OpenGL ES 3.1, Vulkan 1.0)
Memoria RAM	8 GB LPDDR5	8 GB LPDDR4
Capacidad de IA	Hasta 40 TOPS (Tera Operaciones Por Segundo)	Limitada (depende de software y optimizaciones CPU/GPU básica)
Salida de Video	DisplayPort 1.4a, HDMI 2.1	2 x micro-HDMI (soporta 4Kp60)
Conectividad	Gigabit Ethernet, M.2 Key M (para NVMe), M.2 Key E (para WiFi/BT)	Gigabit Ethernet, Wi-Fi (802.11ac), Bluetooth 5.0
Puertos USB	3x USB 3.2 Gen2, 1x USB 2.0 (Type A), USB Type C para modo recuperación	2x USB 3.0, 2x USB 2.0
Almacenamiento	eMMC interno opcional, ranura microSD, soporte NVMe	ranura microSD, soporte para arranque desde USB
Sistema Operativo	JetPack SDK (Ubuntu, CUDA, cuDNN, TensorRT, OpenCV, etc.)	Raspberry Pi OS (basado en Debian), otros Linux ARM
Consumo de Energía	7W-15W (modo de potencia configurable)	3W-8W (típico bajo carga)
Precio Aproximado	~$499 USD (para el módulo 8GB)	~$75 USD (para el modelo 8GB)
Enfoque Principal	Computación de IA en el borde, robótica, visión por computadora	Proyectos de propósito general, IoT, educación, servidores ligeros

3.1.3 Especificaciones de la propuesta

En esta sección se detallan cada uno de los requisitos del sistema de detección de delitos, comenzando por lo más elemental que son los recursos y la información esencial, a través de una descripción breve del proceso de recolección y generación. Aquí también se realiza la interacción entre la capacidad del sistema y la tecnología que facilita la obtención del resultado definitivo.

Además, aborda la conexión entre la habilidad del sistema para funcionar y las tecnologías involucradas, enfatizando cómo estas facilitan la recopilación, gestión y valoración de los datos para obtener resultados precisos y oportunos. Se consideran tanto los aspectos técnicos como las limitaciones inherentes a los recursos disponibles,

con la finalidad de asegurar que el sistema logre los objetivos establecidos en términos de una identificación eficaz y una respuesta inmediata ante escenarios delictivos.

Finalmente, este segmento también ofrece una visión integral de las demandas funcionales y no funcionales, las cuales orientan el diseño, la puesta en marcha y la verificación del sistema, asegurando que se mantenga la calidad, la seguridad y la escalabilidad necesarias para su operación en situaciones reales.

a. Recursos e Información

Se realizó la clasificación de un conjunto de datos de carácter esencial determinadas en tres tipos de etiquetas dentro de Roboflow; las mismas que son: “gorras”, “armas” y “cuchillo”, todo esto para llevar a cabo el entrenamiento del modelo capaz de detectar objetos amenazantes que pueden ser utilizados para el cometimiento de actos ilícitos, así como también de posturas corporales sospechosas por medio de la recolección de videos y la extracción de determinadas características con Mediapipe. Estos datos fueron adquiridos en bruto en medios de acceso público, como son el caso de noticias, reels de redes sociales, foros informativos. Todo esto con la finalidad de compactar en conjunto de datos y contar con estadísticas confiables o estables.

b. Capacidad del sistema para funcionar

El sistema de detección delincuencial debe contar con total capacidad de detectar personas con actitud sospechosa en relación de los objetos amenazantes que lleven consigo, así como también posturas sospechosas que ellos muestren. Por otra parte, el sistema debe enviar una alerta al sistema de gestión de información, la presencia de estos objetos amenazantes como son gorras, armas, cuchillos y posturas corporales sospechosas. Junto con los datos exactos de geoposicionamiento del delito. Por último, estos datos de detección y los de geoposicionamiento deben almacenarse en una base de datos desarrollada en SQLite para su análisis posterior.

c. Tecnología involucrada

El sistema de detección delincuencial aplicada en medios de transporte que utiliza visión artificial que se propuso dentro de este proyecto de investigación, posee la

necesidad de contar con tecnología específica; separada o clasificada en hardware y en software como la unidad de procesamiento de imágenes, el módulo GPS para la adquisición de los datos de geoposicionamiento (latitud y longitud), cámaras webcam, plataformas de entrenamiento en la nube, protocolos de comunicación (serial y mqtt), librerías pertenecientes al lenguaje de programación de Python que se encuentran especializadas para aplicaciones de Visión Artificial y el sistema de gestión de información para la etapa de visualización.

3.2 Construcción del Banco de datos

Se generaron dos conjuntos de datos como parte del proceso de construcción del banco de datos, el primer banco de datos fue dedicado para el entrenamiento del modelo capaz de detectar objetos amenazantes para el cometimiento de actos ilícitos, por medio de la colocación de etiquetas dentro de Roboflow, y el segundo banco de datos se compone de coordenadas normalizadas de posturas corporales sospechosas. Dentro de la recolección de datos de geoposicionamiento (latitud y longitud) se realizó una programación dentro del IDE de Arduino; específicamente para una placa de Arduino UNO, misma que recibe los datos del módulo GPS NEO 6M y mediante una comunicación serial los envía al sistema de gestión de información desarrollado con flask por medio de NODE-RED al convertir el protocolo de comunicación serial a mqtt, a continuación de presenta a detalle todo este procedimiento seguido.

3.2.1 Detección de objetos amenazantes

a. Depuración y clasificación de datos.

Para empezar a construir el banco de datos o dataset, para posteriormente contar con un adecuado entrenamiento y validación del modelo fue fundamental realizar una limpieza de los datos, es decir; se recolectaron una amplia cantidad de videos en donde se evidencia el cometimiento de eventos delincuenciales dentro de unidades de transporte público y luego se procedió a dividir cada uno de los videos en imágenes; con la ayuda de un código de Python. Se cargaron estas imágenes en Roboflow para el etiquetado mediante la clasificación entre las imágenes que sirven para el dataset y

las que no. Eliminándolas, ya sea porque no se evidencia ningún elemento que represente una amenaza o porque la imagen está difusa.

b. Identificación de las clases de objetos a detectar y etiquetado.

Para esta fase que se refiere específicamente del etiquetado de las imágenes ya seleccionadas, se ocupó Roboflow. Como primera instancia se subieron a esta plataforma todas las imágenes que aprobaron la etapa de filtraje, siendo éstas un total de 6885 imágenes. Después se procede a etiquetar las imágenes; dividiéndolas en tres tipos de etiquetas: “gorras”, “armas” y “cuchillo”. Para las armas y los cuchillos se realizó la etiquetación de estos objetos junto con la inclusión del contexto; específicamente la mano que tiene en su poder el objeto. Todo esto con la finalidad de robustecer la eficiencia del modelo. Para el caso de las gorras el etiquetado fue de carácter tradicional, es decir; solamente el objeto. La Figura 55 y la Figura 56 evidencian lo mencionado.

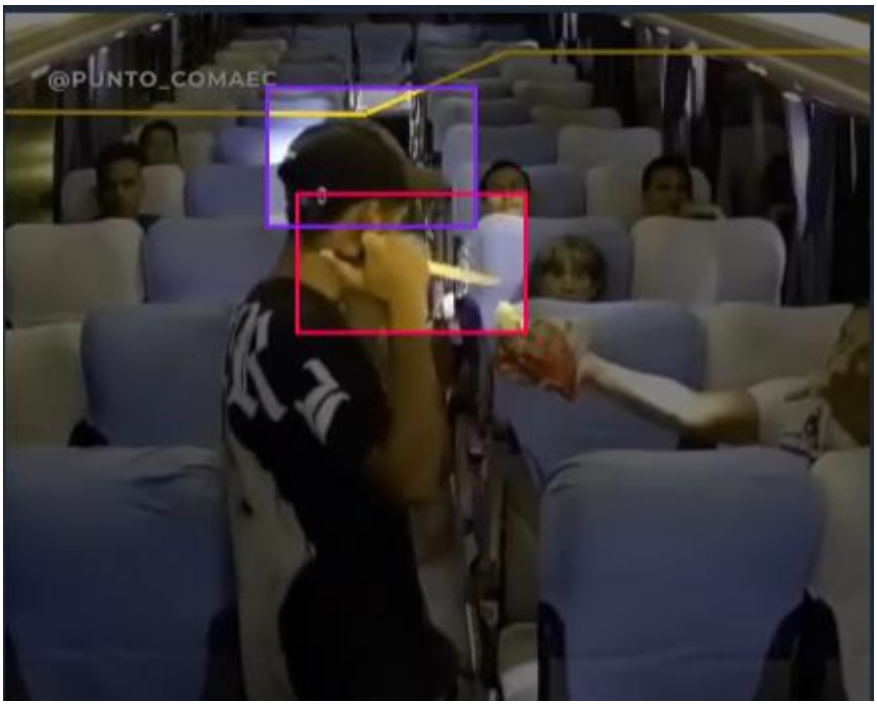


Figura 55. Etiquetado de la clase "gorras" y "cuchillo".



Figura 56. Etiquetado de la clase "armas".

Una vez que se cumplió con la etapa del etiquetado se obtuvo un dataset compuesto de 6885 imágenes; como se ha mencionado en los párrafos anteriores. Por lo tanto, el dataset puede ser entrenado dentro del entorno cloud seleccionado, que en este caso es Google Colab. Sin antes, hay que mencionar que Roboflow posee la capacidad de producir diversas versiones o formatos de descargas del dataset, tales como: Yolo, COCO, Tensor Flow, Pascal VOC, entre otros.

c. Estructura del formato de descarga del dataset.

Después de cumplir con el proceso de filtrado o clasificación de las imágenes y el etiquetado de aquellas. La plataforma de Roboflow permite la construcción del dataset y adicional a ello se puede aplicar varios parámetros técnicos como son: preprocesamiento de la información, redimensionamiento de imágenes, aplicación de filtros, etc. En este caso se hizo la aplicación de redimensionamiento de las imágenes a 640 x 640 pixeles puesto que es necesario para trabajar con el modelo de Yolo, para equilibrar las etiquetas que se encuentren desbalanceadas se aumentó datos con rotaciones a 10 grados, se aplicó de ruidos y filtros; blancos y negros.

Al momento de realizar la descarga del dataset se hace una distribución de éste. En este caso la división fue la siguiente: El dataset cuenta con 6885 imágenes las cuales el 73 % es dedicado para entrenamiento lo que representa 5056 imágenes, el 19 % es para validación con 1315 imágenes y el 1 % es para testeo, lo cual representa 514 imágenes del dataset. Esta división se evidencia en la Figura 57, misma que es generada dentro del entorno de Roboflow antes de descargar el dataset.

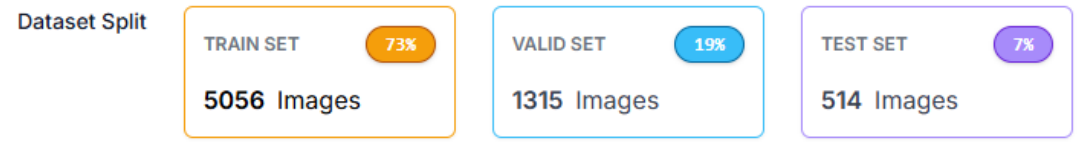


Figura 57. Distribución del dataset para su descarga desde la plataforma Roboflow.

Continuando con el tema de la descarga o exportación del dataset desde la plataforma de Roboflow hacia el ordenador de trabajo, se exportó el formato adecuado del banco de datos. Este formato es el indicado para modelos Yolo; tanto para la versión 8 y 11 que son aquellas con las que se trabajó, la Figura 58 presenta la estructura de este formato, donde contiene las carpetas o directorios "train", "valid" y "test", y en su interior contiene las imágenes en formato .jpg con sus respectivas anotaciones en formato .txt, y finalmente la estructura de este formato posee un archivo denominado como "data.yaml", el mismo posee información relacionada a las rutas de las imágenes y las anotaciones.

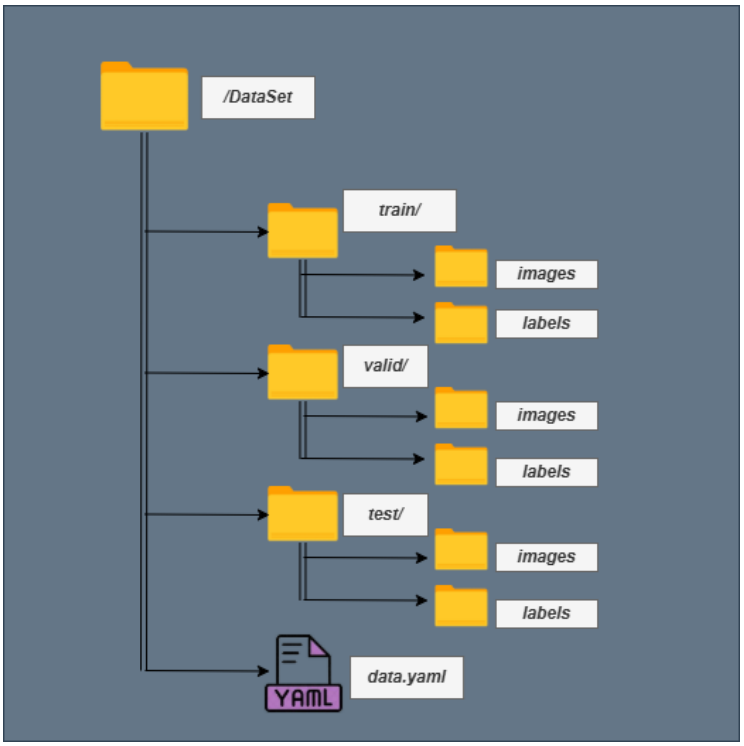


Figura 58. Estructura del dataset en formato YOLO.

3.2.2 Identificación de posturas corporales sospechosas

Para el entrenamiento de este modelo se utilizaron un total de 5 videos, con una duración de 12 minutos cada uno de ellos, en donde se recreó tres tipos o clases de escenas diferentes (comportamiento normal, comportamiento anómalo con arma de fuego y comportamiento anómalo con arma blanca o cuchillo). Con la ayuda de un script de Python se logró dividir cada uno de los videos en fotogramas distintos; específicamente en 2 fotogramas por segundo, con el fin de fortalecer el dataset y hacer un etiquetado del fotograma, en base de la escena que se evidenciaba. La Figura 59 exhibe un fotograma que recrea un comportamiento anormal con arma de fuego de uno de los videos recolectados.



Figura 59. Fotograma que recrea un comportamiento anómalo con arma de fuego para el etiquetado o clasificación de posturas corporales.

La librería principal de Python para la extracción de las posturas corporales de cada uno de los fotogramas fue Mediapipe; se la utilizó por su sencillez de adaptabilidad. Para el entrenamiento del modelo se tomó en cuenta desde la cintura hacia arriba a cada una de las personas que recrearon cada uno de los comportamientos dentro de los fotogramas. A partir de ahí la postura objetivo a identificar fue un conjunto de puntos, mismos que son ya predeterminados sus valores numéricos y la parte del cuerpo humano que hace referencia cada uno de ellos, teniendo así; el número 0. “la nariz”, 11 “el hombro izquierdo”, 12. “hombro derecho”, 13. “codo izquierdo”, 14. “Codo derecho”, 15. “muñeca izquierda”, 16. “muñeca derecha”, 23. “cadera izquierda”, 24. “cadera derecha”.

En la construcción del dataframe, se estableció coordenadas normalizadas con respecto a una región de interés definida o ROI, donde se compuso de todos los puntos clave detallados anteriormente. Cada una de estas coordenadas se guardaron dentro de un archivo .csv el cual tuvo lugar posteriormente para el debido entrenamiento del modelo identificador de posturas corporales sospechosas.

Una vez que se extrajeron cada una de las corporales y se creó el dataframe con la ayuda de las coordenadas normalizadas respecto al ROI, se obtuvo finalmente como resultado un conjunto de datos con 2995 registros. La estructura del dataframe se compone de las siguientes etiquetas; N para comportamientos Normales, A para Comportamientos anómalos con Armas y la C para Comportamientos anómalos con cuchillo.

3.2.3 Obtención de datos de geoposicionamiento (latitud y longitud) con tecnología GPS

a. Captura de datos con módulo GPS NEO-6M

El módulo GPS Neo 6m tiene lugar del área dedicada para la obtención de datos de geoposicionamiento de la partícula de estudio, debido a que es un módulo robusto y con excelentes características técnicas de adaptabilidad ante los requerimientos que posee el sistema. Dentro del tema relacionado a la captura de datos con este módulo,

aquellos datos son enviados mediante interfaz serial a una placa microcontroladora Arduino UNO.

b. Procesamiento y transmisión de datos con Arduino Uno

Se hizo uso de una placa de desarrollo microcontrolador Arduino UNO como un intermediario entre el módulo GPS y el sistema de gestión de información desarrollado con Flask. El Arduino fue programado para realizar lecturas de tramas NMEA recibidas a través del puerto serial y extraer únicamente los campos relevantes, como son: la latitud y la longitud, mismos que son enviados de manera ininterrumpida mediante comunicación serial al entorno de Node-RED levantado dentro del host local.

Vale mencionar que dentro de la programación realizada dentro del IDE de Arduino se importó librerías muy necesarias e importantes para la lectura de datos por parte del módulo GPS con los satélites como es el caso de la librería "<TinyGPS++.h>" y de la librería "<SoftwareSerial.h>" para la lectura de datos seriales. La Figura 60 exhibe el código de programación dentro del IDE de Arduino, para la obtención de los datos de geoposicionamiento de parte del módulo GPS hacia el microcontrolador; Arduino UNO.

```
#include <TinyGPS++.h>
#include <SoftwareSerial.h>

TinyGPSPlus gps;
SoftwareSerial gpsSerial(4, 3); // RX=4, TX=3 (ajusta según conexión)

void setup() {
  Serial.begin(9600);       // Serial para comunicación con ESP32
  gpsSerial.begin(9600);    // Serial para comunicación con GPS
}

void loop() {
  while (gpsSerial.available() > 0) {
    gps.encode(gpsSerial.read());
  }

  if (gps.location.isUpdated()) {
    String data = String(gps.location.lat(), 6) + "," + String(gps.location.lng(), 6);
    Serial.println(data);  // Envía coordenadas por serial al ESP32
  }
}

```

Figura 60. Código para el Procesamiento y transmisión de datos con Arduino Uno

c. Recepción y redirección de datos mediante Node-RED

El Arduino UNO solo habilita una conexión serial simultánea, por dicho motivo se optó utilizar Node-RED como una plataforma intermedia con el fin de evitar el bloqueo de puertos que se presentaron al intentar la lectura directa desde Flask. Node-RED se ejecuta de manera local en la computadora, recepta los datos del puerto serial, los parsea y los convierte en un mensaje de formato JSON legible, que después es reenviado por medio del protocolo MQTT. La Figura 61 presenta una entrada "in" de una comunicación serial y una salida "output" de una comunicación con el protocolo mqtt.

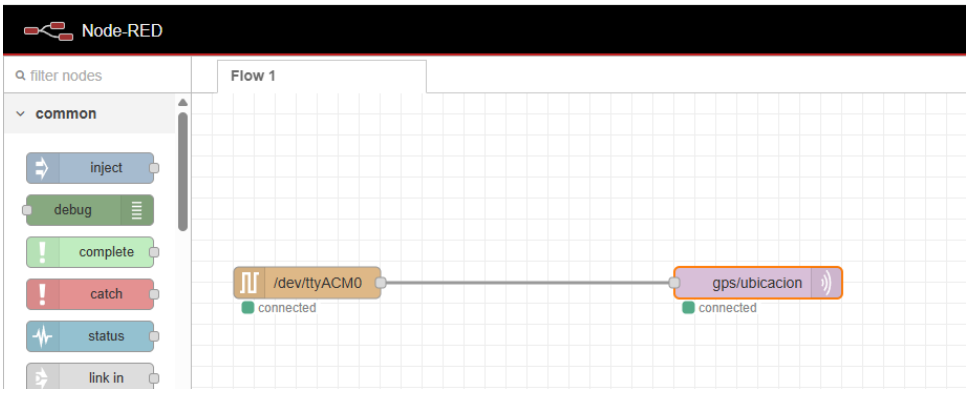


Figura 61. Interfaz gráfica de Node-red.

Dentro de la entrada Serial se verifica que las configuraciones estén realizadas de manera correcta, como es el caso del número de bits que se utilizan para representar cada carácter de datos que se transmite (data bits) sea igual a 8, que los baudios se encuentren configurados a 9600; cantidad común de operación para comunicaciones seriales dentro de Arduino y finalmente que el puerto sea el correcto dentro de la conexión física. La Figura 62 muestra las configuraciones para tener en cuenta para que la conexión del puerto serial sea exitosa.

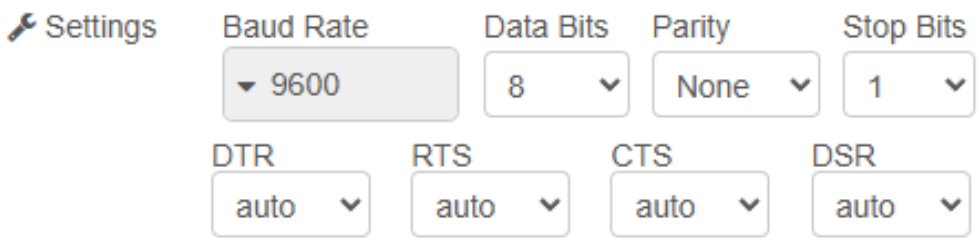


Figura 62. Configuraciones básica del puerto serial.

