



UTMACH

FACULTAD DE INGENIERÍA CIVIL

CARRERA DE TECNOLOGÍAS DE LA INFORMACIÓN

Desarrollo de una aplicación móvil para detección de plazas disponibles y ocupadas en un estacionamiento utilizando Aprendizaje Profundo

**AGUILAR BRAVO SELENA ALEXANDRA
INGENIERA EN TECNOLOGIAS DE LA INFORMACION**

**CARRION LLUMILUISA JHONY ANDRES
INGENIERO EN TECNOLOGIAS DE LA INFORMACION**

**MACHALA
2022**



UTMACH

FACULTAD DE INGENIERÍA CIVIL

CARRERA DE TECNOLOGÍAS DE LA INFORMACIÓN

**Desarrollo de una aplicación móvil para detección de plazas
disponibles y ocupadas en un estacionamiento utilizando
Aprendizaje Profundo**

**AGUILAR BRAVO SELENA ALEXANDRA
INGENIERA EN TECNOLOGIAS DE LA INFORMACION**

**CARRION LLUMILUISA JHONY ANDRES
INGENIERO EN TECNOLOGIAS DE LA INFORMACION**

**MACHALA
2022**



UTMACH

FACULTAD DE INGENIERÍA CIVIL

CARRERA DE TECNOLOGÍAS DE LA INFORMACIÓN

PROPUESTAS TECNOLÓGICAS

Desarrollo de una aplicación móvil para detección de plazas disponibles y ocupadas en un estacionamiento utilizando Aprendizaje Profundo

**AGUILAR BRAVO SELENA ALEXANDRA
INGENIERA EN TECNOLOGIAS DE LA INFORMACION**

**CARRION LLUMILUISA JHONY ANDRES
INGENIERO EN TECNOLOGIAS DE LA INFORMACION**

NOVILLO VICUÑA JOHNNY PAUL

**MACHALA
2022**

Trabajo de titulación

por Selena Aguilar

Fecha de entrega: 09-sep-2022 10:49a.m. (UTC-0500)

Identificador de la entrega: 1895964964

Nombre del archivo: TURNITIN_-_TIC_Aguilar-Carrion.pdf (270.89K)

Total de palabras: 12975

Total de caracteres: 70141

Trabajo de titulación

INFORME DE ORIGINALIDAD

1 %

INDICE DE SIMILITUD

1 %

FUENTES DE INTERNET

0 %

PUBLICACIONES

0 %

TRABAJOS DEL
ESTUDIANTE

FUENTES PRIMARIAS

1 Submitted to Curtin University of Technology <1 %
Trabajo del estudiante

2 repositorio.espe.edu.ec <1 %
Fuente de Internet

3 www.nacion.com <1 %
Fuente de Internet

4 www.upnqueretaro.edu.mx <1 %
Fuente de Internet

5 blogs.embarcadero.com <1 %
Fuente de Internet

6 uvadoc.uva.es <1 %
Fuente de Internet

7 "Qualitative approaches to research on plurilingual education / Enfocaments qualitativus per a la recerca en educació plurilingüe / Enfoques cualitativos para la investigación en educación plurilingüe", Research-Publishing.net, 2017 <1 %
Publicación

CLÁUSULA DE CESIÓN DE DERECHO DE PUBLICACIÓN EN EL REPOSITORIO DIGITAL INSTITUCIONAL

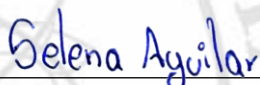
Los que suscriben, AGUILAR BRAVO SELENA ALEXANDRA y CARRION LLUMILUISA JHONY ANDRES, en calidad de autores del siguiente trabajo escrito titulado Desarrollo de una aplicación móvil para detección de plazas disponibles y ocupadas en un estacionamiento utilizando Aprendizaje Profundo, otorgan a la Universidad Técnica de Machala, de forma gratuita y no exclusiva, los derechos de reproducción, distribución y comunicación pública de la obra, que constituye un trabajo de autoría propia, sobre la cual tienen potestad para otorgar los derechos contenidos en esta licencia.

Los autores declaran que el contenido que se publicará es de carácter académico y se enmarca en las disposiciones definidas por la Universidad Técnica de Machala.

Se autoriza a transformar la obra, únicamente cuando sea necesario, y a realizar las adaptaciones pertinentes para permitir su preservación, distribución y publicación en el Repositorio Digital Institucional de la Universidad Técnica de Machala.

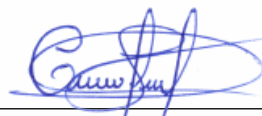
Los autores como garantes de la autoría de la obra y en relación a la misma, declaran que la universidad se encuentra libre de todo tipo de responsabilidad sobre el contenido de la obra y que asumen la responsabilidad frente a cualquier reclamo o demanda por parte de terceros de manera exclusiva.

Aceptando esta licencia, se cede a la Universidad Técnica de Machala el derecho exclusivo de archivar, reproducir, convertir, comunicar y/o distribuir la obra mundialmente en formato electrónico y digital a través de su Repositorio Digital Institucional, siempre y cuando no se lo haga para obtener beneficio económico.



AGUILAR BRAVO SELENA ALEXANDRA

0940702467



CARRION LLUMILUISA JHONY ANDRES

0705970291

DEDICATORIA

Dedico el presente trabajo a mi familia, quienes me han apoyado desde mis inicios hasta la actualidad, especialmente a mis padres por su ayuda incondicional en el transcurso de mi formación académica-profesional, por no dejar de creer en mí y a todas las personas que me han guiado y aconsejado.

Aguilar Bravo Selena Alexandra

Este trabajo está dedicado a mis padres, quienes constantemente me brindan de su apoyo, sabiduría y amor incondicional, lo que me ha permitido lograr cada meta que me he planteado desde pequeño. También, lo dedico a mi familia y amigos que siempre estuvieron pendiente de mí, apoyándome en el transcurso de mi formación académica para llegar a ser un profesional.

Carrión Llumiluisa Jhony Andrés

AGRADECIMIENTO

En primer lugar, quiero agradecer a Dios por darme fortaleza todos los días para poder continuar, por darme la sabiduría y sensatez en las situaciones que se me han presentado. A mis padres por luchar día a día para que logre avanzar en mi vida, por todo su amor y apoyo incondicional.

Finalmente quiero expresar mi más grande y sincero agradecimiento a mi tutor el Ing. Johnny Novillo, quien con paciencia me ha sabido guiar durante el proceso de titulación y a los demás docentes quienes me han impartido sus enseñanzas y conocimientos durante mi periodo de formación.

Aguilar Bravo Selena Alexandra

Primeramente, doy gracias a Dios por permitirme culminar cada una de las metas que me he propuesto a lo largo de mi vida. De igual manera un agradecimiento especial a mis padres por siempre brindarme ese apoyo y amor incondicional, lo que me ha permitido continuar y terminar mi formación profesional. También, un agradecimiento al Ing. Johnny Novillo por la orientación brindada durante todo el proceso de titulación y a cada docente de la carrera por las enseñanzas inculcadas.

Carrión Llumiluisa Jhony Andrés

RESUMEN

Un estacionamiento vehicular es una zona usada por propietarios de vehículos. Al momento que ellos van a buscar un espacio de estacionamiento y se encuentra ocupados se convierte en un problema, es por eso que el aumento vehicular es un fenómeno que se visualiza con mayor frecuencia en varias zonas, ya que la mayoría de las personas tienen la economía suficiente para adquirir un vehículo. Algunas de las causas que provoca la congestión vehicular, es la gran cantidad de vehículos que circulan a ciertas horas en las ciudades, y también por la falta de espacios de estacionamientos disponibles en los parqueaderos, trayendo como consecuencia los retrasos en las movilizaciones de las personas, el aumento de la contaminación, los accidentes de tránsito, el consumo adicional y excesivo de combustible e incluso en ocasiones multas por el incumplimiento de las normas por estacionarse en espacios no permitidos. Un estacionamiento inteligente utiliza una tecnología que promete brindar a los conductores una mejor comodidad al momento de buscar una plaza de estacionamiento libre. Por tal motivo, el presente trabajo se enfocó en desarrollar una aplicación móvil, capaz de detectar plazas de estacionamientos, integrando Aprendizaje Profundo. Para esto, se tomaron algunas imágenes del parqueadero de la Facultad de Ingeniería Civil de la Universidad Técnica de Machala, para la elaboración del dataset, donde se establecieron algunos requisitos, tales como: la imagen debe ser de buena calidad, debe tener buena resolución, debe ser imágenes bajo diferentes condiciones de luz, entre otros. Las imágenes del dataset fueron clasificadas por clases, la primera clase corresponde a las imágenes de estacionamiento libre y la segunda clase al estacionamiento ocupado, para el entrenamiento del modelo de red neuronal convolucional se siguió una metodología genérica. El modelo entrenado fue evaluado con una matriz de confusión, la misma que cuenta con las métricas de precisión, sensibilidad y valor de referencia, esto con el propósito de seleccionar el mejor modelo entrenado. También se desarrolló un back-end que contiene la parte de la detección en tiempo real realizado con el IDE Visual Studio Code y el lenguaje de programación Python. Por otro lado, para el desarrollo de la aplicación móvil se utilizó el IDE de Android Studio y el lenguaje de programación Java, siguiendo los pasos de la metodología Mobile-D, en la que se ejecutaron sus fases de manera adecuada, como el cronograma que permitió tener un mejor control de las actividades durante el desarrollo del aplicativo. El diseño de la interfaz de la aplicación móvil es sencillo para el usuario e intuitivo con retroalimentación, con el fin del que el usuario sea capaz de deducir de una forma rápida cómo funciona cada una de sus botones y sesiones. Como resultado final, se obtuvo una aplicación móvil que permite visualizar, detectar e identificar las plazas disponibles y ocupadas en el estacionamiento de vehículos de la Facultad de Ingeniería Civil de la Universidad Técnica de

Machala, en tiempo real, aplicando Aprendizaje Profundo. Las pruebas se realizaron de manera local para verificar las funciones establecidas en los requerimientos.

PALABRAS CLAVE: Aprendizaje Profundo, Red neuronal convolucional, Detección de Estacionamientos.

ABSTRACT

A parking lot is an area used by vehicle owners. When they go to look for a parking space and find it occupied, it becomes a problem, which is why the increase in vehicle congestion is a phenomenon that is seen more frequently in several areas, since most people have enough money to buy a vehicle. Some of the causes of traffic congestion are the large number of vehicles that circulate at certain times in the cities, and also the lack of parking spaces available in the parking lots, resulting in delays in the mobilization of people, increased pollution, traffic accidents, additional and excessive fuel consumption and even sometimes fines for non-compliance with the rules for parking in spaces not allowed. Smart parking uses technology that promises to provide drivers with better convenience when looking for a free parking space. For this reason, the present work focused on developing a mobile application capable of detecting parking spaces, integrating Deep Learning. For this, some images of the parking lot of the Faculty of Civil Engineering of the Technical University of Machala were taken for the development of the dataset, where some requirements were established, such as: the image must be of good quality, must have good resolution, must be images under different light conditions, among others. The images of the dataset were classified by classes, the first class corresponds to the images of free parking and the second class to occupied parking. The trained model was evaluated with a confusion matrix, which has the metrics of Precision, Recall and F-Score, in order to select the best trained model. A back-end containing the real-time detection part was also developed using the Visual Studio Code IDE and the Python programming language. On the other hand, for the development of the mobile application, the Android Studio IDE and the Java programming language were used, following the steps of the Mobile-D methodology, in which its phases were executed properly, such as the chronogram that allowed to have a better control of the activities during the development of the application. The design of the interface of the mobile application is simple for the user and intuitive with feedback, so that the user is able to quickly deduce how each of its buttons and sessions work. As a final result, we obtained a mobile application that allows to visualize, detect and identify the available and occupied parking spaces in the parking lot of the Faculty of Civil Engineering of the Technical University of Machala, in real time, applying Deep Learning. The tests were performed locally to verify the functions established in the requirements.

KEY WORDS: Deep Learning, Convolutional Neural Network, Parking Detection.