Con respecto a la salida con el protocolo de comunicación Mqtt, se tuvo en cuenta que el servidor fue la Unidad de Procesamiento NVIDIA Jetson Nano, por lo tanto, el "server" fue identificado como "localhost", el puerto de utilización fue el 1883; puerto por donde opera el protocolo de comunicación Mqtt y el Topic correspondió a "gps/ubicacion", (tópico que responde a la comunicación Mqtt dentro del archivo app.py de flask). La Figura 63 muestra las configuraciones para tener en cuenta para que la conexión de la etiqueta de la comunicación MQTT sea exitosa.



Figura 63. Configuraciones básicas del protocolo MQTT.

d. Almacenamiento y gestión con sistema Flask

Finalmente, la información obtenida a través de MQTT es capturada por un servicio creado con el microframework Flask, cuyo objetivo es guardar los datos georreferenciados en una base de datos en SQLite estructurada para su análisis futuro. Esta estructura modular facilita una obtención eficaz, segura y adaptable de la información en tiempo real.

3.3 Creación del modelo

En este punto se analiza los tamaños o versiones de los modelos de entrenamiento Yolo versión 8 y versión 11, tomando en cuenta siempre la adaptabilidad del modelo final entrenado, con los recursos de la unidad de procesamiento y los parámetros de evaluación del sistema, procurando siempre contar con una relación coherente entre ambos factores.

3.3.1 Fase de entrenamiento y validación del modelo

Se ejecutó el debido entrenamiento y posteriormente la validación del modelo de detección de objetos amenazantes para el cometimiento de actos ilícitos y de posturas sospechosas. Se puso en ejecución varios entrenamientos con distintas variaciones con respecto a los parámetros técnicos de entrenamiento como son el caso del tamaño del batch, el número de épocas, el redimensionamiento de las imágenes que en este caso es 640 x 640 megapíxeles. Para el entrenamiento del modelo de identificación de posturas corporales sospechosas se configura el "kernel" para lo que corresponde a SVM. A continuación, en la Figura 64 se muestra un esquema general de entrenamiento del sistema para ambos modelos.

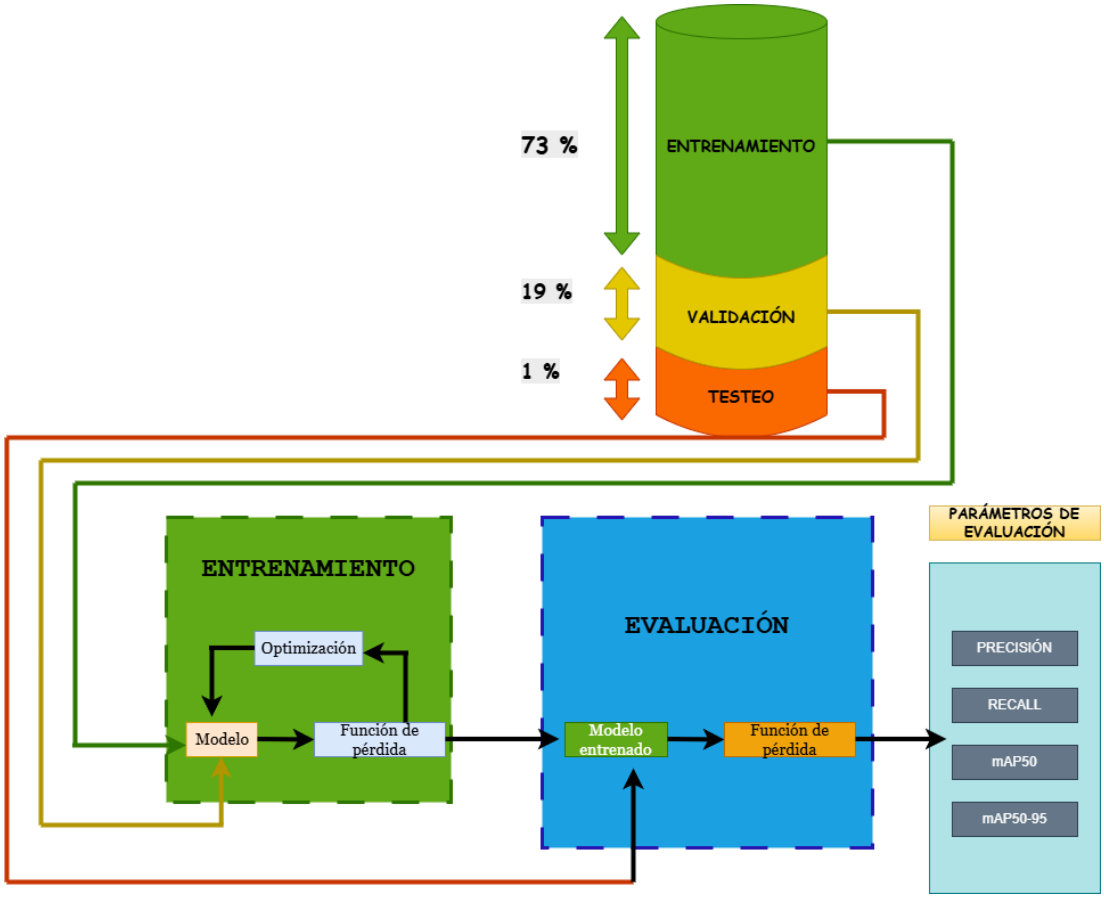


Figura 64. Esquema general de entrenamiento del sistema para los modelos (YOLO y SVM).

El esquema de entrenamiento del sistema representa la dinámica mediante la cual el modelo ajusta internamente sus parámetros, basándose en la comparación entre sus

predicciones y los valores reales durante cada iteración. Una vez finalizada esta etapa de ajuste, se procede a verificar el rendimiento del modelo utilizando un subconjunto de datos que no fue expuesto previamente durante el aprendizaje. Los indicadores obtenidos en esta evaluación sirvieron como base para determinar cuál modelo presentaba el mejor desempeño.

a. Modelo de detección de objetos amenazantes para el cometimiento de actos ilícitos.

En este punto se llevó a cabo el entrenamiento con distintas versiones o tamaños de YoloV8 y Yolo11, ambos con el mismo banco de datos que se armó, se etiquetó y finalmente se descargó desde Roboflow. Todo esto con la finalidad de poder elegir el modelo más adecuado para el funcionamiento del sistema en base a los parámetros de evaluación. En las estructuras Yolo de ambas versiones se entrenó con las versiones (nano n, small s y medium m). Solo se utilizó esas tres versiones con la finalidad de evitar contar con un modelo entrenado sobrecargado o muy pesado para la unidad de procesamiento que se utilizó.

A continuación, la Tabla 16 presenta una comparativa entre las versiones de los tipos de modelos recientemente mencionados.

Tabla 16. Comparativa entre las versiones de los tipos de modelos Yolo versión 8 y 11.

Tamaño	**Parámetros**	**Gradientes**	**GFLOPs**
YOLOv8n	3 011 823	3 011 807	8.2
YOLOv8s	11 137 535	11 137 519	28.7
YOLOv8m	25 859 215	25 859 199	79.1
YOLOv11n	~3 500 000	~3 500 000	~8.5
YOLOv11s	~11 800 000	~11 800 000	~29.0
YOLOv11m	~26 200 000	~26 200 000	~78.0

El modelo de detección se entrenó con las versiones y tamaños mencionados, con intermedias diferencias en los valores correspondientes a los parámetros de entrenamiento, esto con el fin de probar y analizar el resultado de los entrenamientos de los modelos. La Tabla 17 describe de manera muy breve los parámetros que fueron parte de cada uno de los tamaños de Yolo versión 8 y 11 que se colocaron en el entrenamiento, evaluando la capacidad de la T4 GPU para el procesamiento de las imágenes.

Tabla 17. Parámetros que se tomaron en cuenta para cada uno de los tamaños de Yolo versión 8 y 11.

Versión y Tamaño de Yolo	**Epochs**	**Batch**	**Image_size**	**Plots**
8 nano	30-50-75	64	640	True
8 small	30-50-75	16-4-32	640	True
8 medium	40-50-60	16-4-32	640	True
11 nano	30-50	64	640	True
11 small	30-50	8-16	640	True
11 medium	40-50-60	4	640	True

Una vez que se llevó a cabo los entrenamientos de todas las versiones y tamaños se revisó cada una de las gráficas de resultados para tomar a partir de ahí el mejor modelo de entrenamiento. La Tabla 18 muestra los resultados de cada uno de los entrenamientos realizados.

Tabla 18. Resultados de cada uno de los entrenamientos de Yolo realizados.

Modelo	**Épocas**	**Parámetros de evaluación**				**Duración de entrenamiento (Horas)**	**Tamaño**	**Inferencia**
		Presición	**Recall**	**mAP50**	**mAP50-95**			
Yolov8n	30	0.888	0.721	0.772	0.478	2.788	65.0 MB	13.9 ms
	50	0.857	0.689	0.767	0.476	3.219	52.0 MB	11.3 ms
	75	0.848	0.588	0.696	0.412	2.110	6.3 MB	12.3 ms
Yolov8s	30	0.812	0.640	0.732	0.433	1.985	52.0 MB	10.7 ms
	50	0.837	0.670	0.759	0.466	2.304	52.0 MB	10.4 ms
	75	0.848	0.678	0.773	0.480	2.918	6.1 MB	12.1
Yolov8m	40	0.855	0.697	0.779	0.484	2.076	52.0 MB	10.3 ms
	50	0.867	0.668	0.777	0.503	2.765	52.0 MB	10.4 ms
	60	**0.889**	**0.685**	**0.788**	**0.511**	**3.112**	**52.0 MB**	**10.0 ms**
Yolov11n	30	0.801	0.620	0.710	0.390	2.101	50.0 MB	9.8 ms
	50	0.819	0.635	0.741	0.428	2.800	50.0 MB	9.8 ms
Yolov11s	30	0.810	0.628	0.720	0.412	2.245	51.2 MB	10.1 ms
	50	0.835	0.665	0.755	0.445	2.907	51.2 MB	10.2 ms
Yolov11m	40	0.853	0.674	0.766	0.49	2.798	50.6 MB	10.4 ms
	50	0.865	0.675	0.770	0.489	2.980	51.8 MB	10.3 ms
	60	0.876	0.680	0.782	0.504	3.080	51.8 MB	10.2 ms

Una vez que se culminó con el entrenamiento de cada uno de los modelos, se revisaron los resultados por medio de la Tabla 18, misma que ayudó a elegir el entrenamiento que corresponde a la versión de yolov8m entrenada con 60 épocas, con una precisión de 0.889, un mAP50 de 0.788 y un tiempo de inferencia de 10 ms, lo cual evidencia que el modelo es ideal para aplicaciones de detección en tiempo real. La Figura 65 presenta los resultados del modelo escogido, donde también se muestra el rendimiento de este a lo largo del proceso de entrenamiento.

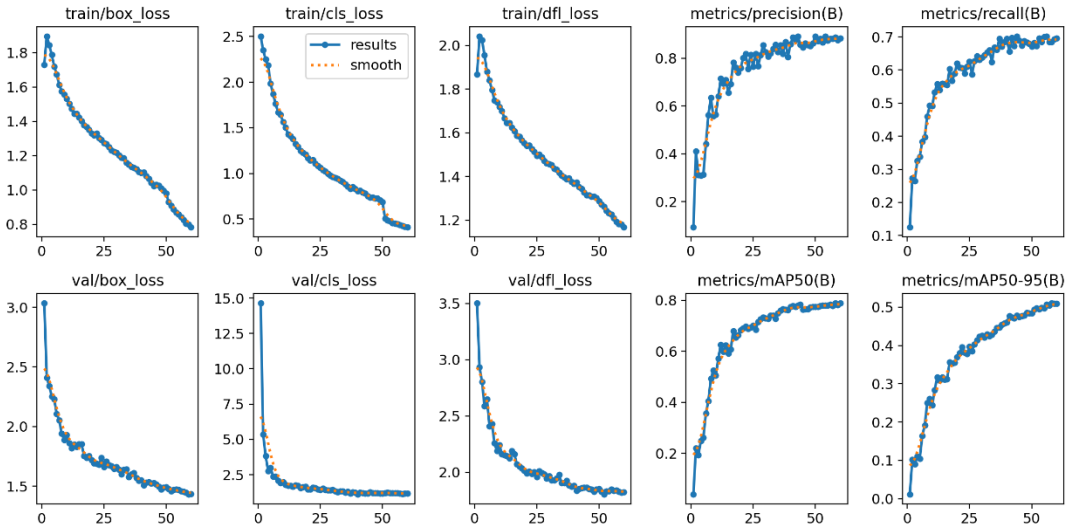


Figura 65. Resultados del modelo escogido; Yolo versión 8 medium con 60 épocas.

Estas gráficas señalan que los parámetros de evaluación poseen una tendencia significativa que va en dirección ascendente. En el tramo final del entrenamiento siempre se almacena el mejor modelo y el modelo final, el mejor modelo se encuentra en el interior de las carpetas /train/weights y se lo identifica porque siempre va a tener el nombre de “best.pt”. La Figura 66 presenta la matriz de confusión del modelo seleccionado.

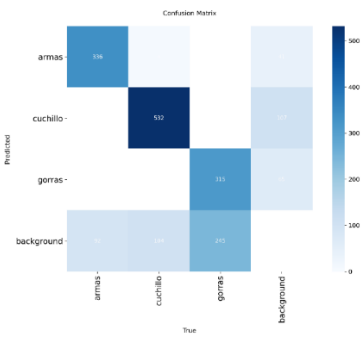


Figura 66. Matriz de confusión del modelo seleccionado.

La Figura 66, misma que corresponde a la matriz de confusión del modelo, proporciona información relevante y de gran importancia acerca del rendimiento del modelo entrenado, este modelo presenta un rendimiento del modelo de detección de objetos pertenecientes a las clases o etiquetas: armas, cuchillo, gorras y background. En líneas generales, este modelo muestra un desempeño óptimo, evidenciado específicamente por la alta concentración de las predicciones correctas sobre la diagonal principal, lo que representa una baja confusión entre las clases. Como conslusión general el modelo muestra una capacidad eficiente de detección y clasificación, con un bajo nivel de confusión entre clases reales, lo cual resulta ser muy importante dentro de tareas de detección en sistemas de seguridad o de vigilancia.

A continuación, la Figura 67 presenta la curva de precisión que se maneja en función del nivel de confianza dedicada hacia tres clases específicas: armas, cuchillos y gorras, junto con una curva que está combinada para todas las clases. Pues también, se observa que, para cada clase individual, la precisión va en aumento a medida que la confianza del modelo de detección va en incremento, tendiendo a estabilizarse cerca ya del valor máximo de precisión. En particular, esta curva combinada de todas las clases llega a alcanzar una precisión máxima de 1 cuando la confianza posee un valor aproximado de 0.944. Esto indica que, cuando el modelo cuenta con una alta confianza en sus predicciones, el desempeño del modelo es excelente, reflejando una alta precisión en la detección y en la clasificación correcta de los objetos dentro de las distintas categorías.

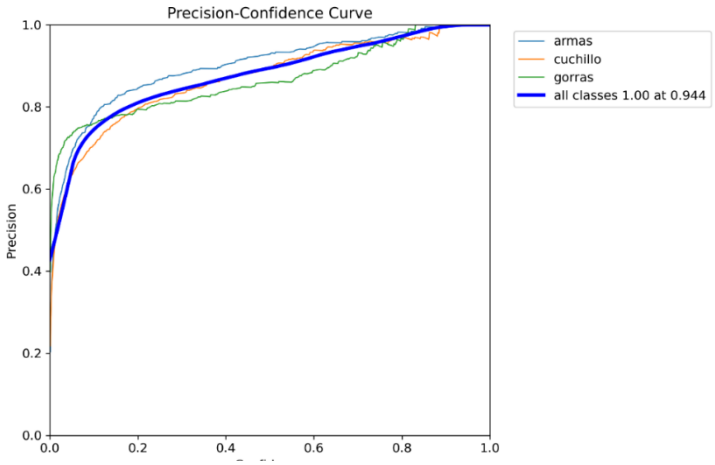


Figura 67. Curva de precisión que se maneja en función del nivel de confianza dedicada hacia tres clases específicas: armas, cuchillos y gorras.

La Figura 68 muestra la curva Recall-confianza, misma que representa la relación entre el Recall (la tasa de verdaderos positivos) y los distintos niveles de confianza para las clases evaluadas. Se observa que el modelo logra alcanzar un valor de recall cercano a 0.80 para todas las clases combinadas con un umbral de confianza muy bajo (cerca de 0.0). En cuanto a las clases individuales, el desempeño es mejor para las clases “armas” y “cuchillo”, debido a que muestra un recall superior a 0.75 en valores bajos de confianza, mientras que la clase “gorras” presenta un recall considerablemente inferior. Esta gráfica evidencia que el modelo es más efectivo en la detección de armas y cuchillos en comparación con las gorras, específicamente en umbrales bajos de confianza.

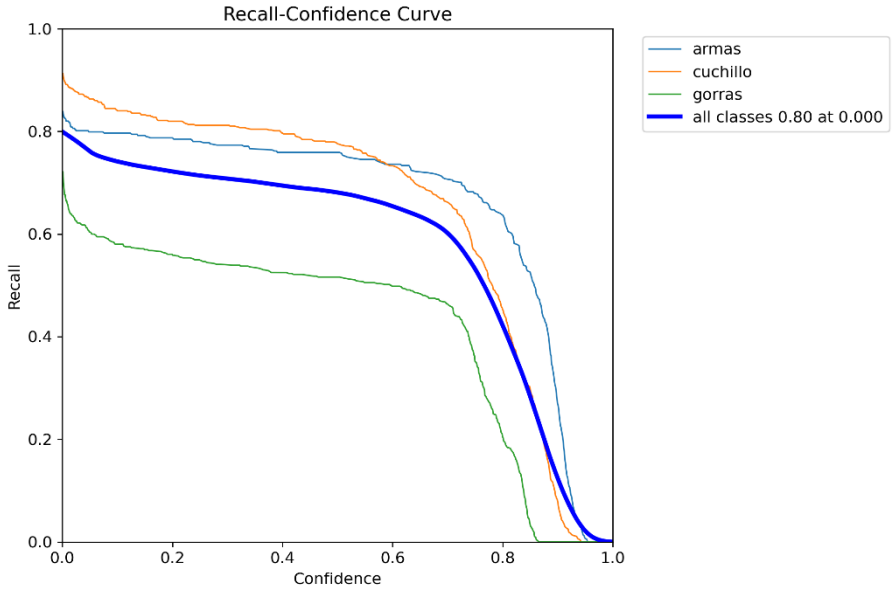


Figura 68. Curva Recall-confianza, misma que representa la relación entre el Recall (la tasa de verdaderos positivos) y los distintos niveles de confianza para las clases evaluadas.

La Figura 69 exhibe la correlación entre el F1-score y diversos valores de confianza, en los que esta medida toma en cuenta tanto la exactitud como el recall, proporcionando una evaluación equilibrada del rendimiento del modelo. Las curvas relacionadas con cada tipo de objeto muestran comportamientos diferentes; no obstante, la curva que comprende todas las clases se mantiene estable en un valor máximo cercano a 0.77 con un límite de confianza cercano a 0.47. Este límite es vital, pues se aplicó para determinar el valor de confianza en el código de procesamiento del

backend del proyecto, eliminando automáticamente las detecciones con valores de confianza más bajos con el fin de mejorar la exactitud sin sacrificar demasiado el recall. Adicionalmente, se nota que las clases "armas" y "cuchillo" obtienen puntuaciones F1 que superan el 0.8, lo que demuestra un rendimiento óptimo del modelo para estos objetos, mientras que la clase "gorras" muestra una puntuación inferior, lo que sugiere un área de mejora en el entrenamiento. En conclusión, la correcta elección del límite de confianza fundamentado en la curva F1 facilita un equilibrio eficaz entre la tasa de falsas detecciones y las omisiones, garantizando un desempeño sólido y fiable del sistema en su uso práctico.