ÍNDICE DE CONTENIDO

DEDICATORIA.....	II
AGRADECIMIENTO.....	III
RESUMEN.....	IV
ABSTRACT.....	VI
ÍNDICE DE CONTENIDO.....	VII
ÍNDICE DE TABLAS.....	XI
ÍNDICE DE FIGURAS.....	XII
GLOSARIO.....	XIV
INTRODUCCIÓN.....	15
CAPÍTULO I. MARCO TEÓRICO.....	19
1.1. Antecedentes de la Investigación.....	19
1.2. Antecedentes Teóricos.....	23
1.2.1. Trabajos Relacionados.....	23
1.2.2. Generalidades de Inteligencia Artificial.....	25
1.2.2.1. Historia de Inteligencia Artificial.....	25
1.2.2.2. Definición de Inteligencia Artificial.....	26
1.2.2.3. Beneficios del uso de Inteligencia Artificial.....	26
1.2.3. Generalidades del Aprendizaje Profundo.....	26
1.2.3.1. Definición de Aprendizaje Profundo.....	26
1.2.4. Redes Neuronales.....	27
1.2.4.1. Definición de Red Neuronal.....	27
1.2.4.2. Tipos de Red Neuronal.....	27
1.2.5. Aprendizaje o Entrenamiento.....	29
1.2.5.1. Definición de Aprendizaje.....	29
1.2.5.2. Tipos de Aprendizaje.....	29
1.2.6. Herramientas y/o materiales para la Construcción de la Red Neuronal.....	30

1.2.6.1.	DataSet.....	30
1.2.6.2.	Roboflow.....	30
1.2.6.3.	Google Colab	31
1.2.6.4.	Jupyter notebook.....	31
1.2.6.5.	YOLOv5	31
1.2.6.6.	TensorFlow	31
1.2.6.7.	Keras	32
1.2.6.8.	Visual Studio Code	32
1.2.6.9.	Python	32
1.2.6.10.	Android Studio.....	32
1.2.6.11.	Java	32
1.2.6.12.	Firebase.....	33
1.3.	Antecedentes Contextuales.....	33
1.3.1.	Ámbito de aplicación	33
1.3.2.	Establecimiento de requerimientos	34
CAPÍTULO II. DESARROLLO DEL PROTOTIPO		34
2.1.	Definición del prototipo.....	34
2.2.	Metodología de desarrollo del prototipo.....	36
2.2.1.	Enfoque, alcance y diseño de investigación.....	36
2.2.2.	Unidades de análisis	36
2.2.3.	Técnicas e instrumentos de recopilación de datos.....	37
2.2.4.	Técnicas de procesamiento de datos para la obtención de resultados.....	37
2.2.5.	Metodologías o métodos específicos.....	38
2.2.6.	Herramientas y/o Materiales	40
2.3.	Desarrollo del prototipo.....	40
2.3.1.	Metodología para la construcción de la Red Neuronal	40
2.3.1.1.	Elaboración del Dataset	40

2.3.1.2.	Construcción y entrenamiento de modelo de red neuronal.....	42
2.3.2.	Metodología Mobile-D para el desarrollo de la aplicación móvil.....	46
2.3.2.1.	Fase de Exploración.....	46
2.3.2.2.	Fase Inicialización.....	48
2.3.2.3.	Fase de Producción y Fase de Estabilización.....	54
2.3.2.4.	Fase de Pruebas del sistema.....	57
2.4.	Ejecución del prototipo.....	58
2.4.1.	Pantalla de inicio	58
2.4.2.	Pantallas informativas	58
2.4.3.	Módulo iniciar cámara.....	59
2.4.4.	Módulo de detección	60
CAPÍTULO III. EVALUACIÓN DEL PROTOTIPO		60
3.1.	Plan de evaluación	60
3.1.1.	Pruebas de la Red Neuronal	61
3.1.1.1.	Métricas.....	61
3.1.2.	Encuesta de cumplimiento de las funcionalidades de la app	62
3.2.	Resultados de la evaluación.....	63
3.2.1.	Resultados del entrenamiento de los modelos.....	63
3.2.2.	Resultados del rendimiento	64
3.2.3.	Resultados de la encuesta del cumplimiento de las funcionalidades de la app.....	67
CONCLUSIONES		70
RECOMENDACIONES		70
REFERENCIAS BIBLIOGRÁFICAS		71
ANEXOS.....		77
Anexo 1.	Matriz de consistencia.....	77
Anexo 2.	Encuesta de diagnóstico inicial para la elaboración del prototipo.....	82
Anexo 3.	Encuesta final para la evaluación del prototipo	84

Anexo 4. Tabulación de la Encuesta de diagnóstico inicial para la elaboración del prototipo .85

ÍNDICE DE TABLAS

Tabla 1. Conceptualización de la hipótesis	17
Tabla 2. Operacionalización de variables.....	18
Tabla 3. Preguntas de Investigación SRL	20
Tabla 4. Distribución de búsqueda por fuentes	21
Tabla 5. Miembros de la FIC y personas externas	36
Tabla 6. Datos de la dataset del entrenamiento y pruebas.....	37
Tabla 7. Técnica e instrumento de recopilación de datos.....	37
Tabla 8. Herramientas y/o materiales	40
Tabla 9. Establecimiento de los stakeholder	46
Tabla 10. Requerimientos Funcionales	47
Tabla 11. Requerimientos No Funcionales	47
Tabla 12. Recursos de Hardware.....	50
Tabla 13. Recursos de Software	50
Tabla 14. Planificación de fases.....	51
Tabla 15. Verificación de vistas	57
Tabla 16. Verificación de pruebas funcionales	58
Tabla 17. Actividades de la semana 10 de la evaluación del prototipo.....	61
Tabla 18. Actividades de la semana 11 de la evaluación del prototipo.....	61
Tabla 19. Matriz de Confusión.....	62
Tabla 20. Indicadores del entrenamiento modelo 1.....	63
Tabla 21. Resultados de métricas del entrenamiento modelo 1	63
Tabla 22. Indicadores del entrenamiento modelo 2.....	63
Tabla 23. Resultados de métricas del entrenamiento modelo 2	64
Tabla 24. Indicadores del entrenamiento modelo 3.....	64
Tabla 25. Resultados de métricas del entrenamiento modelo 3	64
Tabla 26. Resultado del modelo 1 - Matriz de Confusión.....	65
Tabla 27. Resultado del modelo 2 - Matriz de Confusión.....	66
Tabla 28. Resultado del modelo 3 - Matriz de Confusión.....	66
Tabla 29. Métricas de evaluación de las pruebas	67
Tabla 30. Cumplimiento de las funcionalidades de la app.....	69

ÍNDICE DE FIGURAS

Figura 1. Árbol de Problemas	16
Figura 2. Cantidad de trabajos buscados por tópico – Google Académico.....	22
Figura 3. Cantidad de artículos buscados por año – Scopus	22
Figura 4. Cantidad de documentos por autor – Scopus.....	23
Figura 5. Análisis por palabras claves – VOSviewer.....	23
Figura 6. Mapa temático de Revisión Sistemática de Literatura	25
Figura 7: Arquitectura de entrenamiento de la Red Neuronal	35
Figura 8. Arquitectura para la detección de plazas de estacionamiento.....	35
Figura 9. Metodología para la construcción de la Red Neuronal.....	38
Figura 10. Fases de la metodología Mobile-D (Adaptado de la metodología Mobile-D)	39
Figura 11: Software para etiquetar imágenes.....	41
Figura 12: Archivos txt generados de las etiquetas.....	41
Figura 13. Creación del dataset en Roboflow	42
Figura 14. Diseño de la Red Neuronal	42
Figura 15. Configuración del notebook de Google Colab	44
Figura 16. Instalación de instancias de requirements.txt	44
Figura 17. Entrenamiento de la Red Neuronal.....	45
Figura 18. Cronogramas de actividades	49
Figura 19. Estructura NoSQL de Total de estacionamiento disponibles	52
Figura 20. Boceto de la pantalla de inicio	52
Figura 21. Boceto informativo de la app.....	53
Figura 22. Boceto del menú principal	53
Figura 23. Boceto del módulo de detección	54
Figura 24. Estructura del Back-End.....	54
Figura 25. Creación de las URLs del Back-End	55
Figura 26. Función para la detección	55
Figura 27. Estructura del proyecto - Android Studio.....	55
Figura 28. Codificación del módulo Iniciar Cámara.....	56
Figura 29. Diseño de la Interfaz del módulo de detección.....	56
Figura 30. Codificación de la acción refrescar.....	57
Figura 31. Codificación de la función cargar página	57
Figura 32. Ejecución de la pantalla inicio	58
Figura 33. Ejecución de Pantallas informativas	59

Figura 34. Interfaz del módulo iniciar cámara	59
Figura 35. Ejecución de módulo iniciar cámara.....	60
Figura 36. Ejecución de módulo detectar.....	60
Figura 37. Resultado del rendimiento - modelo 1	65
Figura 38. Resultado del rendimiento - modelo 2.....	65
Figura 39. Resultado del rendimiento - modelo 3.....	66
Figura 40. Pregunta 1 de la encuesta de cumplimiento de funcionalidades.....	67
Figura 41. Pregunta 2 de la encuesta de cumplimiento de funcionalidades.....	68
Figura 42. Pregunta 3 de la encuesta de cumplimiento de funcionalidades.....	68
Figura 43. Pregunta 4 de la encuesta de cumplimiento de funcionalidades.....	68
Figura 44. Pregunta 5 de la encuesta de cumplimiento de funcionalidades.....	69
Figura 45. Pregunta 6 de la encuesta de cumplimiento de funcionalidades.....	69

GLOSARIO

A

Algoritmo: Un algoritmo es una secuencia de pasos lógicos que se utiliza para resolver problemas.

M

Metodología: Una metodología es un conjunto de procedimientos que se siguen en un trabajo investigativo, conlleva una serie de etapas específicas para el trabajo conduciendo a la selección de técnicas concretas.

Modelo Neuronal: Un modelo neuronal, es un conjunto de redes neuronales artificiales que parten de un modelo computacional que ha evolucionado con el pasar de los años. En otras palabras, son neuronas artificiales que transmiten señales y están conectadas entre sí.

S

Sistema: Un sistema es un conjunto de objetos que se relaciona con algún componente tangible o intangible.

V

VOSviewer: Es una herramienta instalable para sistemas operativos que ayuda a construir y visualizar redes bibliométricas con ayuda de revistas, investigadores o publicaciones individuales.

INTRODUCCIÓN

El ser humano con el uso de las nuevas tecnologías, ha tenido la inteligencia de diseñar y entrenar redes neuronales, capaces de igualar en pensamiento e imitación con el cerebro humano. Actualmente este tipo de tecnología está teniendo un gran impacto en la sociedad y es capaz de solucionar problemas comunes que se presentan día a día, a raíz de un modelo de red neuronal entrenada, conocido como Aprendizaje Profundo. Este tipo de aprendizaje facilita la detección de objetos, como en este caso, los espacios de estacionamientos disponibles y ocupados en los parqueaderos vehiculares. Para lograr que el modelo sea preciso se necesita una gran cantidad de fotografías, que constituyen el dataset. Entre más fotografías se utilicen para el entrenamiento menos porcentaje de error se tendrá.

i. Declaración y formulación del Problema

Declaración del problema

Actualmente el aumento de la aglomeración vehicular es un problema que se presenta en varias ciudades del mundo por el crecimiento de vehículos públicos y privados.

De acuerdo a [1], indica que la urbanización además de aumentar el tránsito en áreas urbanas, también provoca un aumento de la población. Un equivalente del 30% de las personas buscan una plaza disponible para estacionar su vehículo, esto indica que la congestión vehicular en las áreas urbanas siga en aumento en años posteriores.

El diario El Comercio [2], afirma que en Ecuador los vehículos matriculados han aumentado más de un 1.4 millones en solo un lapso de 10 años, lo que eleva el dígito a más de 2.4 millones de registros vehiculares en el año 2018, este reporte fue dado por el Instituto Nacional de Estadísticas y Censos (INEC).

Por tal motivo, en la Universidad Técnica de Machala se pretende tener una mejor gestión en el estacionamiento vehicular, teniendo como beneficiarios a los docentes, estudiantes, empleados y personas en particular, con el fin de que puedan localizar plazas de estacionamiento disponibles de manera rápida. En la Figura 1, se presenta el problema, causa y efecto del trabajo.

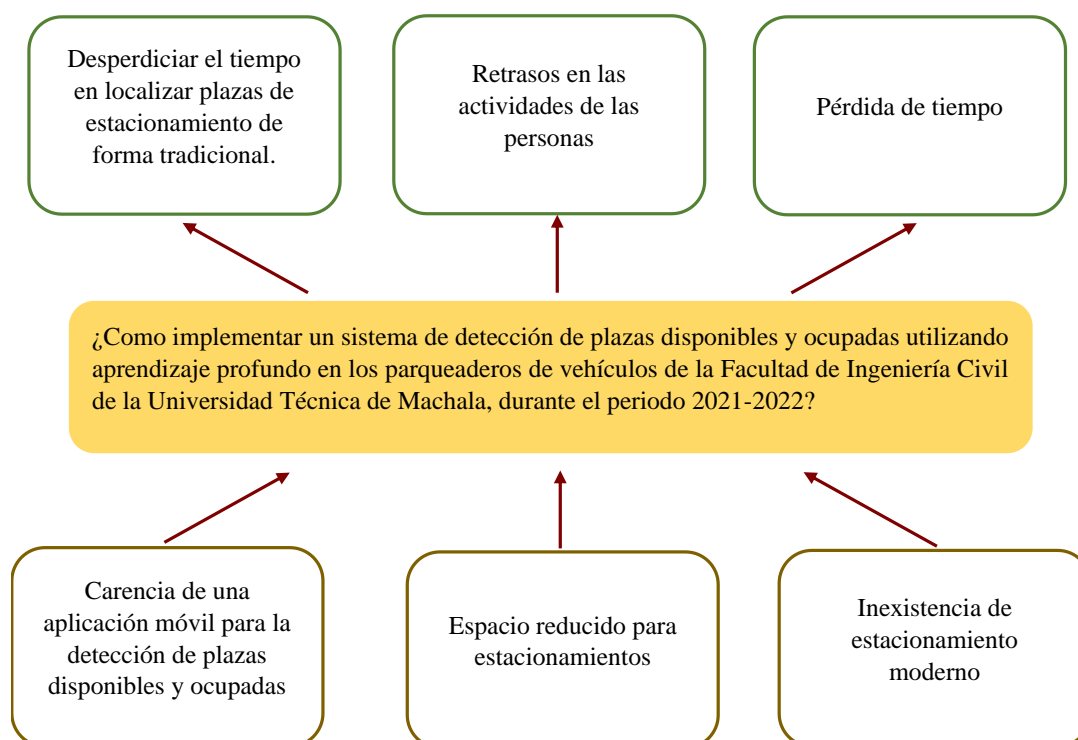


Figura 1. Árbol de Problemas

Formulación del problema

Problema principal:

¿Cómo implementar un sistema de detección de plazas disponibles y ocupadas utilizando aprendizaje profundo en los parqueaderos de vehículos de la Facultad de Ingeniería Civil de la Universidad Técnica de Machala, durante el periodo 2021-2022?

Problemas específicos:

- ¿Cómo obtener imágenes útiles para detectar espacios de estacionamiento disponibles?
- ¿Cuál es la técnica que se utilizará para la detección de plazas de estacionamiento?
- ¿Cómo vincular el aprendizaje profundo a la aplicación móvil, para la detección de plazas de estacionamiento?

ii. Objeto de estudio y Campo de acción

Objeto de estudio

- Sistema de detección de plazas de parqueo en un estacionamiento.

Campo de acción

- Desarrollo de una aplicación móvil integrando aprendizaje profundo.

iii. Objetivos

Objetivo General

Desarrollar una aplicación móvil utilizando un modelo de red neuronal de Aprendizaje Profundo y la metodología Mobile-D, para la detección de plazas disponibles y ocupadas en el estacionamiento de la Facultad de Ingeniería Civil de la Universidad Técnica de Machala.

Objetivos específicos

- Realizar una búsqueda bibliográfica, aplicando la Metodología de Revisión Sistemática de Literatura, para la elaboración del estado del arte y marco teórico.
- Planificar el cronograma y actividades de acuerdo a la metodología Mobile-D, para tener un adecuado seguimiento durante el desarrollo de la aplicación.
- Crear un dataset con imágenes del estacionamiento de la FIC (disponibles y ocupadas), para el entrenamiento del modelo con una red neuronal convolucional utilizando YOLOv5.
- Comprobar el funcionamiento de la detección de las plazas de estacionamiento disponibles y ocupadas, mediante la aplicación móvil.

iv. Hipótesis y variables o Preguntas de investigación

El desarrollo de una aplicación móvil que integra Aprendizaje Profundo o Deep Learning permitirá la detección de espacios disponibles y ocupados en el parqueadero de vehículos de la Facultad de Ingeniería Civil de la Universidad Técnica de Machala.

Variables y dimensionamiento

En la Tabla 1, se describe de manera breve la definición conceptual de las variables independiente y dependiente. Por otro lado, en la Tabla 2, se muestran las variables, categoría, indicadores y técnicas que surgieron al elaborar la hipótesis de este trabajo:

Tabla 1. Conceptualización de la hipótesis

Variables	Conceptos
Variable Independiente: Desarrollo de una Aplicación móvil que integra Aprendizaje Profundo	Aplicación móvil que integre Aprendizaje Profundo, hace referencia al uso de redes neuronales entrenadas para mejorar los sistemas como: procesamiento del lenguaje natural, reconocimiento de voz, etc.
Variables Dependiente: Detectar plazas disponibles y ocupadas en el estacionamiento de vehículos	Detectar plazas disponibles y ocupadas en el estacionamiento de vehículos, se refiere a la implementación del conjunto de algoritmos de aprendizaje automático, conocido como Aprendizaje profundo o Deep Learning.

Tabla 2. Operacionalización de variables

Variab les	Categorías	Indicadores	Técnicas
Variable Independiente: Desarrollo de una Aplicación móvil que integra Aprendizaje Profundo	1. Metodologías de desarrollo de aplicación móvil con IA 2. Deep Learning 3. Sistema de detección de parqueo	1. Tipos o comparación de metodología 2. Herramientas y lenguajes de programación de alto nivel Python y Java. 3. Recopilación de datos (imágenes). 4. Modelo de red neuronal 5. Métrica para evaluar el modelo entrenado.	1 Recopilación de información mediante las técnicas SRL. 1 Construcción de la dataset 2 Entrenamiento del modelo con Aprendizaje Supervisado 3 Red neuronal convolucional 4 Metodología de construcción neuronal 5 Metodología Mobile-D 6 Matriz de Confusión
Variables Dependiente: Detectar plazas disponibles y ocupadas en el estacionamiento de vehículos	Satisfacción del usuario en base a la funcionalidad de la app Eficacia	1. Visualizar los resultados.	Aprender información en base a imágenes

v. Justificación

El Aprendizaje Profundo (Deep Learning o DL), es usado por tener la característica de extraer grandes conjuntos de datos de entrenamiento, la mayor parte sin supervisión humana [3]. También para detectar objetos, reconocer conversaciones, tomar decisiones, entre otros.

Las grandes ciudades del mundo están comenzando a tener inconvenientes por las crecientes demandas de espacios de estacionamiento, este inconveniente es una de las necesidades que tienen todos los dueños de vehículos [4]. En otras palabras, el gran problema que tienen las grandes ciudades o establecimientos comerciales, es la inmensa cantidad de vehículos que circulan a diario, provocando una congestión vehicular y pérdida de tiempo al momento de buscar plazas de estacionamiento disponibles, causando retrasos a las personas.

El desarrollo de una aplicación móvil que integre el aprendizaje profundo ayuda a solucionar problemas que normalmente se encuentran en la vida cotidiana de una manera automatizada, como

el buscar plazas de estacionamiento disponibles. Se pretende mediante un sistema de monitoreo de parqueo ir solventando estas dificultades que se presentan.

La presente investigación nace de las dificultades que se evidencian en la actualidad, sobre los problemas de tráfico y detección de plazas de estacionamiento disponibles y ocupadas. Esto despierta el interés y el deseo de contribuir a solucionar dichos problemas, donde los principales beneficiarios será la comunidad universitaria de la FIC y personas en particular que ingresen a la institución educativa. El alcance de la propuesta se podría generalizar a otros lugares.

vi. Organización del documento

El presente documento se encuentra dividido en varias secciones que conforman la estructura del trabajo de titulación, como se detallan a continuación:

Capítulo I: En este capítulo se realizan los antecedentes de la investigación, teóricos y contextuales relacionados al tema del trabajo.

Capítulo II: Se describe el desarrollo del prototipo como su definición, metodologías y herramientas utilizadas.

Capítulo III: Se evalúa el prototipo con el plan de evaluación y se analizan los resultados del mismo para realizar las conclusiones y recomendaciones.

CAPÍTULO I. MARCO TEÓRICO

1.1. Antecedentes de la Investigación

Para realizar una correcta revisión sistemática, se usó la Metodología de Revisión Sistemática de la Literatura (SRL: Systematic Review of the Literature). La metodología SRL identifica, analiza e interpreta todas las evidencias disponibles en base a las preguntas de investigación, palabras clave o cadena de búsqueda, criterios de inclusión, criterio de exclusión, proceso y resultado de búsqueda [5]. A continuación, se presenta la aplicación de la metodología SRL:

a) Preguntas de investigación

En la Tabla 3, se establecieron las siguientes preguntas para la búsqueda de información sobre el tema de Aprendizaje Profundo y su aplicación en estacionamientos de parqueos.

Tabla 3. Preguntas de Investigación SRL

Pregunta de investigación	Descripción y motivación
RQ1. ¿En el ámbito tecnológico, cuáles serían los beneficios que ha traído a la sociedad el implementar Aprendizaje Profundo?	El objetivo de esta pregunta es identificar qué beneficios ofrece a la sociedad en general el utilizar el Aprendizaje Profundo, si su impacto de aceptación es bueno o malo.
RQ2. ¿Cuáles son las técnicas que se utilizan para entrenar un modelo de red neuronal?	El propósito de esta pregunta es identificar y seleccionar la mejor técnica para el entrenamiento de un modelo.
RQ3. ¿Qué impacto tiene implementar una aplicación móvil que integre aprendizaje profundo para detectar plazas de estacionamientos?	Esta pregunta pretende detectar el impacto que tendría al desarrollar una aplicación móvil que integre aprendizaje profundo para lograr detectar espacios de estacionamiento disponibles y ocupados, mediante el procesamiento de videos.

b) Palabras claves y Cadena(s) de búsqueda

Para obtener la información se utilizaron algunas estrategias de búsquedas, tomando en cuenta varias bases de datos y repositorios científicos, los mismos que fueron útiles para obtener artículos científicos de revistas conocidas, que cumplan con todos los requisitos necesarios. A continuación, se muestran algunas de las bases de datos de búsqueda científica:

- IEEE Xplore
- Science Direct
- ACM Digital Library
- SpringerLink
- MDPI

Después de varias revisiones en las bases de datos de búsqueda científica, se definió la siguiente cadena de búsqueda que permitirá buscar por títulos, palabras claves, resumen y texto completos, realizando la búsqueda en el idioma inglés y utilizando los operadores AND y OR.

Inglés:

- ("Deep Learning" OR "Machine Learning" OR "Convolutional Networks") AND ("Parking lots" OR "Parking space detection" OR "Car parking" OR "Parking control")

c) Criterios de inclusión y exclusión

Para los criterios de inclusión se tomaron en cuenta algunos artículos de revistas y algunas fuentes secundarias como libros, tesis, conferencias, páginas web oficiales, que detallan la definición de Aprendizaje Profundo, sus tipos de entrenamiento, tipos de redes neuronales, desarrollo e implementación de soluciones que involucren la detección de plazas de estacionamiento y el

impacto en la sociedad actual, además estos deben cumplir un rango de tiempo de 5 años (2017-2021) desde su publicación.

Dentro de los criterios de exclusión se consideraron publicaciones como artículos cortos menos de 3 páginas, artículos que no pertenezcan a revistas científicas, que no estén dentro del rango de los 5 años, artículos con estudios duplicados o redundantes y que no tengan relación con los estudios de investigación de detección de plazas de estacionamiento utilizando el aprendizaje profundo.

d) Proceso y resultados de la búsqueda

Para el desarrollo del proceso de búsqueda SRL, se indagó en diferentes bases de datos de búsqueda científica, además de seguir los siguientes pasos:

- Investigar con la cadena de búsqueda establecida.
- Filtrar los artículos de revistas.

En la Tabla 4, se muestra la cantidad de artículos, que fueron encontrados tanto en el idioma español e inglés.