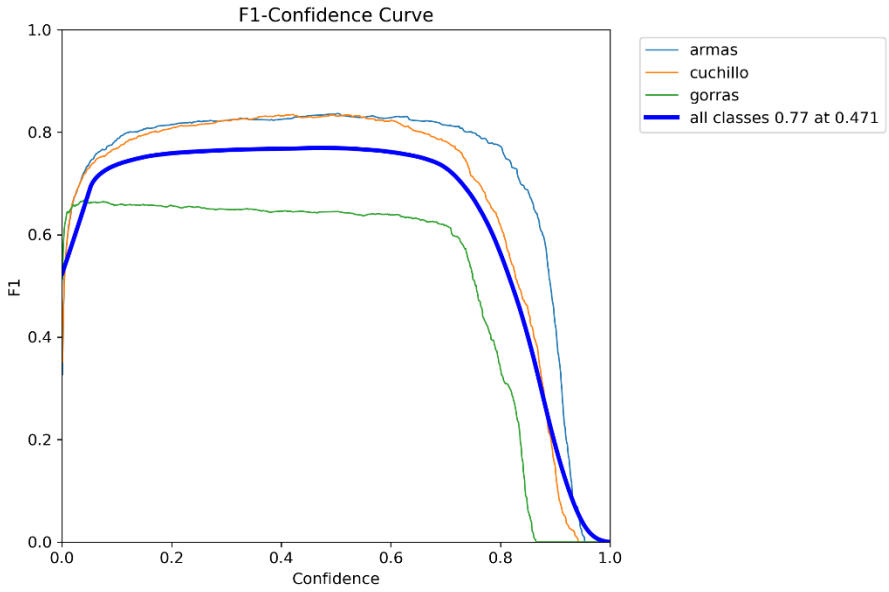


Figura 69. Correlación entre el F1-score y diversos valores de confianza

Como se acabó de mencionar recientemente; el modelo que mejores resultados se obtuvo dentro de los parámetros de evaluación fue el que se entrenó con 60 épocas y con Yolov8m, con el cual se realizó pruebas preliminares. Pero a pesar de ello, se llevó a cabo la modificación del dataset dentro de la plataforma Roboflow; específicamente en el balanceo exacto de la cantidad de etiquetas que poseía el dataset originalmente y se procedió a entrenar el modelo con 40 épocas y con yolov8m; donde se obtuvo parámetros de evaluación más altos y estables. La Tabla 19 detalla cada uno de los parámetros de evaluación obtenidos por este modelo.

Tabla 19. Párametros de evaluación del modelo Yolov8 m con 40 épocas.

Modelo	Época	Parámetros de evaluación				Duración de entrenamiento (Horas)	Tamaño	Inferencia
		Presición	Recall	mAP50	mAP50-95			
Yolov8m	40	0.901	0.912	0.946	0.792	1.950	52.0 MB	10.3 ms

La Figura 70 presenta los resultados del modelo presentado en la Tabla 19, donde también se muestra el rendimiento de este a lo largo del proceso de entrenamiento, en donde se visualiza que el entrenamiento se estabilizó con este número de épocas y con un batch de 32.

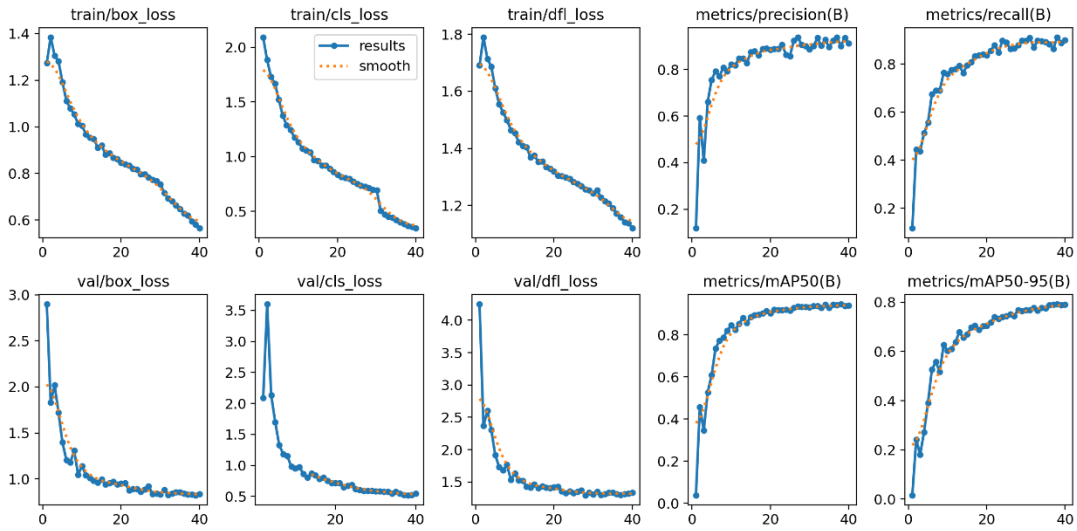


Figura 70. Resultados del modelo escogido; Yolo versión 8 medium con 40 épocas.

La Figura 71, que presenta la matriz de confusión de este modelo, proporciona información clave y detallado referente al rendimiento del sistema de detección de objetos. Aquel modelo fue entrenado con el fin de identificar las clases o etiquetas: armas, cuchillo, gorras y background (fondo), y a breves rasgos, demuestra tener un desempeño óptimo, mismo que se evidencia de forma contundente por la concentración significativa de las predicciones correctas que se ubican a lo largo de la diagonal principal. Dicha característica es un claro indicador de la baja confusión entre las clases.

Teniendo de la Figura 71 los datos exactos:

Para la clase o etiqueta “armas”, dicho modelo identificó de forma correcta 128 instancias en total. La clase “cuchillo”, se realizaron en total 126 predicciones

correctamente Para la clase “gorras”, se obtuvo un total de 132 aciertos. Finalmente, la clase “background” fue clasificada de forma correcta en un total de 128 ocasiones.

Las cifras que se encuentran fuera de la diagonal principal son muy bajas, lo que proporciona una reafirmación de la capacidad del modelo para lograr distinguir de manera muy precisa entre las distintas categorías. Por ejemplo, tenemos; solo 22 instancias de “armas” que fueron clasificadas de forma errónea como “background”, 3 instancias de la etiqueta “cuchillo” como “background” y 5 de “gorras” como “background”. De la misma forma, las confusiones de “background” con otras etiquetas, tales como; (20 como “armas”, 8 como la etiqueta “cuchillo” y 20 como “gorras”) son relativamente inferiores, mostrando hasta cierto punto una buena discriminación del fondo con respecto a los otros objetos de interés.

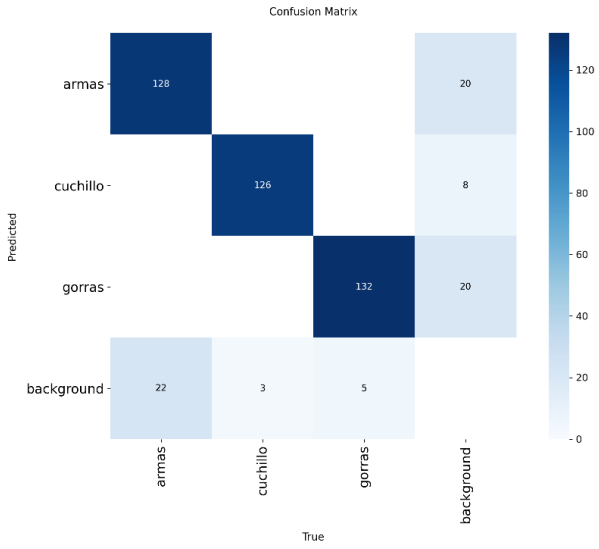


Figura 71. Matriz de confusión.

La Figura 72, muestra la curva de precisión que es manejada en función del nivel de confianza, dirigida hacia tres clases muy específicas, aquellos son; armas, cuchillo y gorras, junto con una curva que se encuentra combinada para todas las clases. Pues también se observa que, para cada una de las clases, la precisión va en incremento a medida que la confianza del modelo de detección va en aumento, tendiendo a estabilizarse cerca ya del valor máximo de precisión. Finalmente, se puede deducir que esta curva combinada con el resto de las clases llega a adquirir una precisión máxima de 1.00 cuando la confianza cuenta con un valor aproximado de 0.952. Lo cual

representa que el modelo cuenta con una alta confianza dentro de sus predicciones, el desempeño del modelo es excelente, mostrando así; una precisión muy alta dentro de la detección y en la clasificación de objetos dentro de cada una de las categorías.

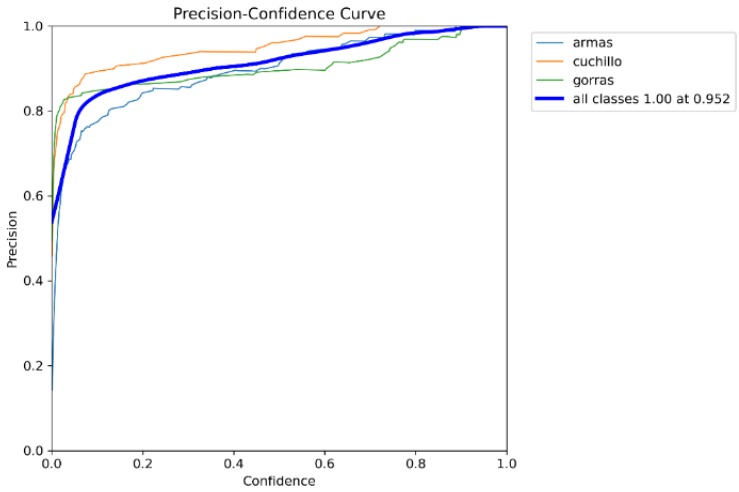


Figura 72. Curva de precisión que es manejada en función del nivel de confianza, dirigida hacia tres clases muy específicas, aquellos son; armas, cuchillo y gorras.

La Figura 73, es una Curva de Precisión-Confianza, y aquella es una herramienta fundamental e importante para la evaluación del rendimiento dentro de un modelo de detección de objetos. El eje X (Confianza) es la representación del umbral de confianza que el modelo hace una asignación a cada una de sus detecciones, donde un valor elevado representa que el modelo está muy seguro de la predicción que hace, mientras que el eje Y (el de la precisión) representa específicamente la precisión del modelo, que es la proporción de las detecciones positivas que son correctas. Aquella curva presenta la variación de la precisión del modelo a medida que se ajusta el umbral de confianza para la aceptación de una detección, es decir; si se exigió una confianza muy alta, el modelo mostró menos predicciones, pero las que realizó fueron bastante precisas. En cambio, si el modelo hizo más predicciones y se redujo la confianza, algunas de ellas pudieron ser erróneas. Cada línea de color (azul para “armas”, naranja para “cuchillo” y verde para “gorras”) representa el comportamiento preciso-confianza para una clase específica, mientras que la línea azul gruesa (“todas las clases 1,00 a 0,952”) representa el comportamiento combinado o medio de todas las clases.

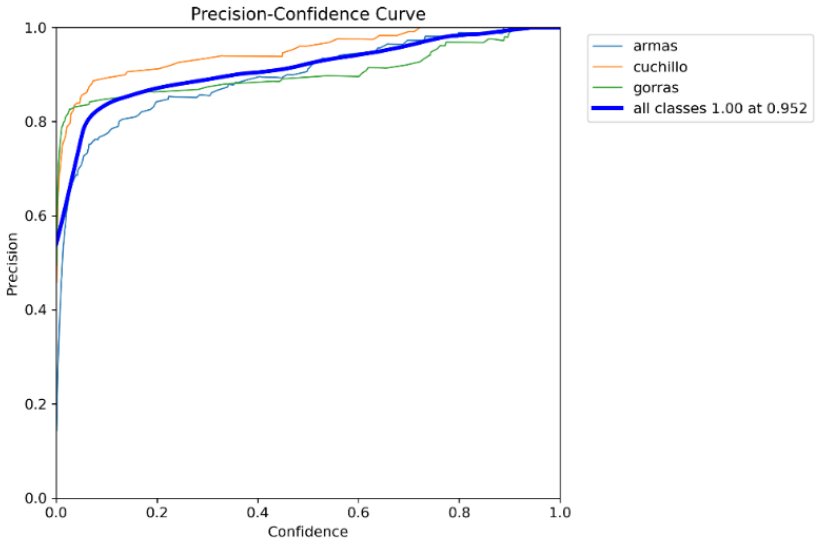


Figura 73. Curva de Precisión-Confianza, y aquella es una herramienta fundamental e importante para la evaluación del rendimiento dentro de un modelo de detección de objetos.

La Figura 74 exhibe la Curva F1-Confianza, donde evalúa el equilibrio o la relación entre la precisión y el recall del modelo a diferentes umbrales de confianza. Aquella gráfica muestra la forma en que el puntaje F1 para cada una de las clases (armas, cuchillo, gorras) y para todas las clases combinadas (en este caso la línea azul gruesa) aumenta inicialmente con la confianza, logra alcanzar un pico, y posteriormente disminuye de forma muy drástica hacia umbrales de confianza muy elevados. Siendo muy específicos, se puede concluir que el F1 score combinado con todas las clases logra alcanzar su puntaje máximo de 0.90 con un umbral de confianza de 0.356. Por lo que este punto resulta ser muy crucial o clave, puesto que indica la confianza óptima para alcanzar el mejor balance entre la detección correcta de los objetos y la minimización de los falsos positivos y falsos negativos, reflejando un rendimiento general muy robusto o compacto del modelo.

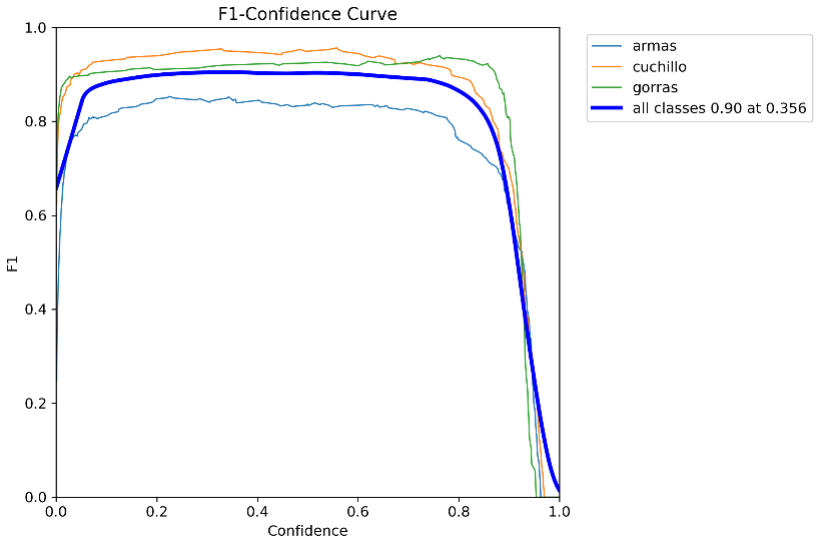


Figura 74. Curva F1-Confianza, donde evalúa el equilibrio o la relación entre la precisión y el recall del modelo a diferentes umbrales de confianza.

b. Modelo identificador de posturas sospechosas.

Se llevó a cabo el entrenamiento y la evaluación de un modelo dedicado a clasificación de tipo Support Vector Machine o por sus siglas (SVM) con un kernel RBF por medio de un script de Python, que se diseñó para la categorización de datos basados en características de puntos clave o (landmarks), mismos que fueron extraídos de un archivo CSV o Excel con el nombre de (etiquetas_landmarks.csv).

Las clases o las etiquetas específicas dedicadas para la clasificación son derivadas de forma directa de un conjunto de datos de entrada. Este proceso de entrenamiento incluye la división de los datos en conjuntos de entrenamiento y de prueba, con una distribución de (80/20). Es crucial destacar que, a diferencia de un ajuste de los hiperparámetros explícito como una búsqueda de malla, dicho modelo SVM hace uso de un valor de gamma=”scale” predeterminado. Una vez ya entrenado este modelo, se guarda para su uso a futuro.

Para la evaluación del rendimiento del entrenamiento, el script realiza el cálculo de métricas importantes o cruciales como es el caso de la precisión (accuracy), genera un reporte de clasificación muy detallado, donde se incluye; la precisión, el recall y el f1-score por cada clase, también construye una matriz de confusión, genera las respectivas curvas ROC (Receiver Operating Characteristic) para cada una de las clases (haciendo

uso de la estrategia "one-vs-rest" para multiclasificación) y finalmente se hace el cálculo del área bajo la curva (AUC) respectiva. Cada uno de estos resultados, incluyendo a las métricas textuales y a las visualizaciones gráficas de la matriz de confusión y las curvas ROC, se almacenan en una carpeta de Resultados.

La Figura 75, demuestra que el modelo cuenta con una capacidad de discriminación totalmente excepcional entre cada una de las clases; “AsaltoArma”, “AsaltoCuchillo” y “Normal”, evidenciada por valores del área bajo la curva bastante altos (0.99 y 1.00). Lo cual, esto indica que el modelo es muy efectivo para lograr distinguir de forma correcta las situaciones de amenaza de las normales, lo cual es muy importante para aplicaciones de seguridad y de vigilancia que demandan una fiabilidad muy grande.

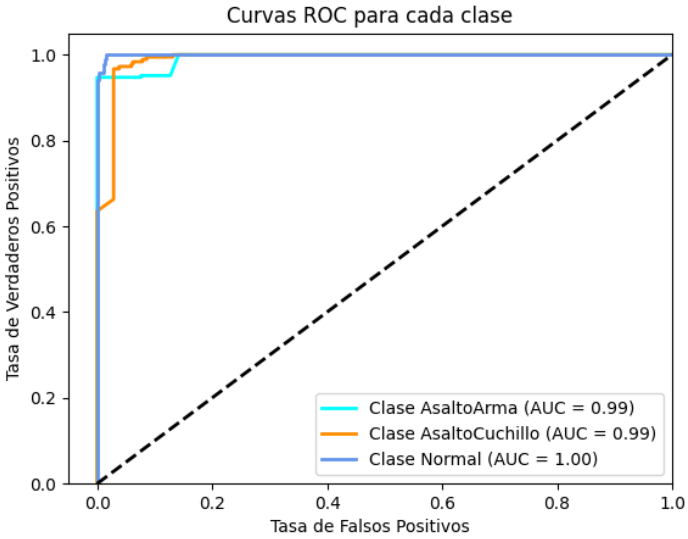


Figura 75. Curvas ROC para cada clase que forman parte del entrenamiento.

La Figura 76, presenta una matriz de confusión que se encarga de evaluar el rendimiento de un modelo de clasificación para los tres tipos de clases "AsaltoArma", "AsaltoCuchillo" y "Normal", en donde las etiquetas verdaderas son representadas en las filas y en cambio las predichas son representadas por las columnas. Por otra parte, los valores que están dentro de la diagonal principal muestran las predicciones correctas, mientras que los valores que se encuentran fuera de ella son los incorrectos en cambio. Se observa también, un rendimiento muy alto; con 233 instancias de “AsaltoArma” y 175 de “Asalto Cuchillo” fueron clasificadas correctamente, con mínimas confusiones (12 y 1, y 6 respectivamente, con otras clases). Destaca

directamente, el rendimiento perfecto para la clase “Normal”, donde las 172 instancias se clasificaron adecuadamente sin ninguna confusión como “AsaltoArma” o “AsaltoCuchillo”.

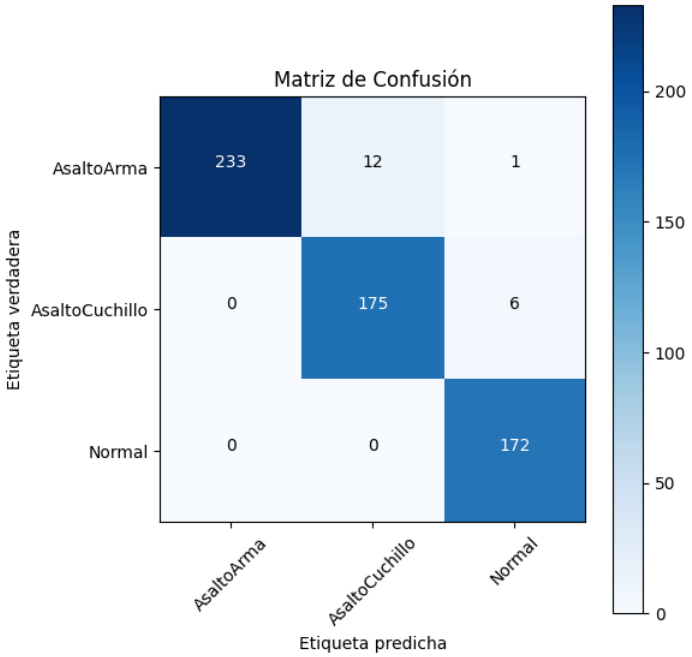


Figura 76. Matriz de confusión que se encarga de evaluar el rendimiento de un modelo de clasificación para los tres tipos de clases "AsaltoArma", "AsaltoCuchillo" y "Normal".

3.4.2 Plataforma de desarrollo para el control de la información

a. Ajustes preliminares

Como la Unidad de procesamiento que se utilizó es la Tarjeta Nvidia Jetson Nano, misma que cuenta con un sistema operativo Ubuntu 20.04, se llevó a cabo la ejecución de ciertos ajustes preliminares del micro-framework web de Python; Flask. Como primer paso se procedió a actualizar la lista de paquetes instalados dentro del sistema operativo y posteriormente se procedió a crear un entorno virtual con la finalidad de evitar conflictos de dependencias y porque se trata específicamente de la ejecución de un proyecto. El entorno virtual se denominó “venv” y dentro de este entorno virtual se instaló todos los paquetes, librerías y dependencias fundamentales para la ejecución de este. La Figura 77 expone los paquetes instalados dentro del entorno virtual “venv”.