Tabla 4. Distribución de búsqueda por fuentes

Fuentes	Artículos encontrados	Filtrado por título	Filtrado por resumen	Filtrado por palabras claves	% por fuente
IEEE Xplore	58	40	11	8	32%
Science Direct	900	129	15	4	16%
ACM Digital Library	4840	82	10	1	4%
SpringerLink	6163	35	13	4	16%
MDPI	81	13	10	6	24%
Pubmed	24	5	3	1	4%
Nature	8	3	2	1	4%
TOTAL	12074	307	64	25	100%

Utilizando el motor de búsqueda de Google Académico, en la Figura 2, se realizó un gráfico de barras, donde se detalla el número de publicaciones buscadas por palabras claves.

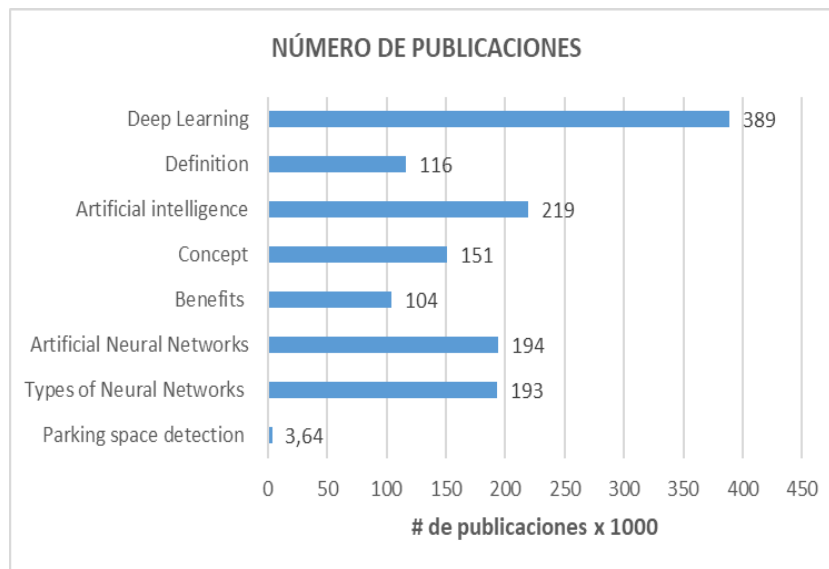


Figura 2. Cantidad de trabajos buscados por tópico – Google Académico

Con la ayuda de Scopus se procedió a realizar un análisis de documentos por años entre el rango de 2015-2022, cuyos resultados se presentan en las Figuras 3 y 4.

En la Figura 3, se visualiza un gráfico de líneas que presenta la cantidad de documentos por años correspondientes al rango de 2015-2022; mostrando como resultado que, durante el año 2019, 2020 y 2021 existe una gran cantidad de trabajos publicados.

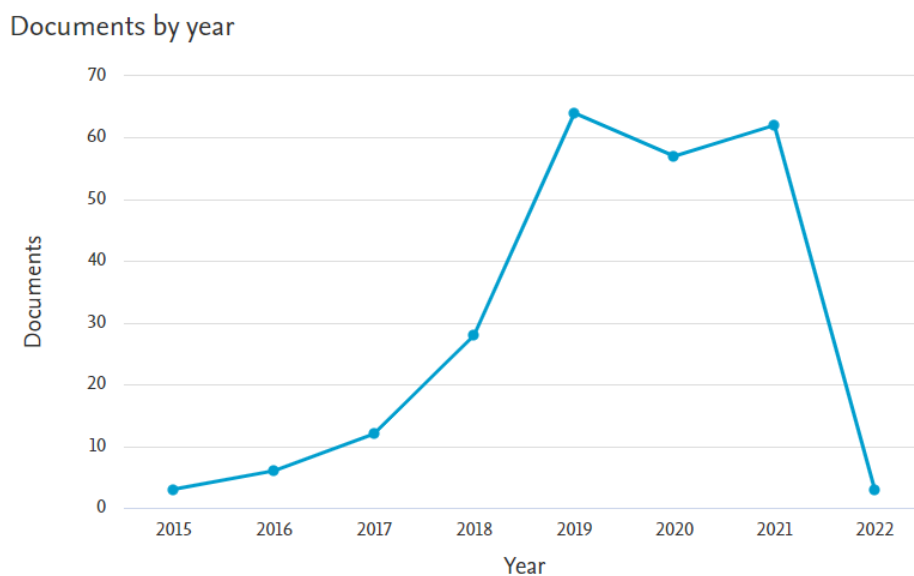


Figura 3. Cantidad de artículos buscados por año – Scopus

En la Figura 4, se visualiza los documentos por autores. Fleyeh y Paidi son los autores con más documentos, estos resultados corresponden al rango de 2015-2022.

Documents by author

Compare the document counts for up to 15 authors.

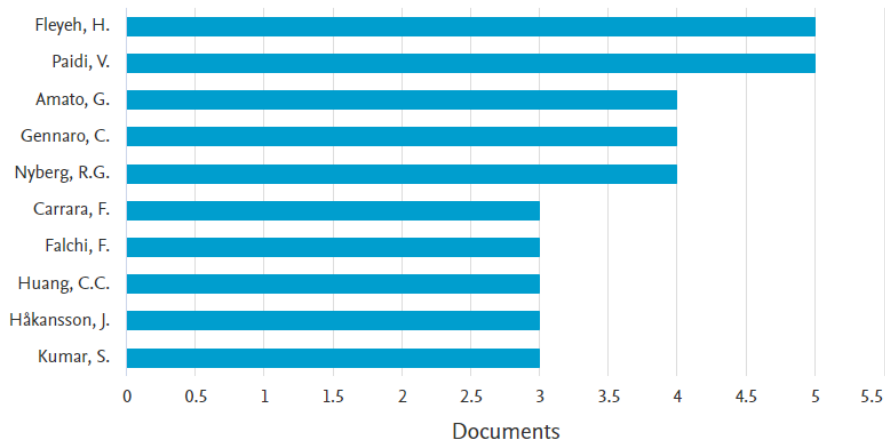


Figura 4. Cantidad de documentos por autor – Scopus

Utilizando la herramienta VOSviewer se pudo realizar el análisis bibliométrico usando palabras claves relacionadas al tema de titulación. El resultado se lo evidencia en la Figura 5, donde se observa que el término Deep Learning, red neuronal convolucional y otros, se encuentran con más frecuencia.

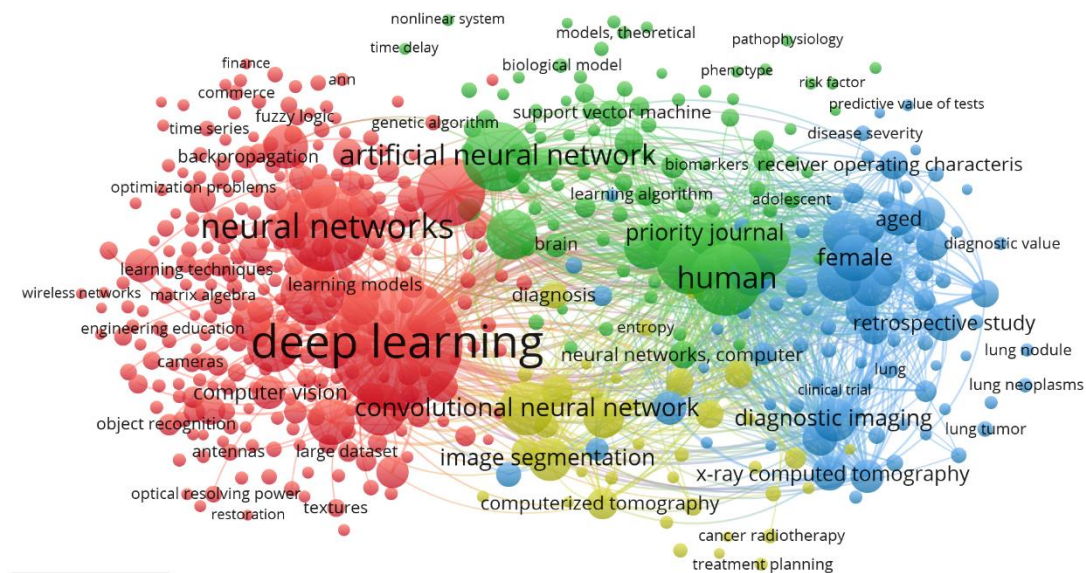


Figura 5. Análisis por palabras claves – VOSviewer

1.2. Antecedentes Teóricos

1.2.1. Trabajos Relacionados

Para esta sección, se abordó realizar una investigación de trabajos de algunos autores con similares temas, analizando el título y conclusiones. A continuación, se presentan algunos de estos trabajos:

En [6], lograron identificar las plazas de estacionamiento de vehículos disponibles y ocupadas utilizando un algoritmo de visión artificial y uso de regiones de interés. Al finalizar tuvieron como resultado de la evaluación cuantitativa que realizaron, que las plazas disponibles y ocupadas pueden ser identificadas con un 92% de exactitud, la sensibilidad con un 79% y por último la precisión con un 80%, dando como conclusión que el algoritmo desarrollado basado en visión artificial tiene un buen desempeño para lograr identificar y segmentar plazas de estacionamiento vehicular disponibles y ocupadas.

En el trabajo de Rosales [7], plantea el desarrollo de un parqueadero inteligente usando sensores electrónicos como ultrasónicos que una vez conectados comienzan a transmitir la información a través de internet para varios usuarios. Esto permite que los dueños de los vehículos puedan tomar una decisión antes de dirigirse al estacionamiento vehicular, verificando si existe un lugar libre.

En [8], presentan un prototipo que permite detectar los espacios de estacionamiento vehicular disponibles, utilizando la plataforma Android OS, entre otros.

En el artículo de Rodríguez, et al. [9], lograron desarrollar un proyecto de estacionamiento inteligente evaluando la calidad y asertividad. Por lo cual, también desarrollaron un prototipo para evaluar la fiabilidad del estacionamiento inteligente. Utilizaron dos espacios de estacionamiento y algunos elementos como: sensores de proximidad LM393, dos diodos Led's de color verde y rojo, vehículos a escala pequeña, todo esto conectado a un Arduino y aun módulo Wifi.

En la investigación elaborada por Amato, et al. [10], presentaron un enfoque para la detección de espacios ocupados en un estacionamiento vehicular en streaming, utilizando un modelo de red neuronal convolucional (CNN), ejecutado junto a una cámara inteligente. Tuvieron como resultado, que la técnica empleada es eficaz y robusta a cambios por condiciones de clima, luz y presencia de sombras.

En el artículo de Gupta, et al. [11], desarrollaron un sistema de detección y reconocimiento de vehículos militares en tiempo real como: camiones militares, tanques militares, aviones militares, helicópteros militares, automóviles civiles y aviones civiles, con ayuda de un vehículo aéreo no tripulado integrando un algoritmo inteligente de detección de objetos. El entrenamiento del modelo lo realizaron con aprendizaje profundo Quantize SSD Mobilenet v2 y Tiny Yolo v3, utilizando un conjunto de datos propias de 6772 imágenes clasificadas por clases.

Para tener una idea más clara para elaborar el marco teórico, se realizó un Mapa Temático, según la Figura 6, sobre los diversos temas relacionados con este trabajo de titulación.

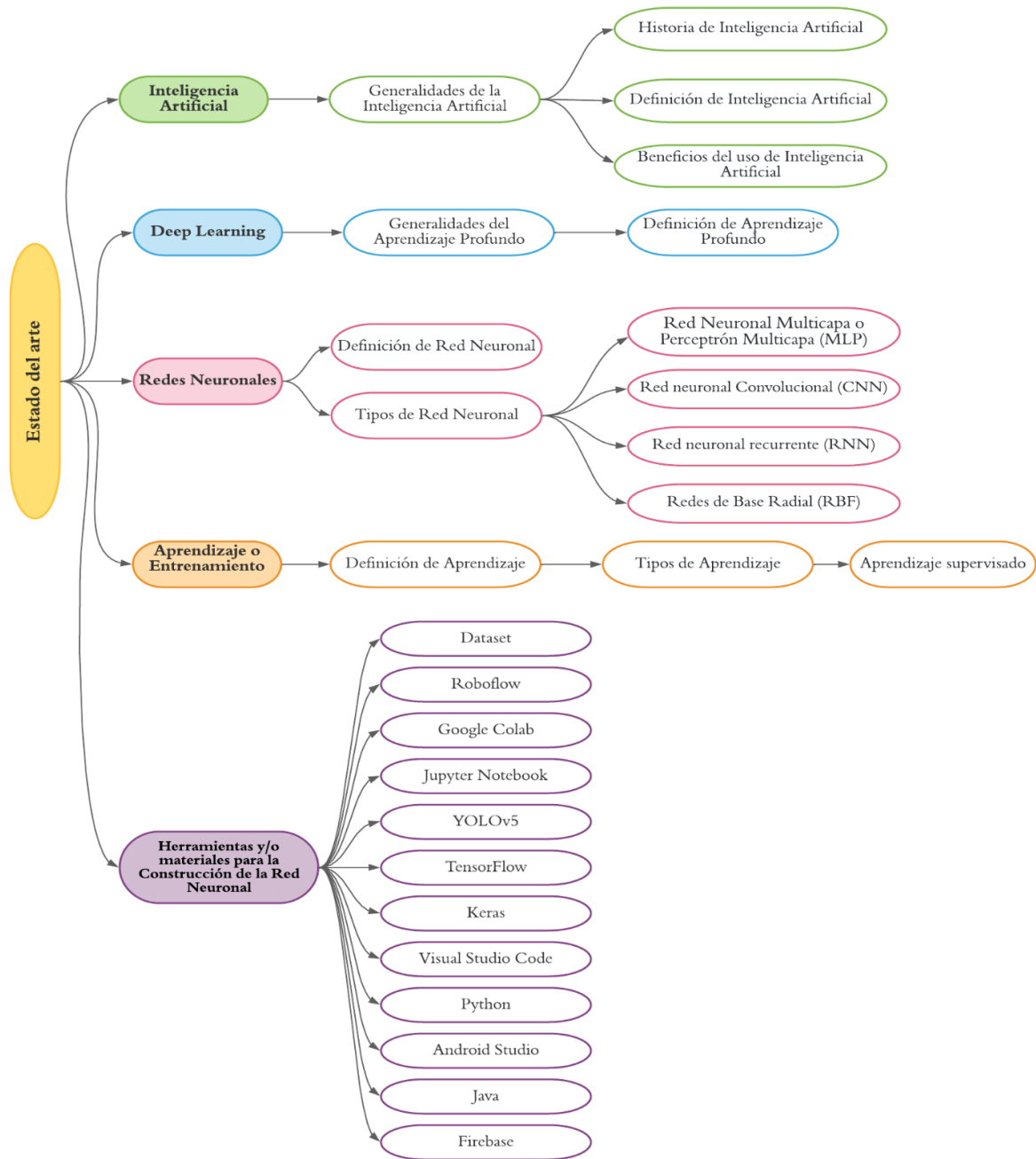


Figura 6. Mapa temático de Revisión Sistemática de Literatura

1.2.2. Generalidades de Inteligencia Artificial

1.2.2.1. Historia de Inteligencia Artificial

La inteligencia artificial surge de manera definitiva en 1950 con Alan Turing quien es considerado como el padre de la inteligencia artificial, gracias a la publicación de su artículo, se obtienen las interrogantes de un método que permita comprobar si una máquina puede tener pensamientos. Algunos de los fundamentos de la AI se encuentran en los experimentos que se exponen en su

artículo y que luego pasaron a denominarse Test de Turing. A lo largo de los años desde los inicios de su aparición hasta la actualidad la inteligencia artificial ha ido evolucionando e incluso ha hecho aportes significativos en la sociedad, como en la rama de la medicina, demostrando mejora en la precisión diagnóstica de enfermedades y la eficiencia del flujo de trabajo [12].

1.2.2.2. Definición de Inteligencia Artificial

Artificial Intelligence (AI) o en español conocido como Inteligencia Artificial (IA), es una mezcla de algunos algoritmos que tienen como finalidad recrear las mismas habilidades y capacidades que tiene un ser humano. Con el pasar de los años esta tecnología ha dejado de ser una ciencia ficción convirtiéndose en algo real y cotidiano para todos.

En el artículo [13], mencionan que la Inteligencia Artificial (IA) se ha convertido en un factor esencial para la economía de varios países y debido a su gran efecto positivo en la sociedad este se está adaptado y quedando para siempre en la vida diaria de las personas. Esta tecnología ha permitido alcanzar grandes avances, en el área espacial, exploración del universo, comunicación espacial, navegación, entre otros aspectos.

1.2.2.3. Beneficios del uso de Inteligencia Artificial

Inteligencia Artificial (IA), es una tecnología que funciona con varios algoritmos que actúan a partir de reglas y fundamentos de programación.

En el trabajo [14], se demuestra que las Redes Neuronales Artificiales (RNA) funcionan de manera excelente cuando se tratan de sistemas complejos y con una gran variedad de datos y que al aplicar Redes Neuronales permite encontrar relaciones entre los patrones de entrada y salida, lo que resulta difícil visualizarlo con enfoques estadísticos. IA ayuda mucho en los procesos de automatización, potenciar tareas creativas, reducir el error humano, dar un mantenimiento predictivo, incluso en ciudades inteligentes jugando un papel activo en la transformación de las infraestructuras permitiendo crear beneficios económicos [15].

1.2.3. Generalidades del Aprendizaje Profundo

1.2.3.1. Definición de Aprendizaje Profundo

En el artículo de Amato, et al. [16], destacan que el Aprendizaje Profundo o Deep Learning (DL), es una disciplina de la Inteligencia Artificial, que imita la forma en que los humanos adquieren los

conocimientos. El objetivo principal es desarrollar técnicas que permitan a las máquinas aprender tareas de percepción complejas, tales como ver y oír, las mismas que estén al nivel de los humanos.

El aprendizaje profundo crea modelos de forma jerárquica más complejas, que los utilizados en el Aprendizaje automático, debido a que el DL busca imitar el modo en los que los humanos aprenden, o las precisiones que tienen para clasificar imágenes, detectar objetos, realizar reconocimiento de voz, entre otras.

El DL es uno de los elementos importantes dentro de la ciencia de datos, mismos que incluyen estadísticas y los modelos predictivos, ya que tiene la tarea de recopilar, analizar e interpretar la información en grandes cantidades de datos, haciendo que este procedimiento sea más sencillo y rápido.

1.2.4. Redes Neuronales

1.2.4.1. Definición de Red Neuronal

En [17], afirman que las Redes Neuronales Artificiales (RNA) están inspiradas en el funcionamiento de las redes de neuronas biológicas del cerebro humano. Tienen la capacidad de realizar los mismos procesos y de ir modificando su estructura dependiendo del objetivo que vaya a realizar.

Las RNA, son un conjunto de neuronas artificiales conectadas entre sí y que se encuentran agrupadas en niveles, los mismo que son denominados capas. Una RNA está formada por elementos bases como los nodos que también son llamados elementos de procesamientos, debido a que ahí es donde se procesa la información y obtiene una respuesta en base a lo solicitado, y su otro elemento son las conexiones. Cada nodo de la red posee su entrada, a partir de las comunicaciones que recibe de otros nodos y su salida, es con la que se comunica con otros nodos o con el entorno. Finalmente, cada nodo tiene la función de transformar sus propias entradas en salidas.

1.2.4.2. Tipos de Red Neuronal

Las redes neuronales son algoritmos que se basan en simular el comportamiento del cerebro humano. Existen varios tipos de redes neuronales, como son los siguientes:

Red Neuronal Multicapa o Perceptrón Multicapa (MLP)

MLP es una red neuronal denominada multicapa, es una red neuronal conocida que está formada por tres capas, una capa de entrada, una o varias capas ocultas y una capa de salida a diferencia de la red neuronal monocapa que solo cuenta con 2 capas [18]. La capa oculta está formada por las neuronas de las entradas que vienen de capas anteriores y las salidas pasan a las capas posteriores. Esta red puede ser usada para preprocesar diferentes datos, por ejemplo, el estado de un usuario para predecir si este se encuentra dentro o fuera del vehículo. [19].

Red neuronal Convolutacional (CNN)

En [20], una CNN es una red que está compuesta por una capa de entrada, varias capas ocultas y una capa de salida. También informa que al momento de utilizar este tipo de red neuronal no es necesario usar una CNN clásica que requiera muchas muestras y que con un conjunto de datos de 7000 es suficiente para entrenar un modelo. Las salidas finales del CNN son las clases que se han entrenado en la red neuronal.

Las neuronas convolucionales aprenden a reconocer diversos objetos dentro de las imágenes, para lograr conseguir eso se necesita de un entrenamiento, el cual debe contar con una cantidad grandes de muestras que pueden ser imágenes, mientras más grande sea el tamaño de la data mayor será la precisión del entrenamiento.

Red neuronal recurrente (RNN)

En el artículo de Badii, Nesi y Paoli [21], mencionan que una RNN es un modelo eficaz para datos secuenciales que son aplicados en secuencias de tiempo y datos ordenados. Una RNN no está estructurada por capas, esta red tiene conexiones que son arbitrarias entre neuronas, incluso crean ciclos y así consiguen la temporalidad, permitiendo que la red tenga memoria y opere en secuencia de variables.

Las RNN son potencialmente usadas en aquellas cosas que tienen que ver con el análisis secuencial, por ejemplo, para analizar textos, sonidos o videos, es decir, analizan datos de series temporales, teniendo en cuenta al tiempo, elemento que no ha sido considerado en las redes neuronales anteriores.

Redes de Base Radial (RBF)

Una red de Base Radial está compuesta por tres capas, la primera capa de entrada, la segunda capa oculta, que además solo cuenta con solo una capa. Y la tercera capa, la de salida que es la que responde a los patrones de entrada.

Este tipo de red se caracteriza por tener un aprendizaje híbrido. Su arquitectura es similar a la MLP, pero la diferencia radica en la capa oculta, debido a que está calcula la salida en función de la distancia a un punto llamado centro. RBF puede ser aplicado en el análisis de series temporales, procesamiento de imágenes, reconocimiento del habla, etc. En [22], utilizaron una red de Base radial para predecir la trayectoria de los vehículos, en base a un carril objetivo y así prevenir las posibles colisiones que puedan existir.

1.2.5. Aprendizaje o Entrenamiento

1.2.5.1. Definición de Aprendizaje

El término aprendizaje, es un proceso que permite adquirir habilidades, conductas, conocimientos y valores. Esto es algo presente en los seres vivos, pero con el avance tecnológico, las máquinas también pueden imitar el razonamiento y aprendizaje humano.

En 2019 Sakhare, Tewari y Vyas analizaron el desarrollo de algoritmos en tiempo real, para la detección de vehículos, donde se dieron cuenta que las imágenes deben ser tomas aéreas y el procesamiento de la información se lo puede realizar con algoritmos de aprendizaje profundo y procesamientos de imágenes [23]. Además, el desarrollo de algoritmos en tiempo real ayuda en la actividad para el reconocimiento de patrones, visión 3D, aprendizaje, entre otros.

En [24], mencionan que el mejor algoritmo de aprendizaje con relación a la velocidad es LR, aunque a veces en los casos presenta un rendimiento malo. Por otra parte, en el artículo Deep Learning realizado por LeCun, Bengio y Hinton [25], explican que después del entrenamiento de una red neuronal, el rendimiento del sistema se lo mide por conjuntos de prueba. Esto ayuda a experimentar la capacidad de la máquina para producir respuestas de nuevas entradas que nunca ha considerado durante el entrenamiento.

1.2.5.2. Tipos de Aprendizaje

El aporte de Xin, et al. [26], determina que:

- El aprendizaje no supervisado, es un tipo de aprendizaje automático que permite deducir las estructuras de los datos sin etiquetar. Por lo tanto, al no estar etiquetado dificulta evaluar la precisión de salida del algoritmo.
- El aprendizaje supervisado tiene la capacidad de analizar todos los datos que fueron entrenados con anterioridad y usarlos para mapear nuevas instancias.

Aprendizaje supervisado

Este tipo de aprendizaje es una técnica que permite deducir una función a partir del entrenamiento de datos. Estos datos consisten en dos partes, los datos de entrada y los datos de salida, es decir, los resultados deseados.

El artículo [27], indica que, para realizar el aprendizaje supervisado, las imágenes se deben anotar según la clase. Sin embargo, a veces se requiere aplicar métodos para establecer aleatoriamente las posiciones de las esquinas de las imágenes, debido a la falta de muestras en los datos originales.

1.2.6. Herramientas y/o materiales para la Construcción de la Red Neuronal

1.2.6.1. DataSet

El término dataset, se puede definir como una colección o un conjunto de datos almacenados en memoria junto a un modelo de programación relacional, sin importar el origen de los datos que lo contiene.