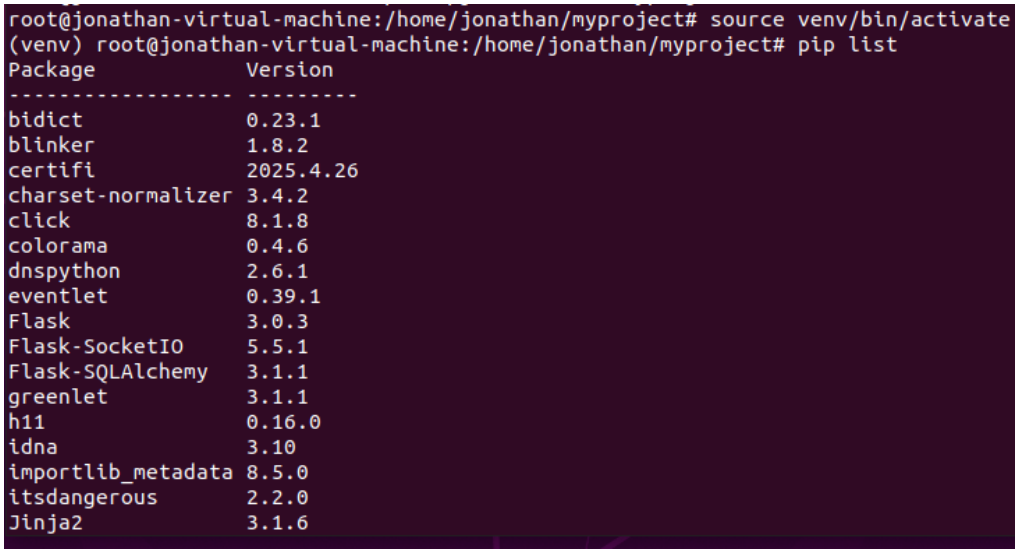


Figura 77. Paquetes instalados dentro del entorno virtual “venv”.

Dentro de los ajustes preliminares y a la vez fundamentales consta la habilitación del puerto 5000 dentro del sistema operativo de Ubuntu, dado que este puerto es por donde se expone el fronted del proyecto desarrollado con Flask. La Figura 78 muestra que el puerto 5000 está permitiendo el tráfico de información por dicho puerto.

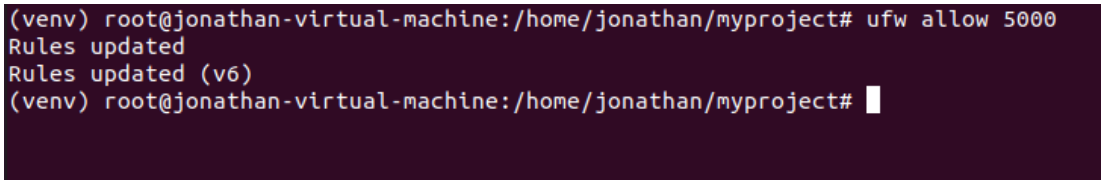


Figura 78. Tráfico de información por el puerto 5000.

Siguiendo con la habilitación de puertos, es muy importante habilitar el puerto 1883 y comprobar que por este puerto también está fluyendo el tráfico de información, puesto que se trabajó con el protocolo de comunicación mqtt, para el envío de datos de geoposicionamiento obtenidos del módulo GPS Neo 6m mediante el intermediario Node-red. La Figura 79 muestra que el puerto 1883 se encuentra habilitado.

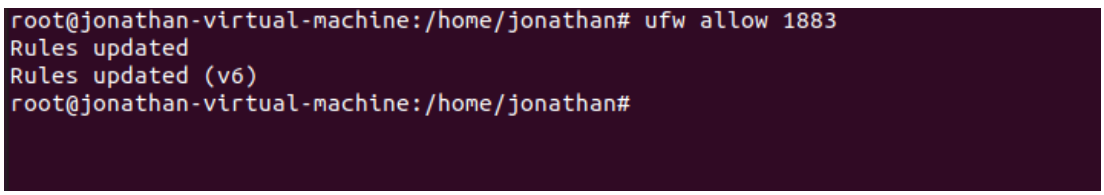


Figura 79. Tráfico de información por el puerto 1883.

b. Estructura del entorno virtual

Dentro del entorno virtual creado se cuenta con una estructura básica para el funcionamiento de este, donde el backend se encuentra gestionado por el archivo principal, denominado como “app.py”, también cuenta con cinco carpetas: “ModeloYOLO” y en su interior cuenta con los modelos entrenados con Yolo versión 8 y versión 11 para la detección de objetos amenazantes para el cometimiento de actos ilícitos, otro directorio es “Posturas” y en su interior cuenta con el modelo entrenado dedicado para la detección de posturas corporales sospechosas, la carpeta “templates” contiene en su interior los archivos fronted o para ser más específicos los archivos con extensión .html, la carpeta “Static” contiene en su interior los archivos necesarios para llevar a cabo el funcionamiento de la interfaz de usuario o la plataforma de desarrollo para el control de la información.

Dentro de la estructura del entorno virtual también se encuentra la base de datos que gestiona la autenticación del usuario que ingrese a la plataforma de desarrollo, posteriormente la base de datos se extiende en el control y almacenamiento de los datos de geoposicionamiento y de las detecciones de actos ilícitos que el sistema reconozca.

c. Backend system

El backend entra en funcionamiento y es gestionado por medio del archivo “app.py” una vez que este se lo pone en ejecución, como primera etapa; el sistema procede a hacer una validación de autenticación del usuario que va a ingresar al sistema, si la autenticación se completa de manera exitosa el sistema redirige a la página web guardada en el directorio “templates” y denominada como “index.html”, y es en esta ventana donde se muestra el video en tiempo real junto con los indicadores del entrenamiento del modelo y los datos de geoposicionamiento adquiridos por el módulo GPS con el que se está trabajando.

El sistema inicia ahora con la configuración de la aplicación Flask, la base de datos SQLite con SQLAlchemy (para tablas User, GPSData, Detection), y con las integraciones de SocketIO y Paho MQTT para una comunicación en tiempo real y la adquisición de los datos del módulo GPS, respectivamente. Los modelos de Machine Learning cargados incluyen un SVM (modelo_entrenado_pose.sav) para la estimación

de posturas corporales sospechosas y un modelo YOLO (best.pt) para la detección de los objetos amenazantes. Adicional a eso, se determinan los parámetros para lograr la integración con Google Drive API para el almacenamiento de las capturas y con la aplicación Slack para el envío de las alertas, utilizando un webhook configurado.

El proceso de análisis de video, controlado desde la parte del usuario o fronted, empieza con la inicialización de la captura de video desde el dispositivo de captación (webcam). Donde por cada fotograma, se hace una estimación de la postura corporal por medio de la librería MediaPipe. Los puntos clave que se ubican en la parte superior del cuerpo son utilizados para calcular una región de interés o ROI, misma que se expande con un margen del 50 % en el ancho y un 75 % en el alto. La postura que se encuentra dentro de esta región de interés es clasificada por el modelo SVM entrenado, y se asigna un pose_score de 0 para la situación "Normal", 70 para la situación "AsaltoCuchillo" y 100 para "AsaltoArma".

Únicamente esta ROI es procesada por el modelo YOLO para la detección de los objetos como "gorras", "armas" y "cuchillos". Se calcula un object_score acumulativo con valores de 25 para las "gorras", 100 para las "armas" y 80 para los "cuchillos". Este valor oscila en un rango de 0 a 100.

La clasificación final del evento, es decir; (etiqueta) es determinada por la combinación de object_score y pose_score. Determinando como: "Normal" donde object_score es inferior o igual a 25, "Sospechoso" si object_score oscila entre 25 y 60 y pose_score es inferior a 30, un "Delito Flagrante" se determina si object_score es superior a 60, o si object_score se encuentra entre 25 y 60 y pose_score es superior o igual a 30 (postura amenazante). La detección de "cuchillo" o "armas" también fuerza la etiqueta de "Delito Flagrante".

Las alertas son activadas y enviadas cuando la etiqueta es "Delito Flagrante" o "Sospechoso", o si el sistema detecta "cuchillos" o "armas". Cada 30 fotogramas, se almacena una captura del fotograma, éste se sube a Google Drive para lograr obtener una URL pública, y se registra la detección (incluyendo el object_score, pose_score y finalmente las coordenadas del GPS obtenidas por el protocolo de comunicación MQTT) en la tabla detecciones de la base de datos. Finalmente, una notificación

detallada, que contenga la URL de la captura de la detección y la ubicación geográfica, que es enviada a Slack

d. Bases de datos

La base de datos tuvo un gestionamiento mediante la utilización de SQLite para lograr garantizar una solución ligera y a la vez eficiente, para el entorno en el que se va a desarrollar la aplicación. Se definieron tres tablas principales: primero la tabla "usuarios" y en ella almacena la información de autenticación de los usuarios con campos como "ID" para la (clave primaria), "usarname" y "password" cifrado para seguridad; la tabla "gps_data" en donde se registran los datos de localización con campos para "ID", "latitude", "longitude" y "timestamp" para el momento exacto de la recepción de la ubicación; y finalmente la tabla "detecciones", la cual está diseñada para el almacenamiento de las alertas de detección con campos "ID", "Fecha" (parámetro temporal de la detección), "Objetos_Detectados" (donde se enlista los objetos identificados en el fotograma), "Ruta_Captura" (donde se especifica la dirección del archivo de imagen capturada) y "Coordenadas" (latitud y longitud del evento). La arquitectura relacional posibilita una gestión organizada y eficiente de los datos, haciendo más fácil la correlación entre los usuarios, ubicaciones y los eventos detectados. Por lo tanto, se soporta la trazabilidad de alertas y se posibilita en un futuro una ampliación para la integración de datos históricos y de las actualizaciones de los modelos de detección. El esquema entidad-relación presenta las relaciones entre los usuarios y las detecciones, por medio de la enfatización de la importancia del vínculo entre la autenticación y el registro de los eventos para garantizar la seguridad y la confiabilidad del sistema de detección delincuencial. Ah, entiendo. Quieres una descripción concisa para un diagrama entidad-relación que refleje la base de datos que has descrito. La Figura 80Figura 80 muestra el diagrama entidad-relación de la base de datos, compuesta alrededor de las tablas usuarios, gps_data y detecciones. Se evidencia la relación existente uno a muchos (1: N) entre los usuarios y las detecciones, así como también entre gps_data y detecciones, permitiendo la correlación de eventos con usuarios y las ubicaciones precisas.

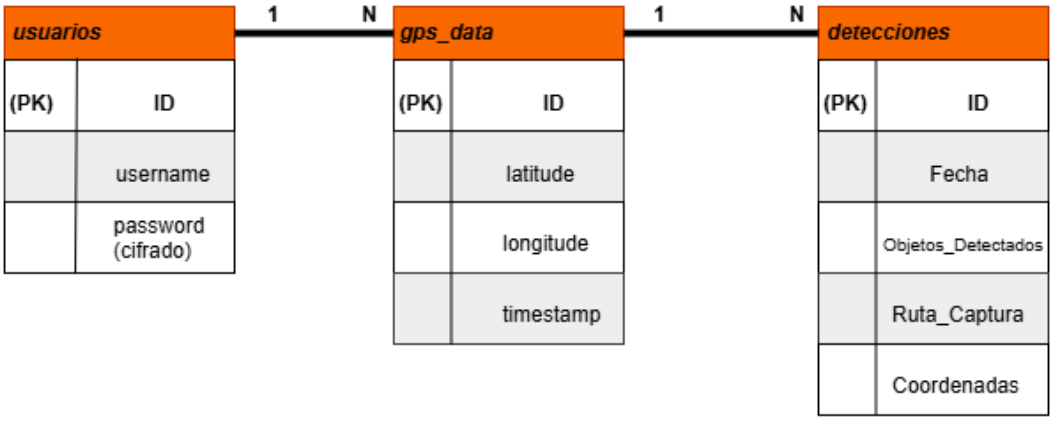


Figura 80. Diagrama entidad-relación de la base de datos.

Las detecciones se manejan por medio de un hilo de procesamiento paralelo, implementado con la librería Thread de Python, mismo que permite ejecutar múltiples tareas que se encuentran de forma concurrente para la optimización del rendimiento del sistema. Este hilo es inicializado al comenzar la captura y el análisis del video en tiempo real, posibilitando que la detección y el registro en la base de datos se lleven a cabo sin interrupciones. La Figura 81 exhibe la función con la que se lleva a cabo los registros de las detecciones y los datos del módulo GPS cuando existe la detección de alguna eventualidad, por otra parte, también se lleva a cabo consultas a la base de datos cuando el usuario lo solicita por medio del archivo detections_history.html.

```python
def save_detection(objects_detected, frame, etiqueta):
    captura_dir = "static/capturas"
    if not os.path.exists(captura_dir):
        os.makedirs(captura_dir)

    timestamp = datetime.now().strftime("%Y%m%d_%H%M%S")
    local_captura_path = f"{captura_dir}/captura_{timestamp}.jpg"
    cv2.imwrite(local_captura_path, frame)

    google_drive_url = upload_to_google_drive(local_captura_path)
    if not google_drive_url:
        print("ADVERTENCIA: No se pudo subir la imagen a Google Drive. La alerta de Slack no incluirá la imagen.")
        captura_to_slack = None
    else:
        captura_to_slack = google_drive_url

    with app.app_context():
        detection = Detection(
            timestamp=datetime.now(),
            etiqueta=etiqueta,
            objetos_detectados=str(list(objects_detected)),
            ruta_captura=local_captura_path,
            latitude=last_latitude,
            longitude=last_longitude
        )
        db.session.add(detection)
        db.session.commit()

        total_detections = Detection.query.count()
        if total_detections > 20:
            detections_to_delete = Detection.query.order_by(Detection.timestamp.asc()).limit(total_detections - 20).all()
            for det_entry in detections_to_delete:
                if det_entry.ruta_captura and os.path.exists(det_entry.ruta_captura):
                    try:
                        os.remove(det_entry.ruta_captura)
                        print(f"Archivo de captura local eliminado: {det_entry.ruta_captura}")
                    except Exception as e:
                        print(f"Error al eliminar el archivo de captura local {det_entry.ruta_captura}: {e}")
                db.session.delete(det_entry)
            db.session.commit()

    send_slack_alert(objects_detected, captura_to_slack, etiqueta, last_latitude, last_longitude, is_resend=False)
```

Figura 81. Función con la que se lleva a cabo los registros de las detecciones y los datos del módulo GPS cuando existe una detección.

e. Frontend (Interfaz de usuario)

Para poder contar con una interfaz de usuario se llevó a cabo la creación de algunos archivos de tipo HTML; todo con el fin de lograr controlar la GUI del sistema o como se lo ha venido denominando durante todo el desarrollo de este proyecto de investigación como “Plataforma de desarrollo para el control de la información”. Al momento de ejecutar el archivo “app.py” y abrir el localhost:5000 en el navegador se mostró el “home.html” donde se obtuvo la interfaz gráfica principal de autenticación del sistema y entra en funcionamiento la base de datos anteriormente mencionada con la tabla que se denomina como “usuarios”. La Figura 82 muestra la interfaz gráfica del sistema donde el usuario procede a introducir sus credenciales y autenticarse con el sistema.



Figura 82. Interfaz gráfica de la autenticación de los usuarios.

Una vez que la autenticación es correcta el sistema redirige a la siguiente página compuesta por el archivo “index.html”; página que se conforma por una interfaz gráfica donde se visualiza los valores del módulo GPS; enviados a la web desde Node-Red por medio del protocolo Mqtt. Se visualiza y se analiza cada una de las métricas de entrenamiento de los dos modelos (el de detección de objetos y el de las posturas corporales) del video que está siendo captado en tiempo real por la cámara conectada a la tarjeta de procesamiento. La Figura 83 muestra el funcionamiento del sistema dentro del archivo “index.html”.

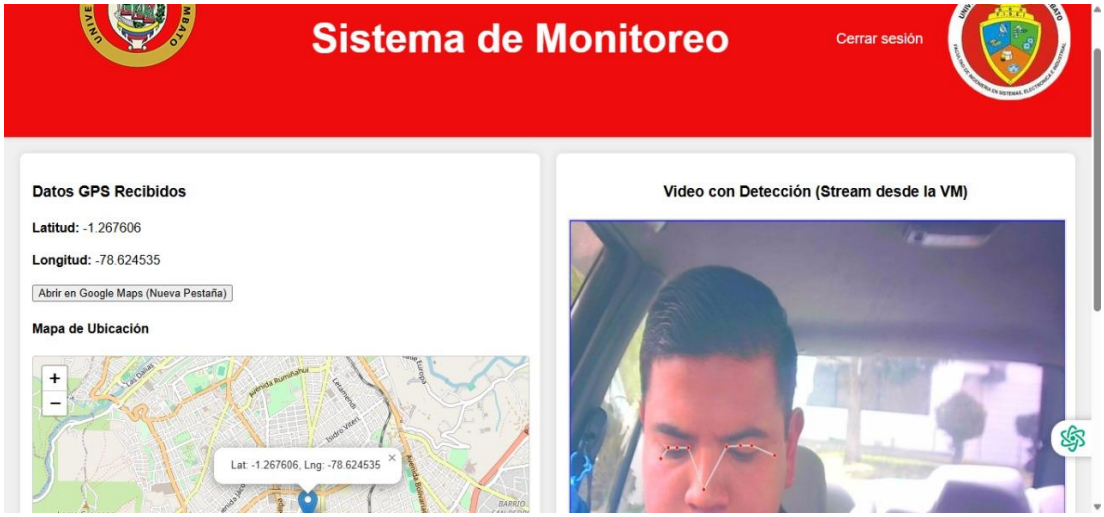


Figura 83. Funcionamiento del sistema dentro del archivo "index.html".

El sistema también cuenta con varias opciones que entre las más destacables está la opción de poder visualizar los últimos 20 registros de los acontecimientos que el sistema detecte una vez que todos sus componentes entren en funcionamiento. En esta etapa entra en funcionamiento las tablas "gps_data" y "Objetos_Detectados" de la base de datos. La Figura 84 muestra el diseño de esta pestaña.



Figura 84. Interfaz gráfica donde se registran las detecciones.

f. Comunicación proactiva

Para tener un control de alertas tempranas de personas sospechosas y delitos flagrantes, se utilizó la plataforma Slack; aprovechando al máximo sus capacidades de integración por medio de Webhooks entrantes.

El proceso seguido para la configuración dentro de Slack implicó los pasos siguientes:

Creación de un Espacio de trabajo y de un canal dedicado. En primera instancia se creó un espacio de trabajo dentro de Slack y ya dentro de este, se hizo la designación de un canal específico para la recepción de las alertas que en este caso el nombre del canal es #seguridad_medios-de-transporte.

Generación del Webhook URL: Dentro del apartado de las configuraciones de las aplicaciones del entorno de trabajo de Slack, se agregó una aplicación de "Webhook Entrante". Aquel proceso procedió a generar una URL única de Webhook (parecida a https://hooks.slack.com/services/T00000000/B00000000/XXXXXXXXXXXXXXXXXXXXXXXXX) que funciona tal cual como un punto final Http al que la aplicación hizo sus envíos de las alertas detectadas. Dicho URL es indispensable para la integración de sistemas, dado que habilita el acceso de cualquier sistema externo que en este caso fue la aplicación Flask, para el envío de mensajes al canal de Slack mencionado anteriormente.

Después que se obtiene la SLACK_WEBHOOK_URL, se lo integró de manera directa dentro del código de Python de la aplicación Flask. La aplicación se encuentra en total capacidad de construir un mensaje JSON que incluye los detalles de la alerta como son los siguientes: la etiqueta de detección, la fecha, hora, los objetos detectados, un enlace con la ubicación del cometimiento del delito y una captura del ilícito. Para luego enviar este JSON mediante una solicitud HTTP POST a la SLACK_WEBHOOK_URL configurada.

Finalmente, la integración de sistemas posibilita que la aplicación haga envíos de mensajes de alerta de manera automática al canal de Slack. Por ejemplo, la figura PONER exhibe un mensaje de alerta anticipada en Slack, detallando la etiqueta de la detección en este caso "Delito flagrante", la hora y fecha precisas, la lista de objetos identificados; en este caso identificó un objeto etiquetado como "armas", la localización a través de un enlace a Google Maps y, sobre todo, una captura del fotograma donde se llevó a cabo la detección, la cual se proporciona directamente desde Google Drive, asegurando el acceso y la persistencia de la imagen. La Figura 85 muestra el funcionamiento del canal de difusión dentro de Slack por medio un mensaje de alerta a partir de una detección.



Figura 85. Alerta difundida por Slack

3.4 Diseño y Montaje de componentes dentro del sistema.

3.4.1 Diseño Estructural del Prototipo del Sistema.

A continuación, se detalla paso a paso la metodología empleada en Fusion 360 para llevar a cabo la creación de una carcasa modular estilo rack. Dicha carcasa se diseñó con la finalidad de albergar componentes críticos de sistemas embebidos, específicamente una unidad de procesamiento NVIDIA Jetson Nano, una placa de Arduino UNO conectada a ella un módulo GPS Neo 6m y finalmente una cámara web para video en tiempo real. La Figura 119 muestra la estructura operativa del sistema mediante un esquema de conexiones de los equipos utilizados. El diseño se enfocó especialmente en la optimización de los recursos para lograr una mejor integración de componentes y la gestión térmica que demanda cada uno de ellos.

a. Plataforma del sistema y ajustes preliminares

La utilización de Autodesk Fusion 360 es fundamental para el proceso de diseño y fabricación, con el aprovechamiento de su potente sistema CAD/CAM/CAE alojado en la nube. Esta plataforma permite desarrollar modelados paramétricos de cuerpos sólidos de manera eficiente, lo cual es crucial para iterar rápidamente en los diseños. Para garantizar la máxima precisión y compatibilidad con los subsiguientes procesos

de impresión 3D y mecanizado CNC, determinación de las unidades de medida en milímetros (mm), una práctica indispensable para la coherencia entre el diseño digital y la producción física. El área de trabajo de Fusion 360 es donde se materializan estas ideas, proporcionando el entorno robusto necesario para el modelado detallado y la preparación para la manufactura.

b. Construcción de la Carcasa principal.