En la propuesta del artículo [28], analizan el método de detección de espacios de estacionamiento vehicular, utilizando un conjunto de datos entrenados en múltiples detectores que les permitieron determinar un lugar libre.

Un dataset, tiene la función de usarse como un modelo de regresión para la entrada de datos y con eso predecir variables. Usualmente es usado para el análisis de datos en Big Data, Inteligencia Artificial y las diversas ramas como Machine Learning y Deep Learning.

1.2.6.2. Roboflow

Roboflow es una herramienta que evalúa la calidad del conjunto de datos, preparando las imágenes previamente etiquetadas para entrenar un modelo de red neuronal, en donde se puede definir el porcentaje de imágenes a utilizar tanto para el entrenamiento, validación y prueba. Utiliza los formatos de anotaciones más comunes como es el TXT, CSV, JSON Y XML [29].

1.2.6.3. Google Colab

Google Colab, es una plataforma online, usada para el aprendizaje automático y el análisis de datos que proporciona Google Research [30]. También puede ser usado para el aprendizaje profundo o aplicaciones científicas que requieran GPU [31].

Esta plataforma permite a cualquier usuario escribir sentencias de Python y también a ejecutar el código en el navegador de su preferencia. Es un servicio alojado de Jupyter Notebook que no requiere de una configuración y que tiene acceso gratuito a recursos informáticos como: GPUs [32].

1.2.6.4. Jupyter notebook

La aplicación Jupyter notebook, es un cuaderno de laboratorio virtual que permite mejorar la reproducibilidad, reutilización de datos, crear visualizaciones, ejecutar códigos como mecanismos de citación de datos o software, entre otros [33]. En otras palabras, es una aplicación que funciona como cliente-servidor, fundada en el año 2015. Esta aplicación permite crear y compartir documentos en formato JSON.

1.2.6.5. YOLOv5

You only look once (YOLO), es un algoritmo de detección de objetos previamente entrenados en COCO, que es un conjunto de datos que divide las imágenes en cuadrícula, esta tecnología es capaz de procesar vídeos en tiempo real en menos de 25 segundos. Durante el entrenamiento YOLO toma la imagen completa para la entrada de la red, luego la coloca en posición de un cuadro delimitador el cual contiene características específicas de la imagen, una vez que tiene las categorías estas pasan a ser parte de la salida de la red [34][35]. YOLO fue desarrollado por Glenn Jocher y ha tenido varias versiones desde YOLOv1 hasta YOLOv7.

1.2.6.6. TensorFlow

TensorFlow es una biblioteca open source, usada para el aprendizaje automático, capaz de entrenar redes neuronales que puedan realizar diferentes actividades similares a las del cerebro humano. Es usada para enseñar a desarrollar y entrenar modelos de Aprendizaje automática (AA). Cuenta con diferentes herramientas, bibliotecas y recursos, hace que la creación e implementación de aplicaciones con tecnología AA sea fácil [36].

1.2.6.7. Keras

Keras es un framework usado para el aprendizaje de Redes neuronales escrita en Python, es de código abierto. El funcionamiento de Keras no es el trabajar como un framework independiente, si no como una interfaz de uso intuitivo (API), por tal motivo es capaz de ejecutar sobre TensorFlow y así trabajar en conjunto, con el fin de facilitar la experiencia de entrenar modelos que permitan clasificar imágenes [37].

1.2.6.8. Visual Studio Code

Visual Studio Code (VS Code), es un editor de código fuente ligero y potente que se ejecuta en un sistema operativo como, Windows, macOS y Linux. Tiene soporte integrado para JavaScript, Node.js y TypeScript. Además, permite realizar codificación en lenguajes como C#, Java, Python, entre otros [38].

1.2.6.9. Python

Python es un lenguaje de programación que proporciona un entorno de desarrollo adecuado para proyectos computacionales como de visualización. Es multiparadigma, es decir, soporta parcialmente la programación orientada a objetos, programación imperativa. Una de las ventajas de este lenguaje, es que permite a los programadores usar diferentes paquetes haciéndola funcional [39].

1.2.6.10. Android Studio

Android Studio, es un entorno de desarrollo integrado conocido como IDE. Fue creado para el desarrollo de aplicaciones móviles de Sistemas Operativos Android basado en IntelliJ IDEA. Ofrece muchas funciones para aumentar la productividad al momento de desarrollar apps, además tiene compatibilidad con C++ y NDK, entre otros [40].

1.2.6.11. Java

Java es un lenguaje de programación de alto nivel, fue comercializado en 1995 por Sun Microsystems. Es usado para el desarrollo de aplicaciones tanto de escritorio como móviles, es rápido, seguro y fiable, además existen aplicaciones y sitios web que su funcionamiento depende de tener Java instalado. Java tiene un enfoque a la programación orientado a objetos, siendo estos uno de los estilos más populares de la programación, permitiendo así crear aplicaciones que se ejecuten en cualquier dispositivo [41].

1.2.6.12. Firebase

Firestore es una herramienta que combina varios servicios de la empresa Google en la nube. Esta herramienta contiene mensajería instantánea, autenticación de usuario, base de datos en tiempo real, entre otros [42].

En la página oficial de Firebase [43], informa que las funciones proporcionadas por esta herramienta online permiten programar apps de alta calidad, mejorar la calidad de las apps, hacer crecer el negocio enfocándose más en los usuarios, ganar dinero, etc., además este sistema de Firebase proporciona transmisión de datos con cifrado SSL.

1.3. Antecedentes Contextuales

La Universidad Técnica de Machala (UTMACH), es una Institución de educación superior, que tiene como misión formar y perfeccionar a profesionales en diversas áreas del conocimiento, haciéndolos competentes y emprendedores [44].

En la Facultad de Ingeniería Civil de la UTMACH, existe un total de 1187 estudiantes matriculados, 45 docentes, 12 personal administrativo y servicio, el cual, se estima que un porcentaje considerado tienen vehículos que lo utilizan para movilizarse a la Institución Educativa. Actualmente la FIC cuenta con aproximadamente 40 plazas de estacionamiento que son ocupadas a diario.

Este trabajo de titulación, está centrado en desarrollar una aplicación móvil para la detección de plazas de estacionamientos disponibles y ocupadas mediante el procesamiento del video en tiempo real, utilizando Aprendizaje Profundo en los parqueaderos de la Facultad de Ingeniería Civil de la UTMACH.

1.3.1. Ámbito de aplicación

Es evidente que la abundante presencia de vehículos, es un problema que se presenta todos los días al momento de querer un espacio libre para estacionarse, producto de la creciente demanda de automotores. En ciertos lugares o establecimientos como universidades, tienen parqueaderos con espacios limitados, siendo difícil para las personas encontrar una plaza de estacionamiento que esté disponible para su uso.

Por tal motivo, se desarrolló una aplicación móvil que permite detectar espacios de estacionamiento de una manera automatizada. El lugar seleccionado fue la Facultad de Ingeniería Civil de la UTMACH, teniendo como punto principal la detección de plazas de estacionamientos en el parqueadero.

1.3.2. Establecimiento de requerimientos

El desarrollo de la aplicación móvil para la detección de plazas de estacionamiento requiere:

- Un modelo de red neuronal convolucional entrenado con un dataset de imágenes clasificadas para el entrenamiento y valoración (estacionamiento disponibles y ocupadas).
- Una función para abrir la cámara y visualizar en tiempo real el parqueadero de la Facultad de Ingeniería Civil de la UTMACH, mostrando los espacios que estén disponibles y ocupados.

CAPÍTULO II. DESARROLLO DEL PROTOTIPO

2.1. Definición del prototipo

La definición del prototipo muestra las arquitecturas que se emplearon para la realización del trabajo de titulación, la misma que cuenta de dos etapas, la primera abarca el entrenamiento del modelo y la segunda el procesamiento del video dentro de la aplicación móvil. A continuación, se detallan cada una de las etapas.

En la Figura 7, se plantea la arquitectura del entrenamiento del modelo de la red neuronal convolucional. Como primer punto se realizó la recolección y selección de imágenes, las mismas que fueron tomadas del estacionamiento de la FIC. Luego se creó el dataset de imágenes utilizando la herramienta LabelImg que nos ayudó a etiquetar las imágenes en 2 clases (libre y ocupado). Al finalizar el etiquetado se procedió a clasificar las imágenes en entrenamiento y validación, usando la herramienta online Roboflow. Posteriormente se realizó el entrenamiento del modelo utilizando el lenguaje Python y YOLOv5 en el entorno de Google Colab. Como resultado del entrenamiento se obtiene el modelo para detectar plazas de estacionamiento en formato pt.

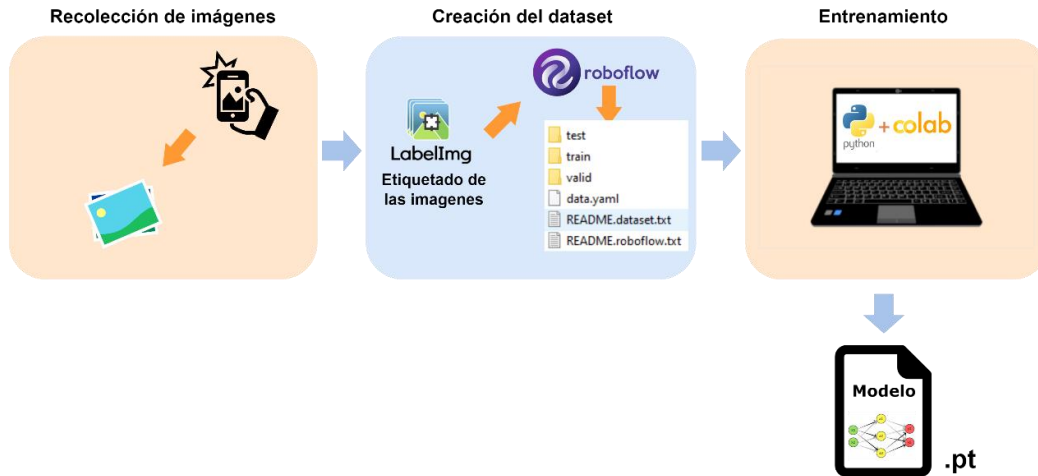


Figura 7: Arquitectura de entrenamiento de la Red Neuronal

En la Figura 8, se puede visualizar el proceso que se realiza al momento de detectar las plazas de estacionamientos, teniendo como primer punto, la entrada de datos mediante una cámara IP conectada en tiempo real (Streaming). Como segundo punto se encuentra el procesamiento de video y predicción del modelo, para lo cual se utilizó la herramienta Visual Studio Code y el modelo entrenado, que permite obtener un sitio web para visualizar el video de la cámara IP y consumir el modelo. La salida de datos realiza la petición al sitio web y este envía la respuesta del video de la cámara IP con la predicción. Esta información es observada desde el dispositivo celular con sistema operativo Android.



Figura 8. Arquitectura para la detección de plazas de estacionamiento

2.2. Metodología de desarrollo del prototipo

2.2.1. Enfoque, alcance y diseño de investigación

El enfoque de investigación de este trabajo fue cuantitativo, ya que se realizó una recolección de datos (imágenes) para crear y entrenar un dataset y además se utilizó la red convolucional YOLOv5, que permitió detectar plazas de estacionamiento disponibles y ocupadas.

El alcance es exploratorio y descriptivo, debido a que al principio se realizó una indagación bibliográfica para tener más información correspondiente al tema de Aprendizaje Profundo aplicados en estacionamiento de vehículos, para luego mostrar las especificaciones más importantes encontradas. El diseño de la red neuronal y la aplicación móvil fue de mayor prioridad, lo que corresponde a la latencia que se presentó al momento de procesar el video en tiempo real (Streaming) quedará como estudio a futuro para el mejoramiento del rendimiento. Además, la ejecución y pruebas finales de la aplicación móvil fueron realizadas de manera local.

El diseño de Investigación fue Cuasi-experimental porque el trabajo está destinado a un grupo de usuarios, los cuales permitieron realizar las pruebas finales de la app móvil.

2.2.2. Unidades de análisis

Población (universo)

En la Tabla 5, se especifica el total de los miembros de la Facultad de Ingeniería Civil clasificados por: estudiantes, docentes, trabajadores administrativos /servicios y personas externas que hacen uso de la institución educativa.

Tabla 5. Miembros de la FIC y personas externas

Población	
Total estudiantes	1 187
Total docentes	45
Total trabajadores administrativos / servicios	12
Estimación de personas externas	30
Total de la población	1274

Muestra

Aplicando la muestra aleatoria simple y con ayuda de la herramienta web SurveyMonkey [45], se obtuvo como resultado la muestra de 90 personas a partir de la población de 1274 perteneciente a la FIC y personas externas de la Universidad Técnica de Machala, se utilizó un nivel de confianza del 95% con un margen de error del 10%.

También se tomó en cuenta como muestra las imágenes, para la construcción del dataset y entrenamiento del modelo, teniendo un total de 1000 imágenes correspondiente al 100%. Para el entrenamiento se tomó en cuenta el 80% (800 imágenes) y para la validación el 20% (200 imágenes). A continuación, en la Tabla 6 se desglosa la información.

Tabla 6. Datos de la dataset del entrenamiento y pruebas

	Cantidad imágenes
Entrenamiento 80%	800
Validación 20%	200
Total imágenes	1 000

2.2.3. Técnicas e instrumentos de recopilación de datos

Para este trabajo, se ha detallado en la Tabla 7 la técnica e instrumentos de recopilación de datos que fueron usados.

Tabla 7. Técnica e instrumento de recopilación de datos

Técnica	Instrumento
Análisis de datos (red neuronal convolucional Yolo)	Análisis del dataset de imágenes o datos
Encuesta	Encuesta realizada con la herramienta online de Google Forms. (Anexo 2) (Anexo 3)

2.2.4. Técnicas de procesamiento de datos para la obtención de resultados

Se usó Deep Learning, con el cuál se entrenó un modelo de red neuronal convolucional, para ello se necesitó un dataset de imágenes que estén etiquetadas por clases, la primera clase fue de imágenes de estacionamientos disponibles y la otra clase de estacionamientos ocupados. El modelo fue evaluado usando la matriz de confusión y métricas derivadas (precisión, sensibilidad y valor de referencia).

Como segunda técnica de recopilación de datos, se realizó una encuesta del cumplimiento de las funcionalidades de la app, en base a la encuesta de diagnóstico inicial para la elaboración del prototipo. Estos resultados obtenidos permitieron realizar las respectivas tabulaciones.

2.2.5. Metodologías o métodos específicos

Metodología para la construcción de la Red Neuronal

Para el siguiente trabajo de titulación se planteó una metodología genérica que consta de los siguientes pasos [46]:

- Primero se creó un dataset con imágenes tomadas en algunas Facultades de la UTMACH. Para su etiquetado y clasificación se utilizaron las herramientas LabelImg y RoboFlow. Posteriormente se aplicó el Aprendizaje Profundo utilizando herramientas como: Python, YOLOv5 (Open CV, Keras, Numpy, Tensorflow, etc), y Jupyter Notebook, para finalmente proceder a entrenar el modelo ejecutándose dentro del entorno de Google Colab.
- Como segundo punto, se realizaron diversas pruebas de ejecución y testeo en el entorno de Google Colab, obteniendo las métricas de cada modelo y así lograr comparar los modelos entrenados utilizando la matriz de confusión.
- Por último, teniendo la comparación se procedió a seleccionar el modelo con más probabilidad de predicción, para ser utilizado en las pruebas finales y así detectar plazas de estacionamiento vehicular. En la Figura 9 se evidencia la secuencia de pasos para la metodología:

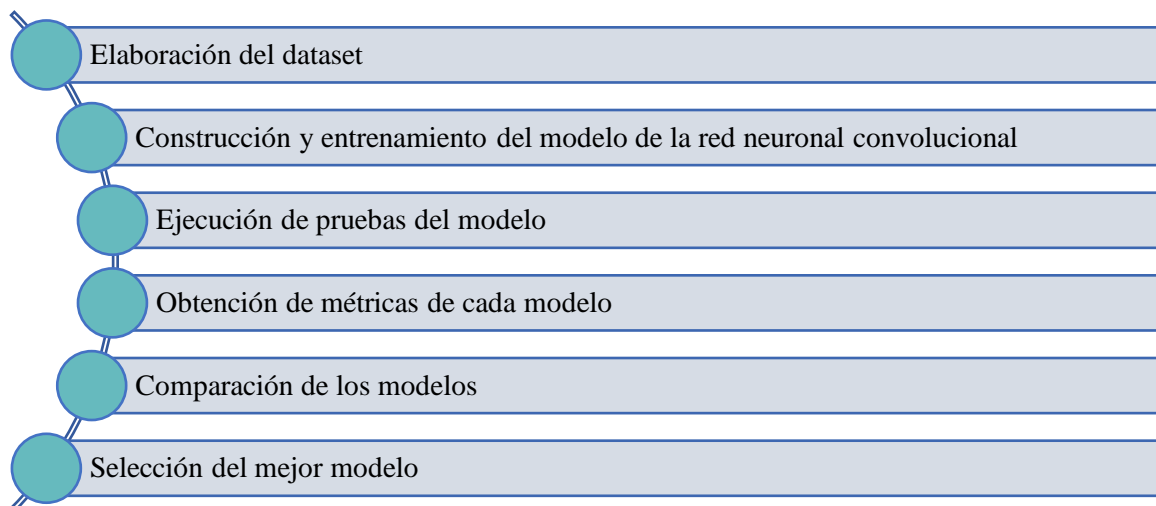


Figura 9. Metodología para la construcción de la Red Neuronal

Metodología Mobile-D para el desarrollo de la aplicación móvil

La metodología Mobile-D está pensada para ser realizada por un equipo con un máximo de diez desarrolladores de software y que esté a la vez pueda resolver las dificultades que implica desarrollar aplicaciones móviles en un corto periodo de tiempo. La metodología Mobile-D consta de cinco fases: Exploración, Inicialización, Producción, Estabilización y pruebas [47], [48].

Cada fase es fundamental para el desarrollo y control de la aplicación móvil. En la Figura 10, se detallan las fases de la metodología Mobile-D.

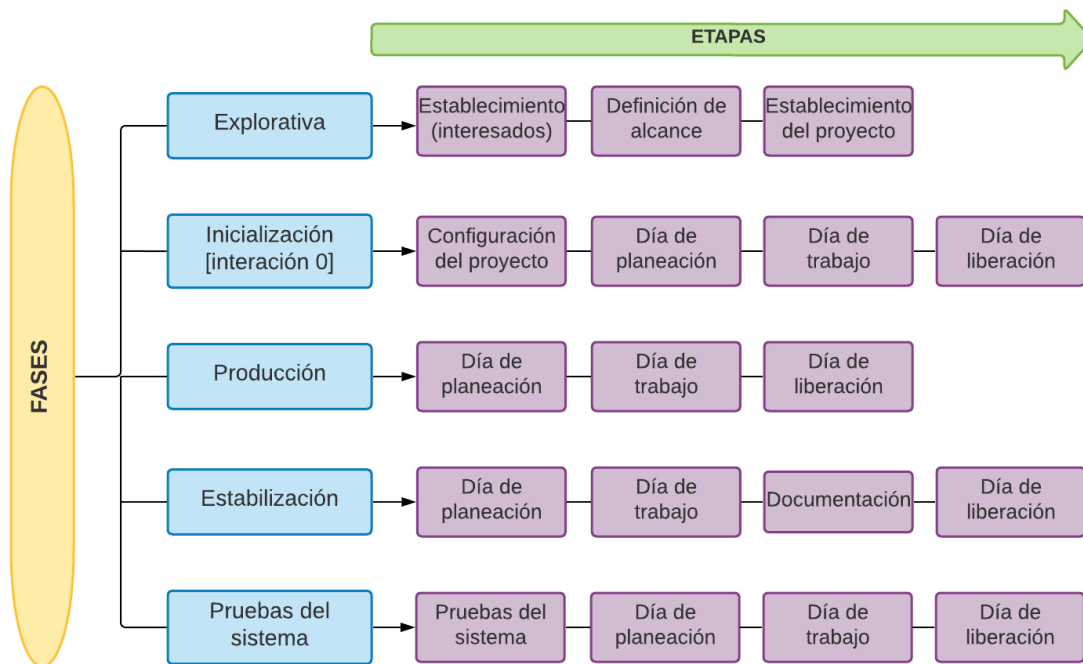


Figura 10. Fases de la metodología Mobile-D (Adaptado de la metodología Mobile-D)

- 1. Exploración:** La primera fase tiene como objetivo definir el grupo de interesados del proyecto, definir objetivos de la aplicación móvil y elegir a los miembros que participaran en el desarrollo de la misma.
- 2. Inicialización:** En la etapa de Inicialización, se realiza la configuración del proyecto y se prepara los recursos para dedicar los días de planificación, trabajo y liberación con iteración 0.
- 3. Producción:** La fase de producción, tiene como propósito implementar las funcionalidades que son requeridas en el producto. Esta fase se divide en día de planeación, día de trabajo y día de liberación.
- 4. Estabilidad:** Esta fase se encarga de integrar los módulos separando y uniendo en una única aplicación. Las etapas son día de planeación, día de trabajo, documentación, día de liberación.
- 5. Pruebas del sistema:** Finalmente la fase de pruebas del sistema, tiene como propósito verificar si el sistema cumple con las funcionalidades establecidas en las fases anteriores, además de encontrar defectos y errores para solventarlos. En este punto la aplicación debe ser estable y funcional para que los clientes puedan utilizarla.

2.2.6. Herramientas y/o Materiales

En la Tabla 8 se presentan algunas de las herramientas y/o materiales que se utilizaron en el desarrollo de la app móvil, para detectar plazas de estacionamientos disponibles y ocupadas.

Tabla 8. Herramientas y/o materiales

Clasificación	Herramientas y/o materiales
Software	<ul style="list-style-type: none">• Android Studio• TensorFlow• Keras como framework• Google colab• Jupyter Notebook• Balsamiq Mockups• Visual Studio Code• Firebase
Lenguajes	<ul style="list-style-type: none">• Python• Java
Datos	<ul style="list-style-type: none">• Dataset de imágenes
Algoritmos	<ul style="list-style-type: none">• Redes neuronales convolucionales: YOLOv5.

2.3. Desarrollo del prototipo

Para el desarrollo del prototipo se implementó dos metodologías, una para la red neuronal y la otra para la aplicación móvil.

2.3.1. Metodología para la construcción de la Red Neuronal

Para el desarrollo de la red neuronal convolucional se utilizó una metodología que consta de las siguientes fases: elaboración de la dataset, construcción y entrenamiento del modelo de red neuronal convolucional, ejecución de pruebas del modelo, obtención de métricas de cada modelo, comparación de los modelos y la selección del mejor modelo.

2.3.1.1. Elaboración del Dataset

Para este punto se recolectaron un total de 1000 imágenes tomadas en algunas Facultades de la UTMACH. Las imágenes fueron etiquetadas manualmente, utilizando la herramienta LabelImg 1.8.0. Las etiquetas fueron hechas en base a dos clases, la primera con espacios estacionamientos libres y la otra con espacios ocupados. En la Figura 11 se evidencia el proceso de etiquetado.

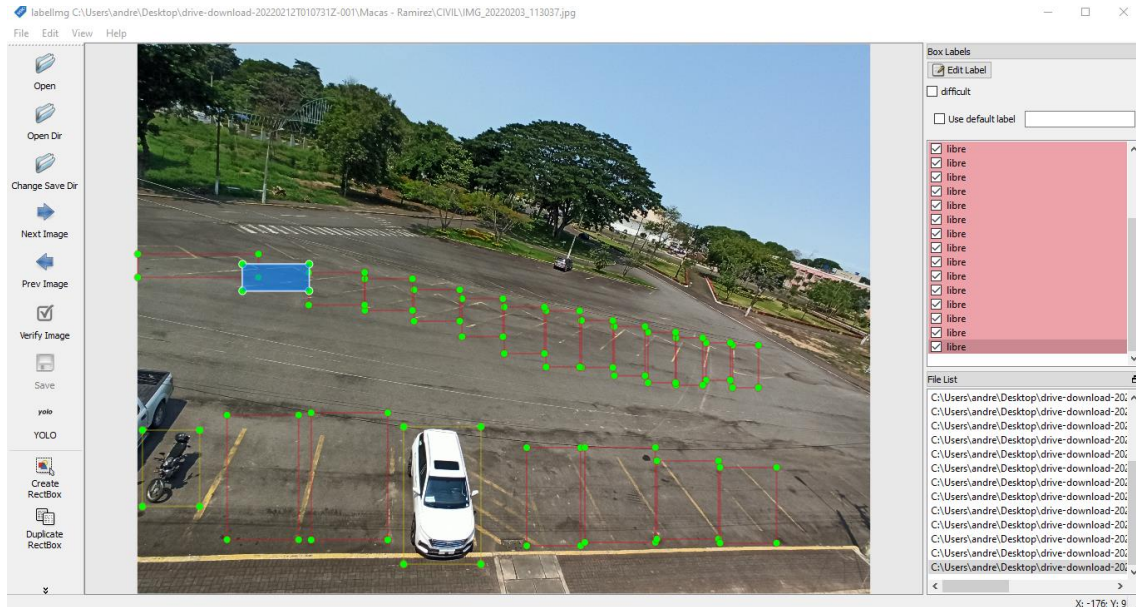


Figura 11: Software para etiquetar imágenes

Una vez que las imágenes están etiquetadas se genera un archivo con extensión txt de cada una de ellas. El archivo txt especifica la numeración de las clases de las etiquetas en 0 (libre) y 1 (ocupado); luego especifica las coordenadas de (x, y) de los cuadros que contiene los objetos.