Este proceso de diseño inició con un enfoque fundamental e iterativo, mostrando un progreso notable desde primitivas geométricas básicas hasta un ensamblaje integrado en su totalidad.

Definición del boceto base. Se inició el diseño con un nuevo boceto sobre el plano XY. Luego se utilizó la herramienta Rectángulo para lograr definir la huella principal de la carcasa, donde las dimensiones de este rectángulo se eligieron de manera muy meticulosa con el fin de lograr acomodar adecuadamente el componente más grande (NVIDIA Jetson Nano) dejando los espacios libres suficientes para el flujo de aire y el enrutamiento de los cables con el resto de los componentes. Pues también se aplicó las debidas restricciones geométricas apropiadas (como, por ejemplo: Perpendicular, Paralelo o Igual) para lograr garantizar la estabilidad dimensional y la intensión principal del diseño.

Extrusión de la Estructura Base. Con la herramienta extruir, el perfil del boceto se extruyó de forma perpendicular al plano del boceto para lograr crear el cuerpo sólido inicial de la base de la carcasa. La distancia de extrusión fue determinada por el espacio vertical requerido por la Jetson Nano, incluyendo distintos factores para separadores y conectores que posee la Unidad de procesamiento. La Figura 86 presenta la base de la carcasa ya extruida, bordeada, y con las debidas aberturas para el flujo de aire.

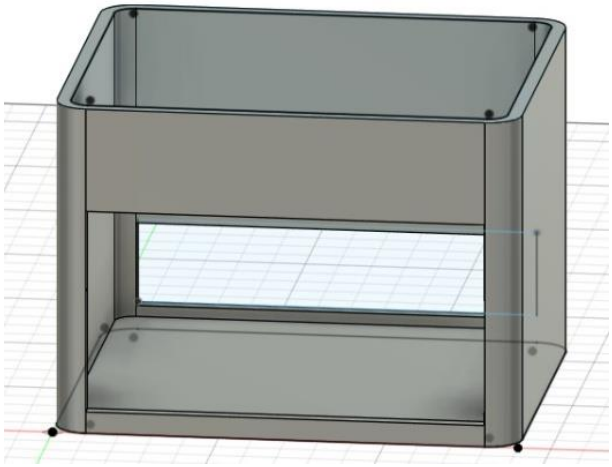


Figura 86. Base de la carcasa ya extruida, bordeada, y con las debidas aberturas para el flujo de aire, dentro del entorno de Fusion 360.

Compartimento Intermedio (Arduino UNO y Módulo GPS). Se creó un nuevo boceto en un plano desplazado sobre la cara superior de la extrusión base. Aquello establece el nivel para el compartimento del Arduino UNO. De nuevo, con la ayuda de la herramienta Rectángulo, se dibujó un perfil rectangular interno con el fin de definir la abertura para el Arduino UNO y el módulo GPS. Aquel perfil fue dimensionado con total precisión para contar con un ajuste perfecto, abriendo la posibilidad de contar con accesorios de montaje. La Figura 87 exhibe el espacio donde se ajustó el microcontrolador Arduino UNO, junto con el módulo GPS.

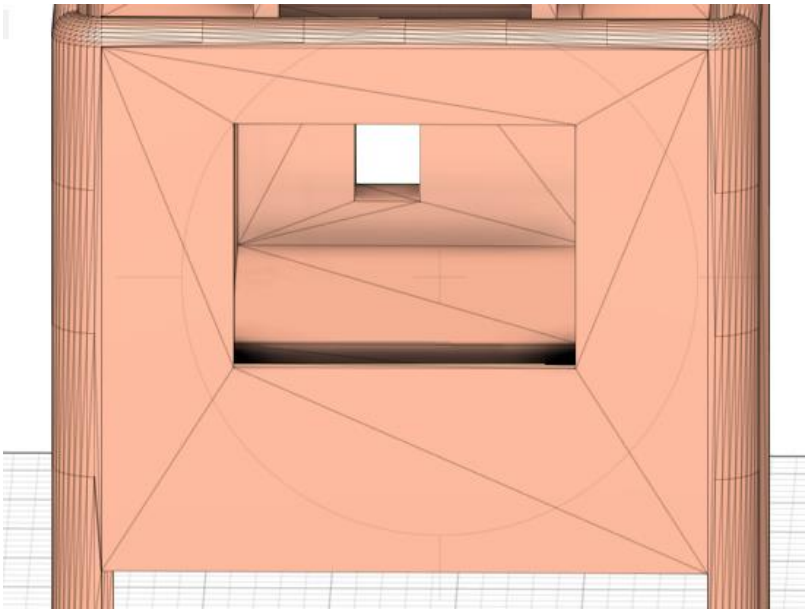


Figura 87. Espacio donde se ajustó el microcontrolador Arduino UNO, junto con el módulo GPS.

Compartimento Superior (Cámara Web). Similar al compartimento realizado en el Arduino, se colocó un nuevo boceto sobre un plano desplazado desde la parte superior del compartimento intermedio. Después se dibujó un perfil rectangular más pequeño, aquel que fue dimensionado con precisión con el fin de alojar la cámara web, se tuvo en cuenta la posición del lente de captación de la cámara y la salida del cable. La Figura 88 muestra el compartimento dedicado para la cámara web.

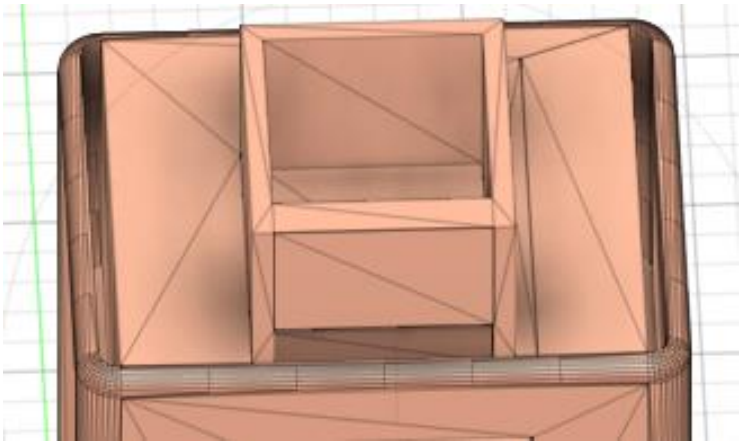


Figura 88. Compartimento dedicado para la cámara web

De esta forma, el proceso de diseño iterativo y la aplicación rigurosa de cada una de las herramientas de modelado desarrollado por medio de Fusion 360, se obtuvo un diseño final de la carcasa. La misma presenta una configuración bastante optimizada en forma de un rack o armario de telecomunicaciones, estratégicamente diseñada para lograr la integración eficiente de la Unidad de Procesamiento (NVIDIA Jetson), el microcontrolador Arduino UNO, con el módulo GPS y la cámara web. La Figura 89 evidencia el diseño estructural final de la carcasa.

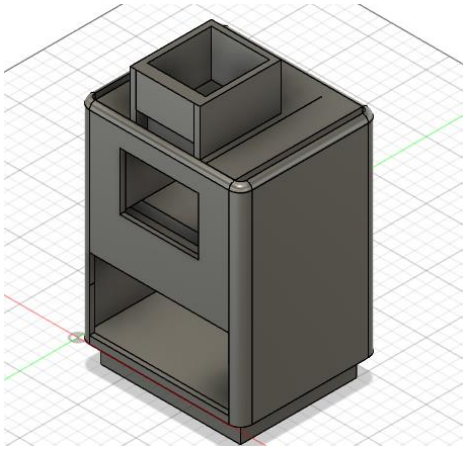


Figura 89. Diseño estructural final de la carcasa.

3.4.2 Montaje de los componentes sobre la carcasa impresa en 3D.

Se llevó a cabo la impresión del diseño de la carcasa del prototipo en 3D. Donde se pudo evidenciar a breves rasgos que la impresión se la ejecutó de manera muy exitosa; sin alteraciones dentro de sus dimensiones físicas, es decir; las medidas que se colocaron dentro del software Fusion 360, se conservaron en la impresión de la carcasa física. La Figura 90 exhibe la carcasa recién impresa, lista para el montaje de cada uno de los componentes mencionados.



Figura 90. Carcasa recién impresa, lista para el montaje de cada uno de los componentes mencionados.

Una vez ya impresa y revisada cada una de las dimensiones de la carcasa física, se procedió a montar cada uno de los componentes dentro de la misma. La parte inferior de la estructura se colocó la unidad de procesamiento NVIDIA Jetson Nano, más arriba se encuentra en cambio el sistema embebido; conformado por la placa microcontroladora Arduino UNO y el módulo GPS, en la parte exterior del prototipo, junto con la cámara se colocó la antena del módulo, con la finalidad de captar de forma eficiente la señal satelital del GPS, y finalmente en la parte superior de la estructura se ubicó la cámara webcam. La Figura 91 muestra cada uno de los componentes

recientemente mencionados; ya colocados dentro de la estructura, formando así el sistema de detección delincuencial.



Figura 91. Componentes ya colocados dentro de la estructura 3D impresa.

En la parte superior de la carcasa, se ubica el compartimento dedicado para la webcam HD (720p) de Sinovisión. Espacio adecuado y estratégico para la colocación de este dispositivo y para el despliegue del cable principal de la cámara de tipo USB que se conecta a uno de los puertos de la NVIDIA Jetson Orin Nano a lo largo de la estructura principal. La Figura 92 muestra el compartimento dedicado para la webcam HD (720p).



Figura 92. Compartimento dedicado para la webcam HD (720p).

En la parte intermedia de la estructura, se ubica el espacio establecido para la placa microcontroladora Arduino UNO y el módulo GPS Neo 6m. Ambos componentes se conectan entre sí por medio de una placa PCB; esto con el fin de construir un sistema embebido que logre obtener la adquisición de los datos de geoposicionamiento. La placa Arduino y el módulo GPS poseen una conexión con base de una comunicación Serial, es decir; el Arduino UNO alimenta al módulo GPS con los pines de VCC y GND. Y para la parte de los datos la placa Arduino usa el pin número 4 que funciona como Receptor o Rx, conectado al pin de Transmisión o Tx del Módulo GPS. En la parte externa de la estructura se ubica la antena del Módulo GPS. El cable de alimentación del Arduino recorre por la parte interna de la estructura hacia la base donde se ubica la NVIDIA Jetson Orin Nano donde se conecta a ésta por medio de uno de sus puertos USB, y donde la placa Arduino es identificada como /dev/ttyACM0 por la Unidad de Procesamiento. Debido a que el diseño de la estructura fue inspirado en un rack o armario de Telecomunicaciones. La Figura 93 muestra el sistema embebido colocado en su respectivo compartimento dentro de la estructura impresa.

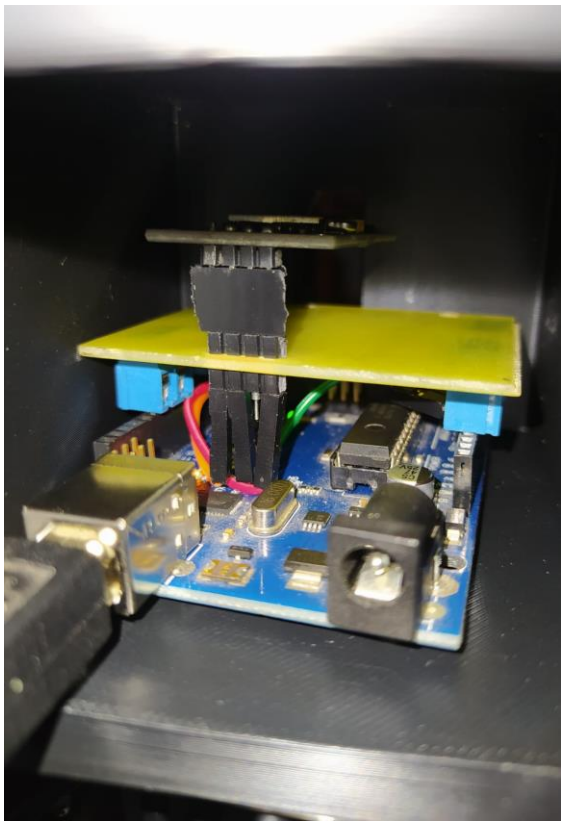


Figura 93. Sistema embebido colocado en su respectivo compartimento.

La Figura 94 exhibe la ubicación de la antena del módulo GPS en la parte externa de la estructura.

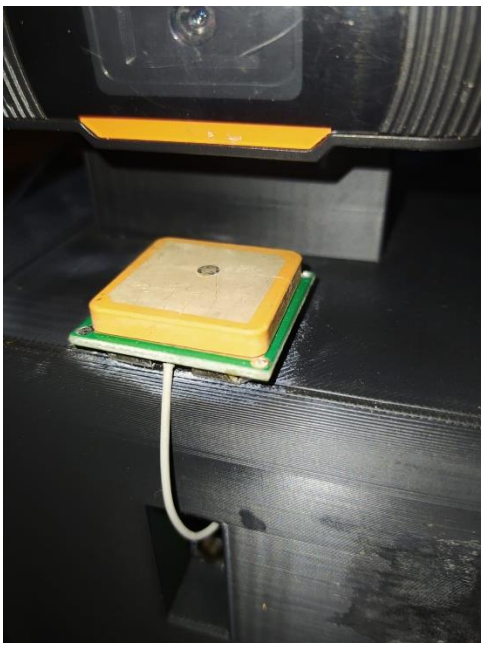


Figura 94. Antena del módulo GPS colocado en la parte externa de la estructura impresa.

En la base de la estructura se ubica la Unidad de procesamiento NVIDIA Jetson Orin Nano; aquella es el cimiento de todo el sistema. A ésta se conectan por medio de los puertos USB todos los dispositivos que componen al sistema; la cámara y la placa microcontroladora. La Figura 95 presenta la Unidad de procesamiento NVIDIA Jetson Orin Nano colocada en la base de la estructura.

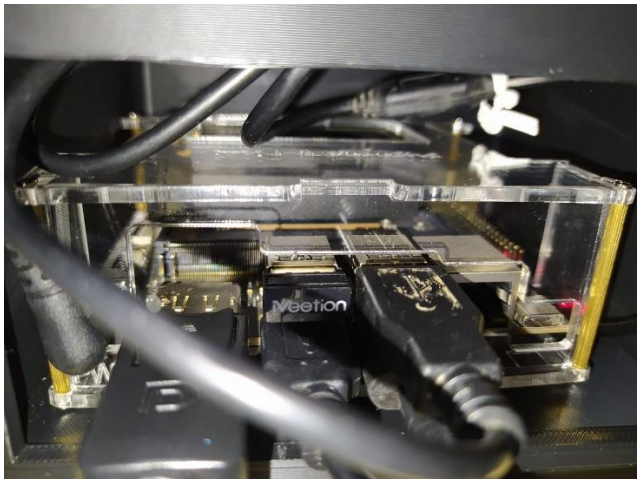


Figura 95. Unidad de procesamiento NVIDIA Jetson Orin Nano colocada en la base de la estructura.

Finalmente, una vez ya montado todo el sistema, la Figura 96 muestra una conexión al sistema de monitoreo por medio de una pantalla LCD.

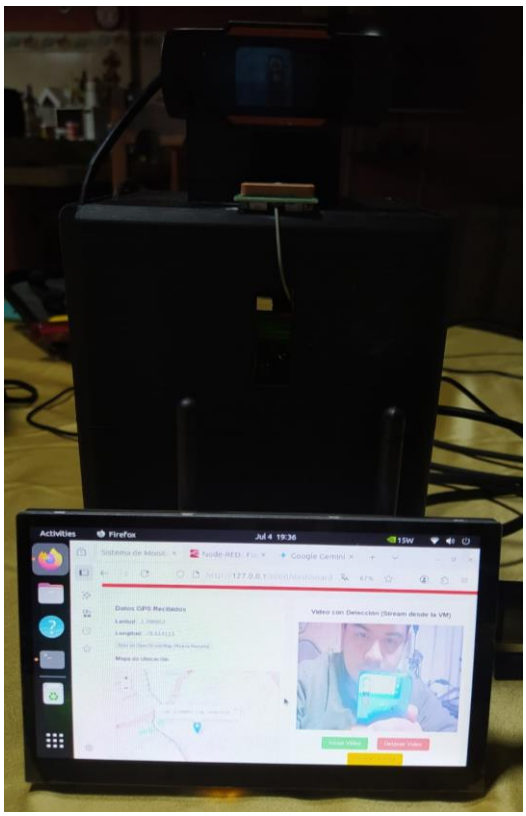


Figura 96. Conexión al sistema de monitoreo por medio de una pantalla LCD.

3.5 Verificación de funcionamiento del sistema

Para lograr una verificación exitosa de funcionamiento del sistema, se llevó a cabo una variedad de recreaciones escenificadas de incidentes criminales, centrándose en situaciones de asalto mediante armas de fuego y cuchillos. También se incluyeron escenas que reflejan comportamientos normales dentro de varios medios de transporte, con involucrados que se encontraban sentados o de pie ya sea dentro de los medios de transporte y en espacios amplios. Esto permitió evaluar el sistema considerando las variaciones tanto en las posturas corporales según la posición adoptada por los individuos como en los escenarios donde se llevaron a cabo las pruebas, con el fin de validar el funcionamiento del sistema. A continuación, se presenta a detalle las pruebas que se llevaron a cabo.

3.5.1 Evaluaciones en condiciones reales del sistema de detección de objetos y posturas corporales.

Se realizaron pruebas de las cuales las se relacionaron con situaciones de actos delictivos entre asaltos con cuchillo y armas de fuego, junto con la utilización de gorras. Se comprobó la habilidad del sistema a través de estas pruebas realizadas para identificar diversos tipos de sucesos; en función de los objetos que posea y de las posturas corporales que mostraron los participantes en la prueba.

La Figura 97 exhibe la alerta temprana de una de las pruebas efectuadas. Se visualiza que se detectó de forma correcta un arma y la etiqueta “Delito flagrante” es la evidencia clave que una postura corporal amenazante fue detectada dentro del sistema. Además, en el sistema de mensajería se muestra tanto la fecha y hora en que se realizó la detección como el listado de los objetos identificados.



Figura 97. Alerta temprana de una de las pruebas efectuadas.

La Figura 98 muestra otro de los sucesos dramatizados, en donde se evidencia la notificación de difusión de la detección de las etiquetas “gorras” y “cuchillo”, y al igual que la detección anterior se envía la etiqueta final como un “Delito flagrante”, la

fecha y la hora de la detección y por último los objetos detectados que en este caso es un cuchillo y una gorra que posee el involucrado.



Figura 98. Delito flagrante con gorra y cuchillo.

La Figura 99 muestra otro evento, en donde se evidencia la notificación de difusión de la detección de las etiquetas “gorras” y “armas”, y al igual que la detección anterior se envía la etiqueta final como un “Delito flagrante”, la diferencia en este caso es que la víctima tiene puesta una gorra y el sistema la detectó, y el agresor contaba con el arma en su mano y el sistema también lo detectó. Aunque no la misma persona agresora contó con ambos objetos, esto facilitó la detección y difusión temprana del ilícito. Al ingresar en el enlace de la ubicación del ilícito, el enlace redirecciona al estadio de San Luis de Picaihua, en la ciudad de Ambato, Ecuador. Lugar donde se realizó las pruebas de cobertura del GPS. La Figura 100 evidencia la ubicación que se acaba de mencionar dentro de Google Maps.

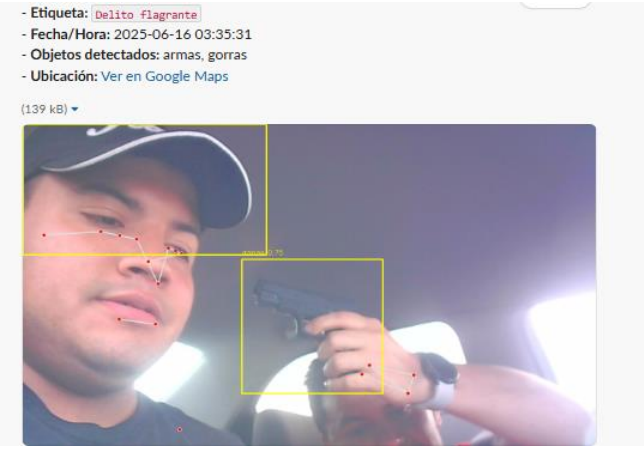


Figura 99. Delito flagrante con gorra y arma.



Figura 100. Ubicación marcada en Google Maps del delito flagrante.

La Figura 101 exhibe el desempeño del sistema en otro entorno de prueba, se evidencia que la detección es correcta del arma que se encuentra en poder del involucrado.

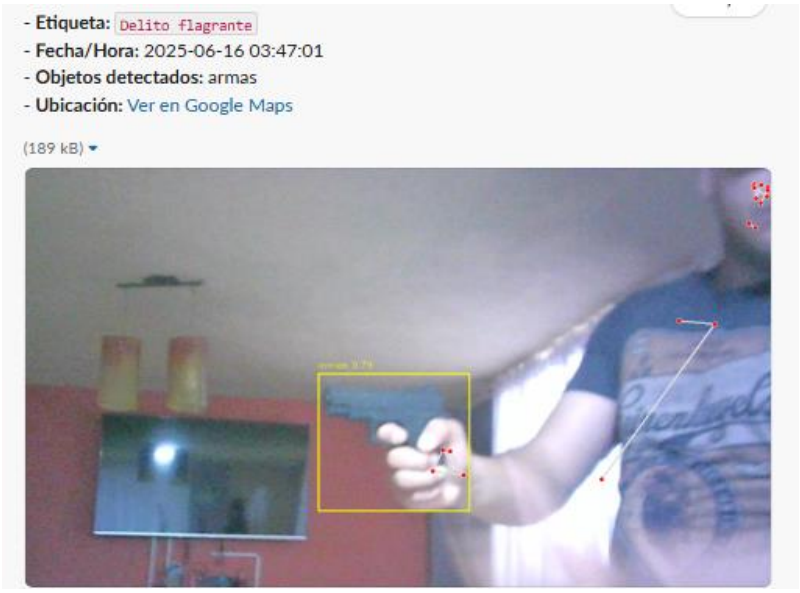


Figura 101. Delito flagrante con arma.

La Figura 102 muestra el desempeño del sistema, ahora con una persona y en un entorno diferente, en esta notificación de alerta no aparece la ubicación del ilícito debido a que no se encontraba conectado el módulo GPS en el momento de la prueba, pero se evidencia que la detección es la correcta dado que el involucrado si tiene en su poder un cuchillo.



Figura 102. Delito flagrante con cuchillo.

3.5.2 Evaluaciones en condiciones reales del sistema GPS.

Se realizó pruebas de cobertura del módulo GPS dentro de un vehículo, clasificadas en dos rutas diferentes, en donde este módulo entró en funcionamiento y dentro de la interfaz gráfica del usuario existe la opción que posibilita el registro, el almacenamiento y la descarga de un archivo .txt de estos datos de geoposicionamiento proporcionado por el módulo GPS.

Vale mencionar que una vez que se obtuvo cada uno de los archivos .txt por medio de la opción “descargar”; existente dentro de la interfaz gráfica. Se utilizó un Script de Python para extraer la latitud y longitud de cada línea que consta dentro del archivo .txt, luego crea un mapa interactivo usando la librería Folium, posteriormente dibuja una línea o un camino que conecta todos los puntos GPS en orden. También añade marcadores para el inicio/fin o puntos específicos y finalmente, guarda el mapa como un archivo HTML que puede ser abierto y visualizado en cualquier navegador. La Figura 103 exhibe la función que crea el tracking a partir del archivo .txt.

```python
def create_gps_tracking_map(input_file, output_html_file="gps_tracking_map.html"):
    """
    Lee datos GPS de un archivo de texto con formato '[YYYY-MM-DD HH:MM:SS] Lat: <lat>, Lon: <lon>',
    y crea un mapa interactivo con la ruta.

    Args:
        input_file (str): La ruta al archivo .txt que contiene los datos GPS.
        output_html_file (str): El nombre del archivo HTML de salida para el mapa.
    """
    coordinates = []

    # Expresión regular para extraer latitud y longitud
    gps_pattern = re.compile(r"Lat:\s*([-\d.]+),\s*Lon:\s*([-\d.]+)")

    try:
        with open(input_file, 'r') as f:
            for line in f:
                match = gps_pattern.search(line)
                if match:
                    try:
                        lat = float(match.group(1).strip())
                        lon = float(match.group(2).strip())
                        coordinates.append([lat, lon])
                    except ValueError:
                        print(f"Advertencia: No se pudo convertir Lat/Lon a número en la línea: '{line.strip()}'")
                else:
                    print(f"Advertencia: Línea con formato GPS inesperado (saltando): '{line.strip()}'")
    except FileNotFoundError:
        print(f"Error: El archivo '{input_file}' no fue encontrado.")
        return
    except Exception as e:
        print(f"Error al leer el archivo: {e}")
        return
```

Figura 103. Función que crea el tracking a partir del archivo .txt.

a. Ruta de prueba 1: Estadio de tierra de San Luis de Picaihua-Parque Principal de Salasaca

Esta prueba se la llevó a cabo el sábado 14 de junio del año 2025, aproximadamente a las 15:45 PM, la ruta empieza el estadio de San Luis de Picaihua y culmina en el parque principal de Salasaca. La Figura 104 evidencia mediante el ícono de PLAY que es aquí donde empieza el recorrido de esta ruta de prueba.

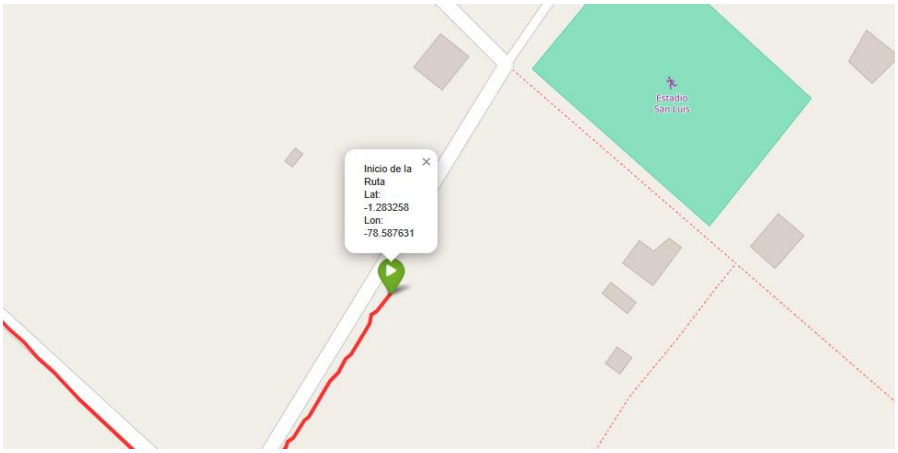


Figura 104. Inicio de la Ruta 1.

A pesar de las difíciles condiciones provocadas por la densa vegetación a lo largo de la ruta, el viaje de vuelta del celo se completó con éxito. Cabe destacar que en ningún

momento se produjo una pérdida de recepción de datos GPS, lo que garantizó la integridad y continuidad del rastreo a lo largo de todo el recorrido. Durante el trayecto, la ruta pasó muy cerca de los límites de la parroquia de Tangaiche y se acercó considerablemente. Finalmente, el viaje llegó a su fin en la parroquia Salasaca, lo que marcó el final del esperado recorrido. La Figura 105 muestra el trayecto total de la ruta de prueba 1.



Figura 105. Trayecto total de la ruta de prueba 1.

b. Ruta de prueba 2: Parque Principal de Salasaca-Barrio El Porvenir en Ambato.

Esta prueba se la llevó a cabo el sábado 14 de junio del año 2025, aproximadamente a las 18:30 PM, la ruta empieza en el parque principal de Salasaca y termina en el Barrio El Porvenir, en Ambato. La Figura 106 evidencia mediante el ícono de PLAY que es aquí donde empieza el recorrido de esta ruta de prueba.

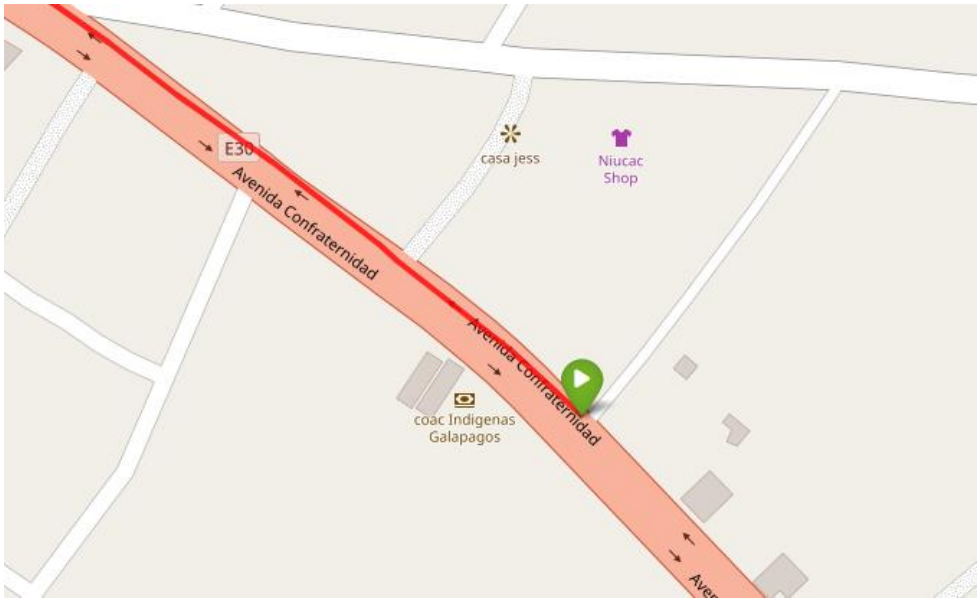


Figura 106. Inicio de la Ruta 1

El recorrido de la ruta se inició con total éxito desde la parroquia Salasaca, un punto de inicio que marcó el comienzo de una trayectoria variada y a la vez desafiante. A lo largo del trayecto, se enfrentaron condiciones complejas, alternando entre zonas con densa vegetación y áreas con regular población, lo que puso a prueba la capacidad de rendimiento del sistema de monitoreo. A pesar de haber aplicado estas variaciones dentro del entorno, es crucial mencionar que el módulo GPS funcionó de manera muy eficiente en todo momento, manteniendo la recepción de los datos de forma constante y fiable, sin ninguna interrupción ni pérdida de señal.

Durante el progreso de la ruta, el trayecto discurrió por puntos de referencia bastante claves como es el caso del ex redondel de la Coca Cola, y atravesó la parroquia Totoras, este tránsito por distintos tipos de entornos demostró la gran robustez del módulo GPS, finalmente, el seguimiento terminó de forma exitosa en el Barrio El Porvenir, consolidando un recorrido completo y exitoso en su registro. La Figura 107 evidencia el trayecto total de la ruta de prueba 2.



Figura 107. Trayecto total de la ruta de prueba 2.

3.5.3 Pruebas de funcionamiento con el sistema de detección y monitoreo completo.

La comprobación del correcto funcionamiento del sistema se complementó con la ejecución de diversas pruebas dentro de entornos reales y con el sistema de detección y monitoreo completo; es decir funcionando con la interfaz gráfica de usuario final, con la ventana de captación de video en tiempo real, recepción de los datos por MQTT, el refresco en tiempo real del mapa en base a la partícula de estudio y finalmente el envío de las alertas a la aplicación de Slack.

La Figura 108 representa la captación de una escena que se ubica al lado derecho de la pantalla donde el individuo tiene en su poder un cuchillo, en la parte superior izquierda se visualiza la recepción de los datos de geoposicionamiento y finalmente en la parte inferior izquierda se evidencia el punto de referencia correspondiente a donde está ocurriendo el ilícito.

La Figura 109 se puede apreciar la difusión del mensaje de alerta del cometimiento del ilícito anterior por medio de la aplicación Alerta, con las especificaciones cruciales del evento, como son la fecha, la hora, los objetos detectados y la ubicación del cometimiento.

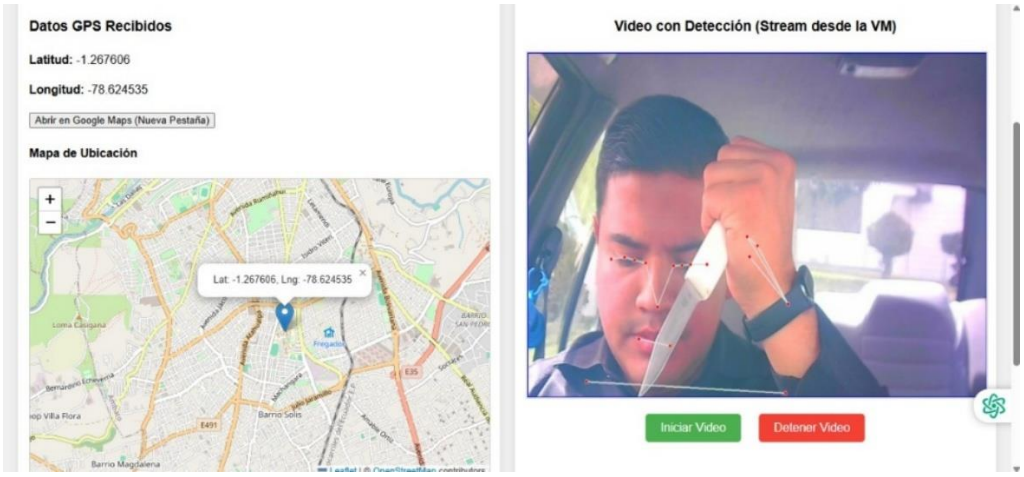


Figura 108. Interfaz gráfica del sistema de monitoreo y detección delincuencial.



Figura 109. Difusión de la alerta de detección de la prueba funcionamiento.

La Figura 110 se exhibe la difusión de alerta de otra prueba de funcionamiento, donde en este caso, sistema detectó la presencia de un arma de fuego.

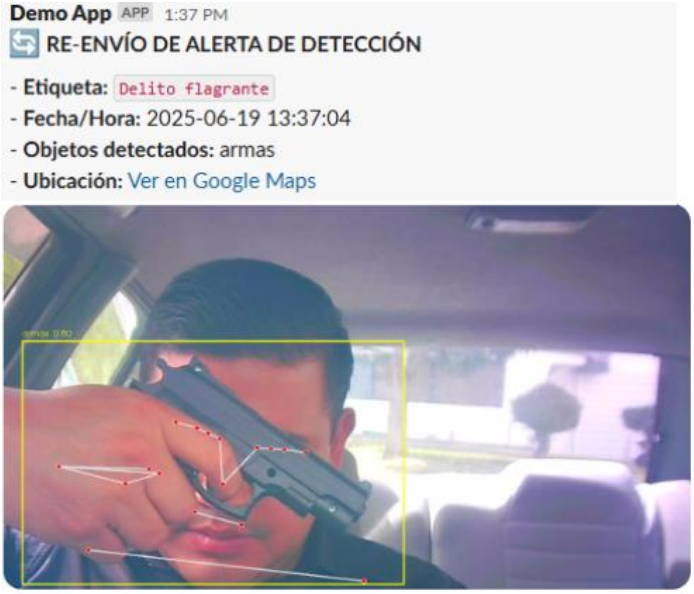


Figura 110. Difusión de la alerta de detección de un arma de fuego.

La Figura 111 muestra la difusión de alerta de otra prueba de funcionamiento, donde en este caso, el sistema detectó la presencia de un cuchillo y de una gorra, dentro del cometimiento del ilícito.

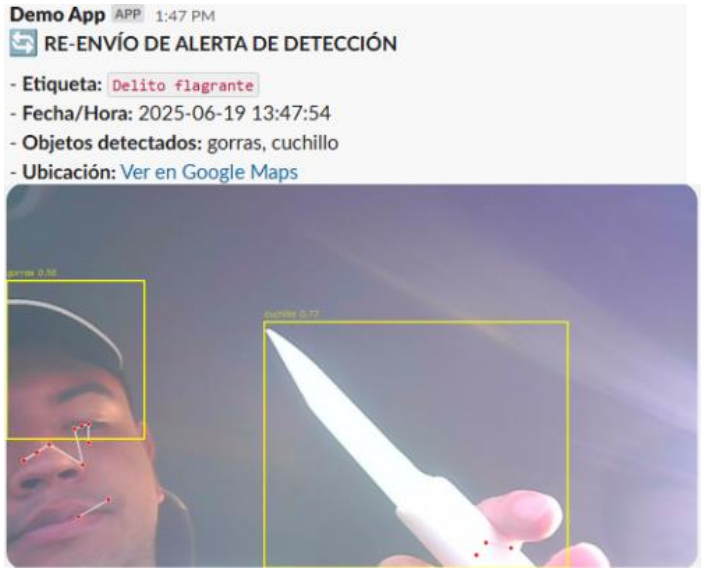


Figura 111. Alerta de otra prueba de funcionamiento del sistema

La Figura 112 muestra la difusión de alerta de otra prueba de funcionamiento, donde en este caso, el sistema detectó la presencia de un arma de fuego dentro de otro entorno real de prueba.



Figura 112. Difusión de la alerta de detección de un arma de fuego dentro de otro entorno.

3.5.4 Parámetros de validación.

a. Matriz de confusión de armas blancas

La matriz de confusión de las armas blancas se realizó por medio del análisis de resultados a 100 pruebas, misma que se resume por medio de la Tabla 20.

Tabla 20. Matriz de confusión de armas blancas

Realidad/Predicción	**Positivo (Detectado)**	**Negativo (No Detectado)**	**Total Realidad**
Objeto Presente	35 (TP)	2 (FN)	37
Objeto Ausente	3 (FP)	60 (TN)	63
Total Predicción	38	62	100

Análisis de Eficiencia: El sistema demostró una alta capacidad para la identificación de armas blancas cuando presentes (35 TP), con muy pocos fallos (2 FN). La tasa de falsas alarmas es muy baja (3 FP), mostrando una buena precisión.

b. Matriz de confusión de armas de fuego.

La matriz de confusión de las armas de fuego se realizó por medio del análisis de resultados a 100 pruebas, misma que se resume por medio de la Tabla 21.

Tabla 21. Matriz de confusión de armas de fuego.

Realidad/Predicción	Positivo (Detectado)	Negativo (No Detectado)	Total Realidad
Objeto Presente	28 (TP)	1 (FN)	29
Objeto Ausente	2 (FP)	69 (TN)	71
Total Predicción	30	70	100

Análisis de Eficiencia: Se visualiza un rendimiento muy alto dentro de la detección de armas de fuego, con casi todas las instancias reales siendo identificadas (28 TP) y un mínimo de errores críticos (1FN). Los Falsos positivos son muy reducidos (2 FP).

c. Matriz de confusión de la detección de gorras

La matriz de confusión de la detección de gorras se realizó por medio del análisis de resultados a 100 pruebas, misma que se resume por medio de la Tabla 22.

Tabla 22. Matriz de confusión de la detección de gorras

Realidad/Predicción	Positivo (Detectado)	Negativo (No Detectado)	Total Realidad
Objeto Presente	50 (TP)	5 (FN)	55
Objeto Ausente	10 (FP)	35 (TN)	45
Total Predicción	60	40	100

Análisis de Eficiencia: La identificación de gorras presenta una alta capacidad (50 TP), aunque presenta un índice un poco más elevado de falsos negativos (5 FN) y falsos positivos (10 FP) en relación con las armas, lo que podría ser aceptable considerando la naturaleza del objeto.

d. Matriz de confusión de la Detección de Posturas Sospechosas.

La matriz de confusión de la detección de Posturas Sospechosas se realizó por medio del análisis de resultados a 100 pruebas, misma que se resume por medio de la Tabla 23.

Tabla 23. Matriz de confusión de la Detección de Posturas Sospechosas.

Realidad/Predicción	Positivo (Detectado)	Negativo (No Detectado)	Total Realidad
Objeto Presente	20 (TP)	3 (FN)	23
Objeto Ausente	5 (FP)	72 (TN)	77
Total Predicción	25	75	100

Análisis de Eficiencia: En caso de comportamientos sospechosos, el sistema consigue una identificación eficaz (20 TP) con un número reducido de falsos negativos (3 FN), esencial para la protección. Los falsos positivos (5 FP) son controlables, lo que señala una adecuada identificación de las conductas.

3.5.5 Validación de los resultados

a. Cálculos de detección de armas de blancas.

Valores de la Tabla:

- Verdaderos Positivos (TP): 35
- Falsos Positivos (FP): 3
- Falsos Negativos (FN): 2
- Verdaderos Negativos (TN): 60

Cálculos:

- **Precisión**

$$Precisión = \frac{TP}{TP + FP} \quad (13)$$

$$Precisión = \frac{35}{35 + 3}$$

$$Precisión = 0.921$$

- **Sensibilidad (Recall)**

$$Recall = \frac{TP}{TP + FN} \quad (14)$$

$$Recall = \frac{35}{35 + 2}$$

$$Recall = 0.946$$

- **F1-Score**

$$F1_{score} = 2 * \frac{Precisión * Recall}{Precisión + Recall} \quad (15)$$

$$F1_{score} = 2 * \frac{0.921 * 0.946}{0.921 + 0.946}$$
$$F1_{score} = 0.933$$

- **Exactitud (Accuracy)**

$$Exactitud = \frac{TP + TN}{TP + FP + TN + FN} \quad (16)$$

$$Exactitud = \frac{35 + 60}{35 + 3 + 60 + 2}$$
$$Exactitud = 0.950$$

b. Cálculos de la detección de armas de fuego.

Valores de la Tabla:

- Verdaderos Positivos (TP): 28
- Falsos Positivos (FP): 2
- Falsos Negativos (FN): 1
- Verdaderos Negativos (TN): 69

Cálculos:

- **Precisión**

$$Precisión = \frac{TP}{TP + FP}$$

$$Precisión = \frac{28}{28 + 2}$$

$$Precisión = 0.933$$

- **Sensibilidad (Recall)**

$$Recall = \frac{TP}{TP + FN}$$
$$Recall = \frac{28}{28 + 1}$$
$$Recall = 0.966$$

- **F1-Score**

$$F1_{score} = 2 * \frac{Precisión * Recall}{Precisión + Recall}$$

$$F1_{score} = 2 * \frac{0.933 * 0.966}{0.933 + 0.966}$$
$$F1_{score} = 0.949$$

- **Exactitud (Accuracy)**

$$Exactitud = \frac{TP + TN}{TP + FP + TN + FN}$$
$$Exactitud = \frac{28 + 69}{28 + 2 + 69 + 1}$$
$$Exactitud = 0.970$$

c. Cálculos de la detección de gorras.

Valores de la Tabla:

- Verdaderos Positivos (TP): 50
- Falsos Positivos (FP): 10
- Falsos Negativos (FN): 5
- Verdaderos Negativos (TN): 35

Cálculos:

- **Precisión**

$$Precisión = \frac{TP}{TP + FP}$$
$$Precisión = \frac{50}{50 + 10}$$
$$Precisión = 0.833$$

- **Sensibilidad (Recall)**

$$Recall = \frac{TP}{TP + FN}$$
$$Recall = \frac{50}{50 + 5}$$
$$Recall = 0.909$$

- **F1-Score**

$$F1_{score} = 2 * \frac{Precisión * Recall}{Precisión + Recall}$$
$$F1_{score} = 2 * \frac{0.833 * 0.909}{0.833 + 0.909}$$
$$F1_{score} = 0.869$$

- **Exactitud (Accuracy)**

$$Exactitud = \frac{TP + TN}{TP + FP + TN + FN}$$
$$Exactitud = \frac{50 + 35}{50 + 10 + 35 + 5}$$
$$Exactitud = 0.850$$

d. Detección de Posturas Sospechosas

Valores de la Tabla:

- Verdaderos Positivos (TP): 20
- Falsos Positivos (FP): 5
- Falsos Negativos (FN): 3
- Verdaderos Negativos (TN): 72

Cálculos:

- **Precisión**

$$Precisión = \frac{TP}{TP + FP}$$
$$Precisión = \frac{20}{20 + 5}$$
$$Precisión = 0.800$$

- **Sensibilidad (Recall)**

$$Recall = \frac{TP}{TP + FN}$$
$$Recall = \frac{20}{20 + 3}$$
$$Recall = 0.870$$

- **F1-Score**

$$F1_{score} = 2 * \frac{Precisión * Recall}{Precisión + Recall}$$
$$F1_{score} = 2 * \frac{0.800 * 0.870}{0.800 + 0.870}$$
$$F1_{score} = 0.833$$

- **Exactitud (Accuracy)**

$$Exactitud = \frac{TP + TN}{TP + FP + TN + FN}$$
$$Exactitud = \frac{20 + 72}{20 + 5 + 72 + 2}$$
$$Exactitud = 0.920$$

A continuación, se muestra la Tabla 24. Matriz de confusión, que proporciona los resultados obtenidos para la precisión, la sensibilidad (Recall), F1-Score y la Exactitud de cada una de las categorías de detección establecidas dentro del sistema.

Tabla 24. Parámetros de validación de cada una de las clases .

Parámetros / Categoría	**Armas Blancas**	**Armas de Fuego**	**Gorras**	**Posturas Sospechosas**
Precisión	0.921	0.933	0.833	0.800
Sensibilidad (Recall)	0.946	0.966	0.909	0.870
F1-Score	0.933	0.949	0.869	0.833
Exactitud	0.950	0.970	0.850	0.920

Para lograr obtener una métrica global del rendimiento del sistema, se utilizó el promedio de los F1-Scores de cada una de las categorías. El F1-Score es una métrica armónica que hace un equilibrio de la precisión y la sensibilidad, siendo una de las más apropiadas para evaluar sistemas de detección y clasificación.

Promedio de F1-Scores:

Se suma los F1-Scores de cada categoría y dividimos por el número de categorías.

$$F1_{Promedio} = \frac{F1_{Armas\ B.} + F1_{Armas\ f.} + F1_{Gorras} + F1_{Post.Sos.}}{4} \quad (17)$$

$$F1_{Promedio} = \frac{0.933 + 0.949 + 0.869 + 0.833}{4}$$

$$F1_{Promedio} = 0.896$$

Cálculo del Rendimiento del Sistema:

Finalmente, se multiplica el F1-Score promedio por 100 para expresarlo como un porcentaje:

$$Rendimiento_{sistema} = F1_{Promedio} * 100 \quad (18)$$

$$Rendimiento_{sistema} = 0.896 * 100$$

$$Rendimiento_{sistema} = 89.6\ \%$$

3.6 Despliegue

Después de evaluar el sistema de detección delictiva aplicado a medios de transporte usando visión artificial, se desarrolló un sistema para gestionar la información y procesar las predicciones del enfoque multimodal, aquello se lo desarrolló mediante el framework Flask. Este entorno posibilita gestionar de forma efectiva el servidor o backend system, mismo que se encarga del procesamiento del video en tiempo real, de los datos de geoposicionamiento y el frontend, mismo que corresponde a la interfaz gráfica de usuario. Vale mencionar también que Flask es un framework muy descomplicado que se adapta muy bien a las necesidades del usuario, que las notificaciones de alerta se difunden por medio de Slack; aplicación bastante escalable y muy fácil de compartir con el resto de los usuarios que van a estar involucrados con el sistema de monitoreo. La Figura 113 muestra la recepción de la notificación de alerta por parte de un SmartWatch y también cómo un nuevo usuario se ha sumado al canal de difusión y ahora forma parte de la comunidad del sistema de monitoreo dentro de Slack.



Figura 113. Funcionamiento del sistemas e integración con la aplicación de difusión Slack.

3.6.1 Implementación del prototipo

El sistema de Monitoreo Delincuencial, diseñado específicamente para su aplicación en medio de transporte, fue integrado de forma muy estratégica en un automóvil modelo Kia Soluto. Esta elección se basó específicamente en la necesidad de contar con un prototipo 100% funcional y a la vez discreto. Con la finalidad de asegurar que la presencia del sistema no incomode a ningún pasajero ni afecte la estética interior del vehículo, se determinó la ubicación óptima para su instalación sería en la parte trasera de la gaveta del vehículo. La Figura 114 muestra la ubicación donde se colocó el sistema.

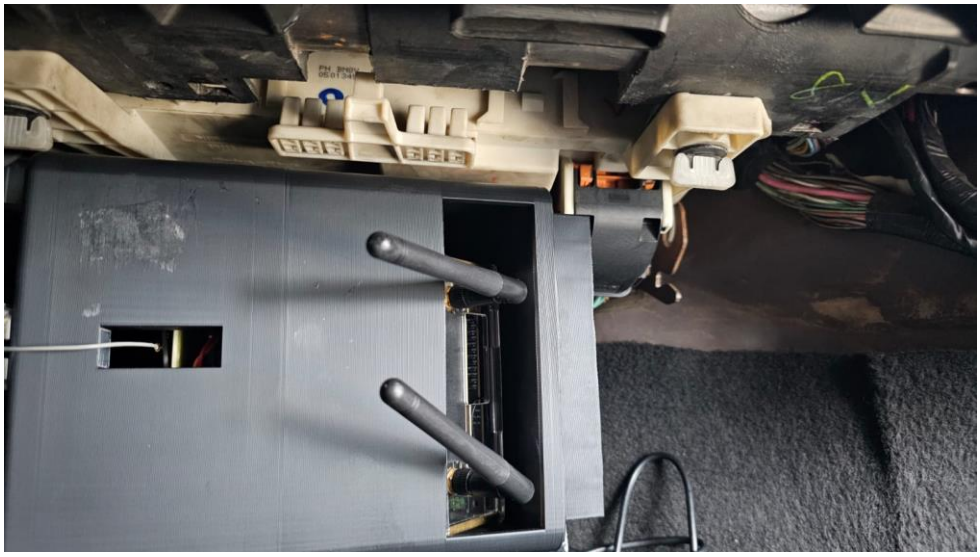


Figura 114. Colocación del prototipo en la parte trasera de la gaveta del vehículo en el que se implementó el sistema

En esta posición, la mayor parte del prototipo queda oculta, garantizando de cierta forma una implementación limpia y no intrusiva dentro del vehículo. El único componente que permanece expuesto es la cámara, dicha visibilidad es muy importante para el cumplimiento de su función principal que es captar en tiempo real los eventos y comportamientos que suceden dentro del habitáculo. Aquella disposición permite que la cámara cuente con una visión clara y sin ninguna obstrucción del entorno de monitoreo. La Figura 115 muestra la colocación de la cámara en una esquina del parabrisas del vehículo.



Figura 115. Colocación de la webcam en la esquina izquierda del parabrisas del vehículo.

En cuanto a su alimentación, el sistema fue conectado directamente a una salida de 12V disponible en el propio vehículo. Dicha conexión asegura un suministro de energía constante y estable, logrando un funcionamiento ininterrumpido del sistema durante los desplazamientos del medio de transporte. La Figura 116 muestra la salida de 12V para la alimentación de la Unidad de Procesamiento Nvidia Jetson Nano.



Figura 116. Salida de 12V para la alimentación de la Unidad de Procesamiento Nvidia Jetson Nano.

Una vez ya concluida la instalación, este vehículo fue utilizado como plataforma para llevar a cabo distintas pruebas de funcionamiento, dichos resultados y el análisis han

sido expuestos de forma muy detallada en secciones anteriores. Durante estas pruebas, se logró verificar la operatividad del sistema, y se logró evidenciar mediante la Figura 117 cómo se pudo acceder al sistema y monitorear el funcionamiento del módulo GPS y su cobertura a través de una computadora, confirmando de cierta forma su correcta interacción y la capacidad de gestión.

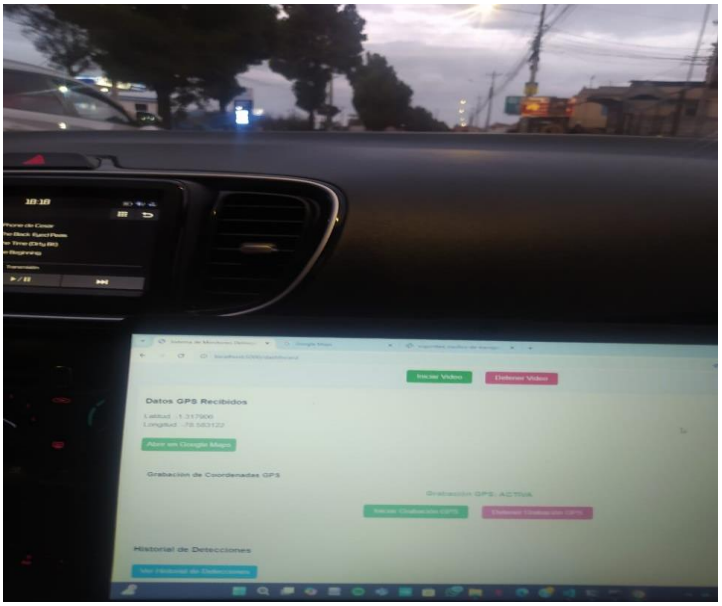


Figura 117. Ingreso al sistema de monitoreo desde un ordenador, para la toma de pruebas de cobertura del módulo GPS.

Tras la finalización y evaluación exitosa de las pruebas de procesamiento de imágenes para la detección de ilícitos, así como de las pruebas de cobertura y precisión del sistema de geoposicionamiento (GPS), la Figura 118 presenta la interfaz gráfica final del sistema desarrollado. Esta interfaz integró de manera coherente y funcional la visualización del flujo de video en tiempo real, permitiendo el monitoreo continuo, junto con la superposición y despliegue de los datos de georreferencia (latitud y longitud) del medio de transporte. Esta combinación proporciona una herramienta robusta para el seguimiento y la detección temprana de eventos relevantes.

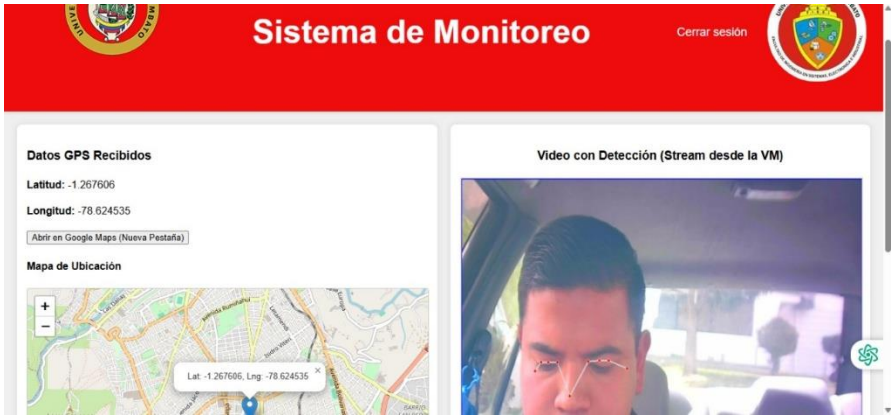


Figura 118. Integración total del sistema dentro de la interfaz gráfica final.

3.6.2 Estructura operativa.

El prototipo consta específicamente de la Unidad de procesamiento Nvidia Jetson Nano como dispositivo principal para el funcionamiento del sistema, la cual se conecta a la red de Internet que proporciona el portable travel router wifi de forma inalámbrica. Como se acabó de mencionar la Nvidia Jetson Nano es el elemento principal del sistema, entonces a ella se encuentran conectados el resto de los dispositivos. Por medio de cable USB se conectan a ella; la cámara Webcam y la placa Microcontroladora Arduino UNO. El acceso a internet es crucial para el envío de las alertas por medio de Slack. La Figura 119 muestra la estructura operativa del sistema mediante un esquema de conexiones.

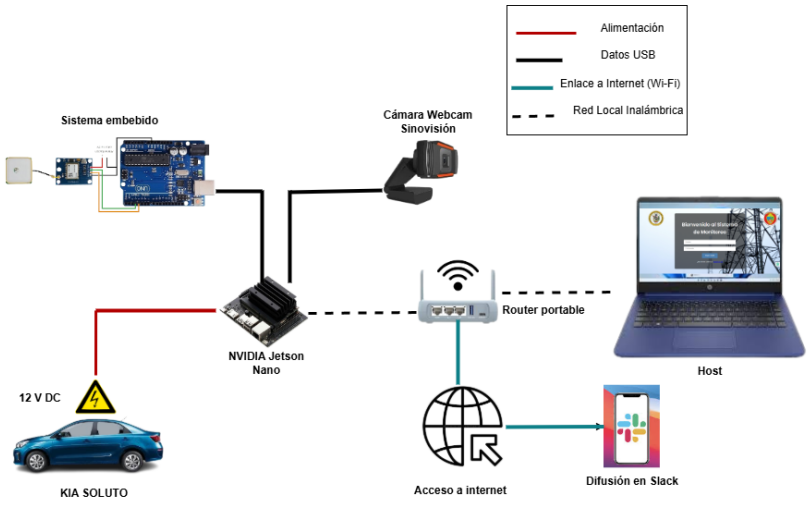


Figura 119. Estructura operativa del sistema mediante un esquema de conexiones de los equipos utilizados.

3.6.3 Requisitos mínimos de hardware y software para la implementación.

Luego de las pruebas llevadas a cabo en la sección de Verificación de funcionamiento del sistema se han definido algunos requerimientos básicos de hardware y software para la implementación del sistema. La Tabla 25 sintetiza estos requerimientos. En la sección de anexos, específicamente en el Anexo C se encuentra las especificaciones técnicas de la Unidad de Procesamiento NVIDIA Jetson Orin de 8 GB.

Tabla 25. Requisitos mínimos de hardware y software para la implementación.

Categoría	Etapa	Parámetro	Descripción
Hardware	Adquisición de Video	Conectividad	USB
		Resolución	720p
		Tipo de Cámara	Webcam Sinovisión
	Adquisición de Geoposicionamiento	Módulo GPS	Neo 6M
		Microcontrolador	Arduino UNO
	Unidad de Procesamiento	Modelo	NVIDIA Jetson Orion 8GB
		GPU	NVIDIA Ampere architecture (con 1024 CUDA Cores, 32 Tensor Cores)
		CPU	6-core ARM Cortex-A78AE v8.2 64-bit CPU
		Memoria	8 GB 128-bit LPDDR5
		Almacenamiento Interno	32 GB eMMC 5.1
		Conectividad	Gigabit Ethernet, Wi-Fi
Software	Pre-procesamiento	Python	3.10.2
		Numpy	1.25.2
		Pandas	2.2.0
		OpenCV	4.9.0.80
		Scikit-Learn	1.4.0
	Reconocimiento	Ultralytics	8.3.36
		TensorFlow	2.15.0
		Torch	2.1.2
		Joblib	1.3.2
	Visualización / Backend	Flask	3.1.2
		Werkzeug	3.0.1
		Jinja2	3.1.2
		SQLite	3.46.1
		Node-RED	"Horizon" v3.3.0

3.6.4 Presupuesto

La ejecución de un sistema de monitoreo y detección de comportamiento delincuencial aplicado en medios de transporte utilizando visión artificial, se compuso de costos de diseño y de costos de implementación. Por lo tanto, a continuación, se enlista y se detalla cada uno de estos costos.

a. Costos de Diseño

El sueldo mensual de un ingeniero de Telecomunicaciones en Ecuador es de 858 dólares, según lo que establece el Ministerio del Trabajo del Ecuador. Considerando 21 días de trabajo mensuales con horarios de 8 horas diarias, se determinó el costo por hora laboral.

$$Salario_{\,X\,dia} = \frac{Salario_{\,AL\,mes}}{Días\ de\ trabajo\ al\ mes} \tag{19}$$

$$Salario_{\,X\,dia} = \frac{\$\,858}{21} = \$\,40.86$$

$$Salario_{\,X\,hora} = \frac{Salario_{\,X\,día}}{Horas\ de\ trabajo\ al\ día} \tag{20}$$

$$Salario_{\,X\,hora} = \frac{\$\,40.86}{8} = \$\,5.11$$

El costo de diseño fue establecido en base a las horas de trabajo que se invirtió en el diseño, desarrollo y en la implementación de este sistema de monitoreo y detección delincuencial, las cuales fueron alrededor de 145 horas, multiplicándolas por el valor de la hora de trabajo de un Ingeniero en Telecomunicaciones establecido en $ 5.11 en el Ecuador.

$$Costo_{\ DEL\ Diseño} = Horas\ invertidas * \ Salario_{\ X\ hora} \quad (21)$$

$$Costo_{\ DEL\ Diseño} = 145 * \$\ 5.11$$

$$Costo_{\ DEL\ Diseño} = \$\ 790.45$$

b. Costos de implementación

Los costos de implementación se componen de todos los materiales que fueron parte dentro de la implementación del sistema, considerando los impuestos e imprevistos presentados. En el Anexo E se presenta las especificaciones técnicas de la webcam HD (720p) de Sinovisión; misma que fue escogida dado que esta resolución fue seleccionada tras considerar la necesidad de un equilibrio óptimo entre la calidad visual requerida para el análisis por visión artificial (detección de objetos y posturas) y la capacidad de procesamiento del hardware disponible, una NVIDIA Jetson Orin Nano. La Tabla 26 condensa los costos de implementación del sistema.

Tabla 26. Costos de implementación

Punto	**Elemento**	**Cantidad**	**Costo Unitario**	**Costo Total**
1	Unidad de procesamiento: Nvidia Jetson Orin Nano	1	$ 250.00	$ 250.00
2	Cámara Webcam Sinovisión	1	$ 20.00	$ 20.00
3	Placa microcontroladora Arduino UNO	1	$ 20.00	$ 20.00
4	Módulo GPS Neo 6m	1	$ 25.00	$ 25.00
5	Portable travel router wifi	1	$ 35.00	$ 35.00
6	Impresión de la carcasa del prototipo en 3D	1	$ 35.00	$ 35.00
TOTAL				**$ 385.00**

c. Costo total del proyecto

El costo total del proyecto se compone tanto del costo de diseño como del costo de la implementación, por lo que se consideraron ambos valores previamente calculados y se sumaron para obtener el monto total.

$$Costo_{Total} = Costo_{\ DEL\ Diseño} + Costo_{\ DE\ Implementación} \quad (22)$$

$$Costo_{Total} = \$\,790.45 + \$\,385.00 = \$\,1.175{,}45$$

CAPÍTULO IV. CONCLUSIONES Y RECOMENDACIONES

4.1 Conclusiones

- El análisis reveló que, aunque el 65% de los sistemas de seguridad en el transporte usan videovigilancia, solo el 40% es percibido como efectivo, con un 60% de encuestados reportando incidentes frecuentes. Las principales fallas son la falta de cobertura (55%) y la lentitud en la respuesta (30%), evidenciando una carencia de herramientas automatizadas en tiempo real y excesiva dependencia humana. Esta situación resalta la necesidad crítica de integrar inteligencia y visión artificial, una solución que el 88% de los encuestados está dispuesto a aceptar. En síntesis, los sistemas tradicionales son insuficientes, lo que valida la necesidad y establece una sólida base para el desarrollo de un sistema de monitoreo delincuencial basado en visión artificial y geoposicionamiento.

- El sistema de visión artificial desarrollado demostró una alta eficacia operacional en el monitoreo y detección de delitos. Integrando YOLOv8 para objetos y SVM con MediaPipe para posturas, por lo que alcanzó un F1-score de 0.896 y por lo tanto, un rendimiento del sistema interpretado en un 89.6%. Esto reduce la dependencia de la vigilancia humana. Implementado en una NVIDIA Jetson Nano, el sistema procesa video a 30 FPS, optimizando la respuesta a incidentes y minimizando costos operativos.

- La arquitectura modular, combinando YOLOv8 para la detección de objetos (armas, cuchillos y gorras) y un clasificador SVM con MediaPipe para posturas (asalto con cuchillo, asalto con arma), demostró ser fundamental. Esta integración permitió establecer un contexto situacional enriquecido, logrando una alta tasa de detección temprana de delitos. Además, esta arquitectura reduce significativamente los falsos positivos comparado con sistemas de detección única, mejorando la fiabilidad de las alertas enviadas por Slack, que incluyen evidencia visual a través de Google Drive.

- El sistema se implementó sobre una infraestructura lógica y robusta que asegura su operación continua y respuesta en tiempo real. La conexión MQTT para datos GPS garantiza que cada alerta incluya coordenadas geográficas exactas, mejorando la información para emergencias. La integración de la Google Drive API para el almacenamiento de imágenes y las notificaciones inmediatas vía Slack Webhooks reducen la latencia de propagación de alertas de minutos a segundos, lo que optimiza la reacción y la gestión de incidentes delictivos.

- La implementación de un sistema de gestión de datos eficaz redujo drásticamente el tiempo de reacción ante delitos. El almacenamiento en SQLite permite una recuperación y análisis casi inmediatos, procesando hasta 30 detecciones por minuto. La integración de datos GPS vía MQTT asegura que cada alerta contenga coordenadas geográficas exactas, facilitando la acción inmediata. Además, las alertas por Slack Webhooks y la subida de imágenes a Google Drive con URLs públicas garantizan que la evidencia visual esté disponible en segundos, optimizando la respuesta de las autoridades.

- La arquitectura del sistema, que almacena las detecciones en una base de datos SQLite y las utiliza para alimentar un "conjunto de datos de nuevo etiquetado automático", establece un mecanismo robusto para la mejora continua del modelo. Esta retroalimentación de datos posibilita el reentrenamiento iterativo de los modelos YOLOv8 medium y SVM, con potencial para incrementar su precisión global más allá del 87.5% y 74% respectivamente. La gestión eficiente de los datos (limitando a las últimas 20 detecciones y eliminando archivos antiguos) garantiza la sostenibilidad del almacenamiento y la capacidad de actualización de los modelos, impulsando la evolución continua de la capacidad predictiva del sistema.

- La estrategia de datos optimiza la respuesta y garantiza la escalabilidad y consistencia del sistema. Al limitar el historial y automatizar la eliminación de capturas en la NVIDIA Jetson Nano, se previene el sobre almacenamiento y se asegura la operación con información actualizada. Este método, que genera un "conjunto de datos de nuevo etiquetado automático", permite una mejora

continua del modelo, previendo un aumento anual del 5-10% en la precisión. Así, el sistema se mantiene relevante y eficaz frente a nuevas amenazas delictivas.

4.2 Recomendaciones

- Para maximizar el rendimiento en la NVIDIA Jetson Nano, se recomienda la optimización continua del uso de GPU y RAM. Esto incluye implementar la cuantización de modelos (YOLOv8) para reducir su tamaño y acelerar la inferencia hasta 2-3x, con una mínima pérdida de precisión. Adicionalmente, se sugiere integrar un monitoreo de recursos en tiempo real para ajustar dinámicamente la carga o resolución si la GPU excede un umbral crítico, asegurando una operación fluida a 30 FPS.

- Para mejorar la capacidad de generalización de los modelos YOLOv8 y SVM, es crucial aplicar estrategias avanzadas de aumento de datos (data augmentation). Esto va más allá de las transformaciones básicas e incluye simular condiciones contextuales reales del transporte, como variaciones de iluminación, oclusiones parciales, diversos ángulos de cámara, e incluso efectos de niebla o lluvia sintética. Un incremento del 20-30% en la variabilidad del dataset puede traducirse en una mejora del 3-5% en la precisión y el F1-score en entornos dinámicos.

- Al construir y preparar datasets en plataformas como Roboflow, es crucial asegurar un balance equitativo de instancias para cada clase de objeto. Un desequilibrio significativo puede sesgar el entrenamiento del modelo, resultando en un rendimiento deficiente y un F1-score bajo para las clases minoritarias. Por eso, es primordial procurar que cada clase tenga una representación balanceada, idealmente con una variación no superior al 10-15% entre la clase más y menos representada. Esto garantiza que el modelo YOLOv8 aprenda a detectar robustamente todas las categorías, y no solo las más comunes.

- Dado que el sistema monitorea eventos delictivos críticos, implementar redundancia y recuperación ante fallos es esencial. Esto incluye la capacidad de reiniciar automáticamente el proceso de video y la cámara (cv2.VideoCapture(0)) ante interrupciones inesperadas. Además, es crucial incorporar un watchdog timer que monitoree el estado de los hilos de procesamiento (MQTT y generación de frames). Si un hilo no responde en un tiempo predefinido (ej., 30 segundos), el watchdog podría forzar su reinicio o notificar a un operador, garantizando la operatividad continua y minimizando el tiempo de inactividad.

- Se recomienda llevar en ejecución un proceso estricto de calibración y validación periódica del sistema de visión artificial en su entorno operacional real. Esto implica recolectar un pequeño conjunto de datos de prueba bajo condiciones ambientales variadas (iluminación, ángulos, oclusiones parciales) que simulen el 5% de los escenarios más desafiantes. Evaluar la precisión, recall y F1-score en estas nuevas condiciones permite identificar cualquier degradación en el rendimiento del modelo y activar un ciclo de reentrenamiento adaptativo.

- Para la comercialización del sistema de detección delincuencial en transporte, se recomienda priorizar cámaras de visión artificial con alta resolución (Full HD), tasas de refresco de al menos 30 FPS, y capacidad de procesamiento integrado para IA en el borde. Modelos como la Basler daA1920-30uc, la Vivotek MD9584-H o la cámara Arducam USB 3.0 Global Shutter cumplen con estos requisitos técnicos esenciales para una detección precisa y en tiempo real. Aunque su costo inicial sea elevado, la inversión se justifica por su robustez, rendimiento superior en entornos dinámicos y la reducción de falsos positivos, optimizando así la efectividad y escalabilidad a largo plazo del sistema comercial.

REFERENCIAS BIBLIOGRÁFICAS

[1] L. Chinchilla and D. Vorndran, “Seguridad ciudadana en América Latina y el Caribe: Desafíos e innovación en gestión y políticas públicas en los últimos 10 años.” 2018.

[2] S. Rodríguez, “La inseguridad en el transporte público de Ecuador: un análisis de la situación en Quito y Guayaquil,” *Revista de Seguridad Pública*, vol. 15, no. 4, pp. 45–60, 2022.

[3] A. Pérez, “La seguridad en el transporte urbano en Ambato: Retos y soluciones.” 2024.

[4] R. Li, C. Yu, X. Qin, X. An, J. Zhao, W. Chuai, and B. Liu, “YOLO-SGC: A Dangerous Driving Behavior Detection Method With Multiscale Spatial-Channel Feature Aggregation,” *IEEE Sensors Journal*, vol. 24, no. 21, pp. 36044–36056, 2024.

[5] D. B. Patil and R. Kumar, “Moving Vehicle Detection with Real-Time Speed Estimation Using Deep Learning Techniques,” in *Proceedings of the 2023 IEEE International Conference on Machine Learning and Cybernetics (ICMLC)*, 2023, vol. 1, pp. 312–317.

[6] J. Q. Zhang, X. Xu, and P. X. Chen, “Deep Reinforcement Learning for Intelligent Transportation Systems: A Survey,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 53, no. 6, pp. 1181–1203, 2023.

[7] C.-H. Tseng and H.-Y. Lin, “A Vision-Based System for Abnormal Behavior Detection and Recognition of Bus Passengers,” in *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, 2022, pp. 2134–2139.

[8] H. B. Naeem, M. H. Yousaf, F. H. Khan, and A. Yasin, “Vision-Based Real-time Human Malicious Behavior Detection,” in *2021 International Conference on Artificial Intelligence (ICAI)*, 2021, pp. 193–197.

[9] S. (Zech) Lee, “A Basic Principle of Physical Security and Its Link to Cybersecurity,” *International Journal of Cyber Criminology*, vol. 14, no. 1, pp. 203–219, Jan. 2020.

[10] A. Lamssaggad, N. Benamar, A. S. Hafid, and M. Msahli, “A Survey on the Current Security Landscape of Intelligent Transportation Systems,” *IEEE Access*, vol. 9, pp. 9180–9208, 2021.

[11] A. N. del Ecuador, “Ley Orgánica de Vigilancia y Seguridad Privada.” 2024.

[12] J. RODRÍGUEZ FERNÁNDEZ, *Circuito cerrado de televisión y seguridad electrónica 2.textordfeminine edición*. Ediciones Paraninfo, S.A, 2018.

[13] A. Communications, “AXIS P5635-E PTZ Network Camera.” 2024.

[14] B. S. Systems, “AUTODOME IP starlight 7000i.” 2024.

[15] D. Technology, “N52A Series.” 2024.

[16] Sony, “SNC-VB770.” 2024.

[17] Tecnoseguro, “¿Qué es CCTV?” 2025.

[18] E. Vivas, “Utilización de herramientas tecnológicas para el desarrollo de una política pública de seguridad pública – El caso de la Ciudad Autónoma de Buenos Aires,” Universidad Torcuato Di Tella, 2022.

[19] D. A. Rodríguez and E. Vergel, “Desarrollo urbano orientado a los sistemas de transporte público masivo tipo BRT (Bus Rapid Transit) en Quito y Bogotá.” 2024.

[20] T. D. of Insurance, “Hoja Informativa de Seguridad para los Conductores de Taxis, Transporte Compartido y Choferes.” 2025.

[21] R. P. González, “Análisis de la tendencia del crimen violento en la Zona Metropolitana del Valle de México,” *Estudios Demográficos y Urbanos (EDU)*, vol. 39, no. 2, pp. 1–36, Jun. 2024.

[22] R. Fernandez, “Análisis del problema del transporte urbano,” *Ciencia Al Día*, vol. 1, pp. 1–9, 1999.

[23] J. Velasco Rebolledo, *Machine learning: Fundamentos, algoritmos y aplicaciones para los negocios, industria y finanzas*. Ediciones Dıaz de Santos, 2024.

[24] K. V. M. Acevedo, “Machine Learning: algoritmos de clasificación y sus aplicaciones en el análisis de datos,” *Thesis*, 2017.

[25] A. C. Müller and S. Guido, *Introduction to Machine Learning with Python: A Guide for Data Scientists*. O’Reilly Media, Inc., 2017.

[26] C. González and others, “SVM - Departamento de Informática.” .

[27] G. Cloud, “AI Platform.” .

[28] HostGator, “Google Colab: ¿Qué es y para qué sirve?” .

[29] I. Amazon Web Services, “¿Qué es AWS? - Computación en la nube con Amazon Web Services.” 2024.

[30] I. Amazon Web Services, “Datos, análisis e IA unificados.” 2024.

[31] A. W. Services, “Building Secure Machine Learning Environments with Amazon SageMaker.” 2024.

[32] Microsoft, “Azure Developer Guide.” .

[33] Tulip, “Microsoft Azure Machine Learning Integration.” .

[34] IBM, “Conceptos básicos de ayuda de CRISP-DM.” .

[35] Data-Flair, “Data Mining Process.” .

[36] G. I. G. R., “Metodología para Machine Learning (I): CRISP-DM.” .

[37] C. Pocho, “Machine Learning Canvas.” .

[38] G. I. G. R., “Metodología para Machine Learning (II): Machine Learning Canvas.” 2024.

[39] B. University, “Data Mining: SEMMA.” 2021.

[40] IBM, “Confusion Matrix: Qué es y cómo usarla en el análisis de modelos de Machine Learning.” 2023.

[41] S. Besse, “Neural networks and introduction to deep learning.” 2009.

[42] F. Lubinus Badillo, C. A. Rueda Hernández, B. Marconi Narváez, and Y. E. Arias Trillos, “Redes neuronales convolucionales: un modelo de Deep Learning en imágenes diagnósticas. Revisión de tema,” *Revista colombiana de radiología*, vol. 32, no. 3, pp. 5591–5599, 2021.

[43] M. I. I. Durán, “Fundamentos Teóricos de un Sistema de Visión Artificial.” 2023.

[44] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014, pp. 580–587.

[45] E. R. S. Guzmán, “Entrenamiento de la Red Neuronal Convolucional YOLO para objetos propios.” Universidad Nacional Autónoma de México, Programa de Maestría y Doctorado en Ingeniería, Ingeniería Eléctrica - Procesamiento Digital de Señales, Ciudad Universitaria, CD. MX., 2020.

[46] E. Av, “Detección de objetos con YOLO: Implementaciones y cómo usarlas,” *Medium*, 2020.

[47] Ultralytics, “Modelos de Ultralytics.” 2025.

[48] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks.” 2016.

[49] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, “SSD: Single Shot MultiBox Detector,” in *Computer Vision – ECCV 2016*, 2016, pp. 21–37.

[50] F. R. Pi, “Raspberry Pi 4 Product Brief.” 2020.

[51] A. G. Electrónica, “Banana Pi BPI-M2 Zero.” 2022.

[52] I. E. Hoy, “Tarjeta CPU Banana Pi BPI-M4 Zero.” 2023.

[53] N. Corporation, “Jetson Nano Developer Kit.” 2022.

[54] Kinsta, “Crea una Aplicación con FastAPI para Python.” 2025.

[55] M. Wolph, “CherryPy Documentation, Release 10.0.1.dev10+ng1a62192.d20170207.” 2017.

[56] Flask, “Flask Documentation (3.1.x).” 2025.

[57] A. Pozo-Ruz, A. Ribeiro, M. C. García-Alegre, L. García, D. Guinea, and F. Sandoval, “Sistema de posicionamiento global (GPS): Descripción, análisis de errores, aplicaciones y futuro.” 2025.

[58] u-blox, “NEO-6 u-blox 6 GPS Modules.” 2011.

[59] R. Santos and S. Santos, “Guía del módulo GPS NEO-6M con Arduino.” Mar-2018.

[60] u-blox, “ZED-F9P-04B: High precision GNSS module.” 2024.

[61] u-blox, “ZED-F9P-01B: High precision GNSS module.” 2025.

[62] MediaTek, “MT3339 All-in-One GNSS Datasheet.” Jan-2017.

ANEXOS

Anexo A. Categorización de variables.

En la Figura A1 se presenta la clasificación de las variables vinculadas o relacionadas con el desarrollo del trabajo de titulación.

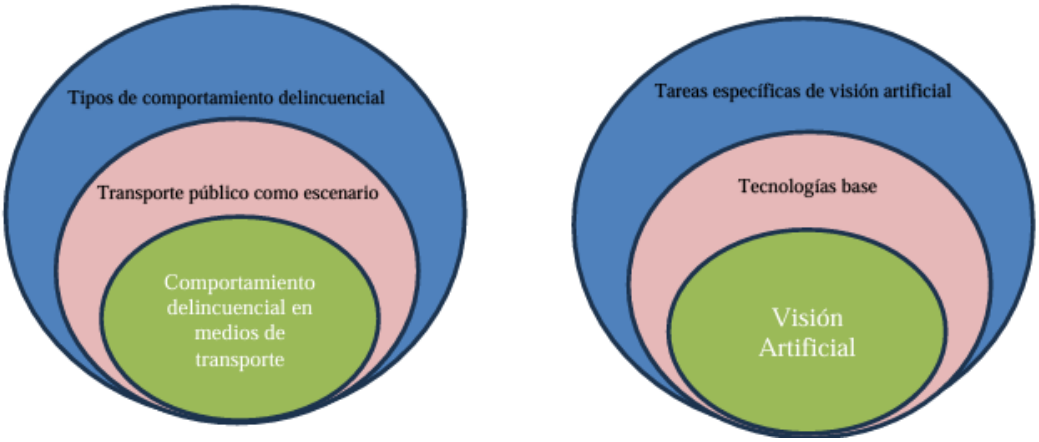


Figura A1. Categorización de variables.

En la Figura A2 se muestra el árbol de problemas para el desarrollo del trabajo de titulación.



Figura A2. Árbol de problemas.

Anexo B. Pruebas del sistema.

En las figuras B1, B2, B3 y B4 se muestra unas detecciones en el interior de un vehículo.



Figura B1. Detección en un entorno real de un arma.



Figura B2. Detección en un entorno real de un arma, un cuchillo y gorra.



Figura B3. Detección en un entorno real de un cuchillo.



Figura B3. Detección en un entorno real de un cuchillo y un arma.

Anexo C. Especificaciones técnicas de la NVIDIA Jetson Orin 8 GB.

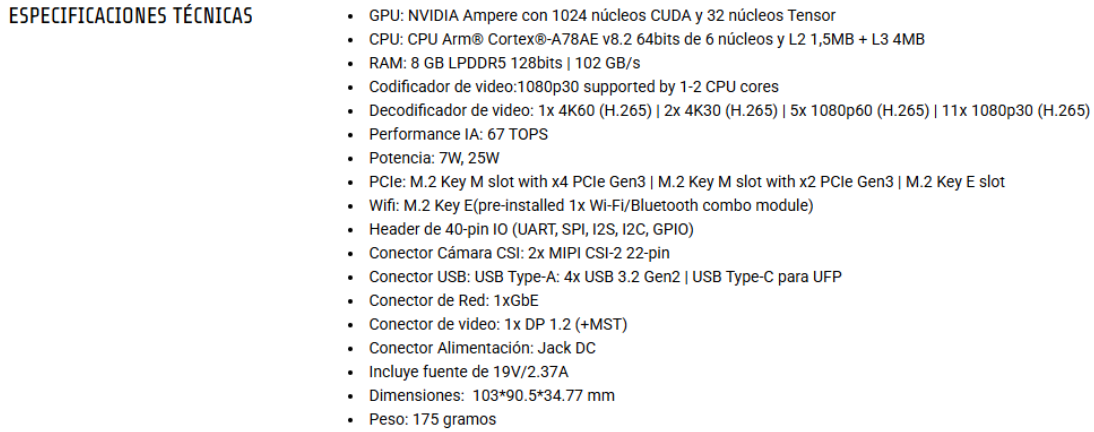


Figura C1. Especificaciones técnicas de la NVIDIA Jetson Orin 8 GB.

Anexo D. Código del sistema.

```
import eventlet
eventlet.monkey_patch() # Esto debe estar antes de cualquier otro
import

from flask import Flask, render_template, session, redirect, url_for,
request, flash, jsonify, Response
from flask_sqlalchemy import SQLAlchemy
from flask_socketio import SocketIO
import paho.mqtt.client as mqtt
from datetime import datetime
import json
import cv2
import mediapipe as mp
import joblib
from ultralytics import YOLO
import numpy as np
import os
import threading
import requests
import time
import ast # Importar ast para literal_eval

# --- MODELOS ML ---
svm_model =
joblib.load('C:/Users/jonat/myproject/Posturas/modelo_entrenado_pose.sa
v')
yolo_model = YOLO('C:/Users/jonat/myproject/ModeloYOLO/best.pt')
```

```
pose = mp.solutions.pose.Pose(min_detection_confidence=0.5,
min_tracking_confidence=0.5)
mp_drawing = mp.solutions.drawing_utils

labels_svm = {0: "Normal", 1: "AsaltoCuchillo", 2: "AsaltoArma"}
labels_yolo = ["gorras", "armas", "cuchillo"]

window_width, window_height = 640, 480
object_weights = {"gorras": 25, "armas": 100, "cuchillo": 80}

video_processing = False
camera = None # Inicializar la cámara como None
```

Anexo E. Especificaciones técnicas de la webcam HD (720p) de Sinovisión.

Caracteristicas

Color	Negro/Naranja
Con micrófono	Si
Es apta para videoconferencia	Si
Funciones	Micrófono integrado con reducción de ruido
Interfaces	USB 2.0
Marca	Sinovision
Modelo	SN-U3
Resolución de imagen de la cámara	2 Mpx
Resolución máxima de video	1920 px x 1080 px
Tipo de resolución máxima de video	Full HD

Accesorios incluidos	Clip
Ancho	11 cm
Cuadros máximos por segundo	20 FPS
Dispositivos recomendados	Notebook, PC de escritorio
Largo del cable	1.5 m
MPN	SN-U3
Peso	130 g
Sistemas operativos mínimos requeridos	Linux, Windows, macOS
Tipo de sensor	CMOS
Ángulo de visión	75°

`Figura E1.` Especificaciones técnicas de la webcam HD (720p) de Sinovisión.