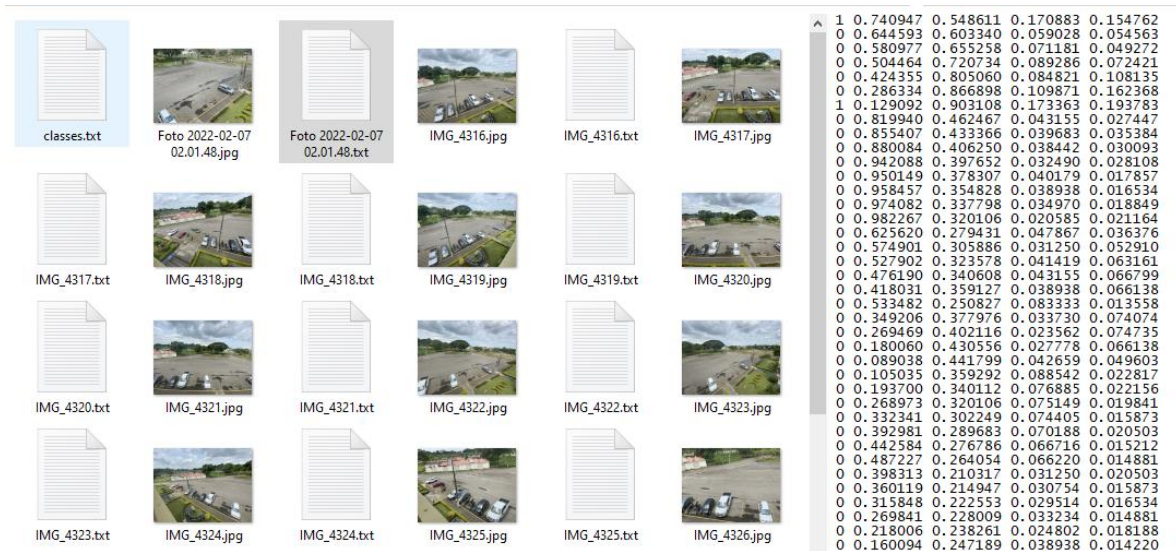


Figura 12: Archivos txt generados de las etiquetas

Se utilizó la herramienta web roboflow para simplificar el proceso de clasificación de conjuntos de datos. Roboflow divide el conjunto de datos de manera aleatoria; el 100% de las imágenes fueron divididas en 80% para el entrenamiento y 20% para validaciones. En la Figura 13, se evidencia la información detallada del dataset.

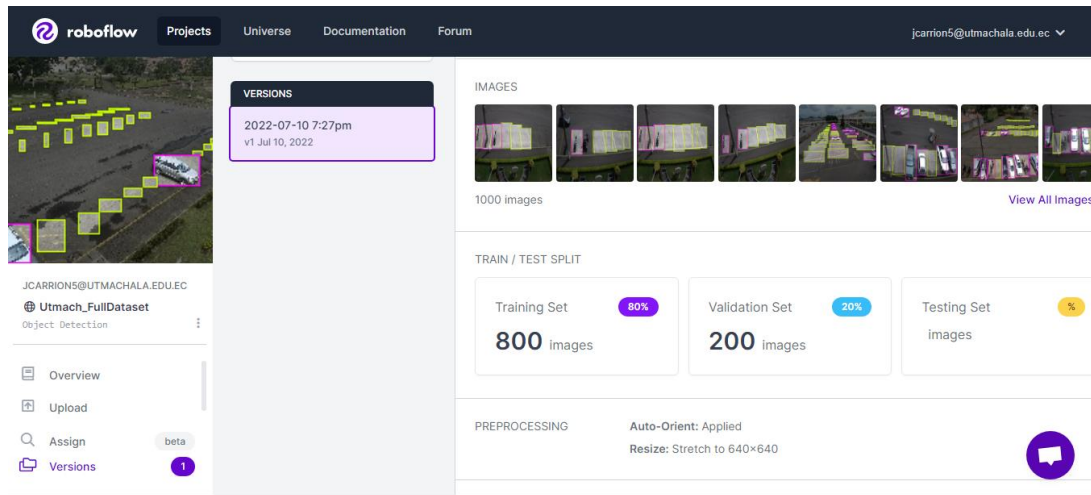


Figura 13. Creación del dataset en Roboflow

2.3.1.2. Construcción y entrenamiento de modelo de red neuronal

Para la construcción de la red se utilizó un modelo pre-entrenado de YOLOv5 y el entorno de Google Colab.

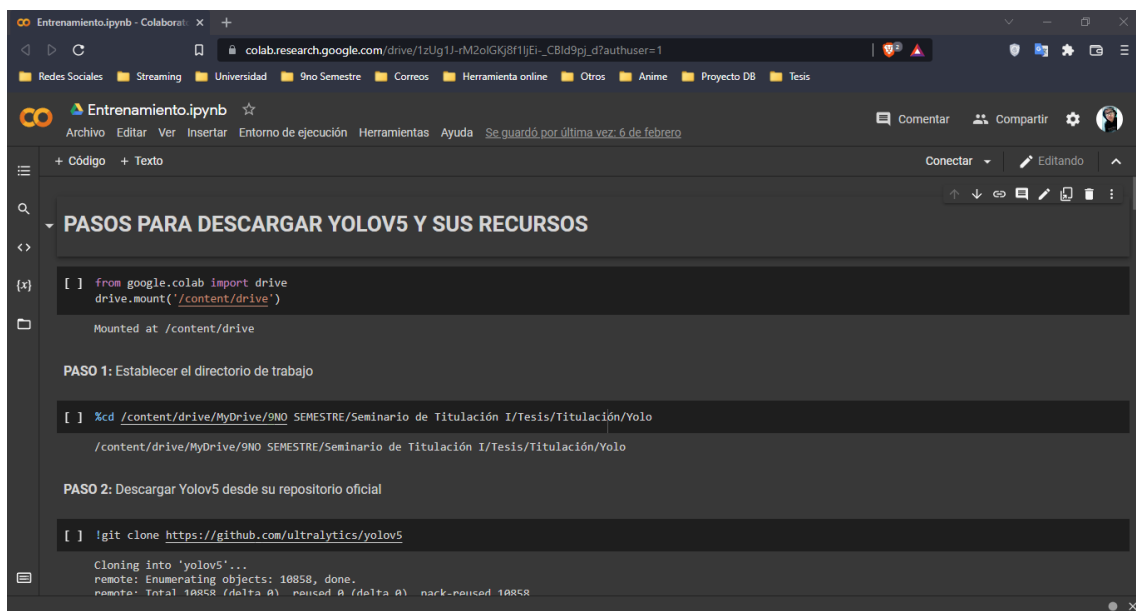


Figura 14. Diseño de la Red Neuronal

A continuación, se detallan los pasos para la construcción y el entrenamiento del modelo de la red neuronal, estos pasos fueron obtenidos del repositorio oficial de YOLOv5 [49].

Pasos para descargar YOLOv5 y sus recursos

Paso 1: Conectar con Google drive

Para comenzar con el entrenamiento de la red neuronal, en el entorno de Google Colab se debe otorgar los permisos para tener acceso a la cuenta de Drive, usando las siguientes sentencias:

```
from google.colab import drive
drive.mount('/content/drive')
```

Paso 2: Establecer el directorio de trabajo

Antes de establecer el directorio de trabajo se debe crear una carpeta en el Drive, en este caso llamada “Yolo”, esta será la carpeta que tendrá todos los archivos del entrenamiento. La sentencia para dirigirse al directorio es el siguiente:

```
%cd /content/drive/MyDrive/9NO SEMESTRE/Seminario de Titulación
I/Tesis/Titulación/Yolo
```

Paso 3: Descargar YOLOv5

Con el siguiente código se descarga YOLOv5 desde su repositorio oficial en GitHub, que se guardará en la carpeta creada anteriormente.

```
!git clone https://github.com/ultralytics/yolov5
```

Cuando se clona el repositorio hay que dirigirse a la carpeta descargada con el siguiente comando:

```
%cd yolov5
```

Paso 4: Descargar pesos pre-entrenados (Transfer Learning)

YOLO ofrece algunas subversiones como: Nano (YOLOv5n), Small (YOLOv5s), Medium (YOLOv5m), Large (YOLOv5l), XLarge (YOLOv5x) que son pesos pre-entrenados, que ayudan a no entrenar un modelo desde cero, estos pesos utilizan el dataset de COCO. Con la siguiente sentencia se descargan los archivos:

```
!curl -L
"https://github.com/ultralytics/yolov5/releases/download/v5.0/yolov5s.pt" >
yolov5s.pt
!curl -L
"https://github.com/ultralytics/yolov5/releases/download/v5.0/yolov5m.pt" >
yolov5m.pt
!curl -L
"https://github.com/ultralytics/yolov5/releases/download/v5.0/yolov5l.pt" >
yolov5l.pt
!curl -L
"https://github.com/ultralytics/yolov5/releases/download/v5.0/yolov5x.pt" >
yolov5x.pt
```

Pasos para poder ejecutar el entrenamiento con YOLOv5

Paso 1: Activar el GPU si está disponible

Google Colab ofrece la opción de cambiar el entorno de ejecución, permitiendo usar una GPU haciendo que el proceso de entrenamiento sea rápido.

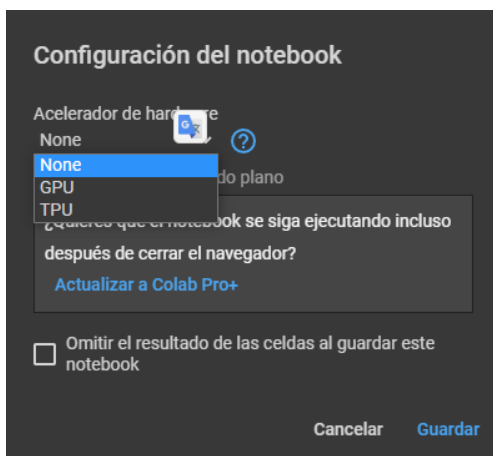


Figura 15. Configuración del notebook de Google Colab

Con la siguiente sentencia se puede verificar el tipo de tarjeta gráfica que proporciona Google Colab.

```
!nvidia-smi
```

Nota: El tiempo de ejecución en Google Colab es limitado, depende de muchos factores como: inactividad, consumo excesivo de los recursos, entre otros., en tal caso se debe esperar un día para comenzar a utilizarlo o comprar la versión pro.

Paso 2: Instalar requerimientos

En el caso de cambiar de entorno de ejecución (GPU), se debe volver a otorgar los permisos para tener acceso a la cuenta de Drive y dirigirse a la carpeta que se clono del GitHub, para luego instalar los requerimientos con la siguiente sentencia:

```
!pip install -r requirements.txt
```

```
*requirements.txt: Bloc de notas
Archivo Edición Formato Ver Ayuda
# pip install -r requirements.txt
# Base -----
matplotlib>=3.2.2
numpy>=1.18.5
opencv-python>=4.1.2
Pillow>=7.1.2
PyYAML>=5.3.1
requests>=2.23.0
scipy>=1.4.1
torch>=1.7.0
torchvision>=0.8.1
tqdm>=4.41.0

# Logging -----
tensorboard>=2.4.1
# wandb
|
# Plotting -----
pandas>=1.1.4
seaborn>=0.11.0

# Export -----
# coremltools>=4.1 # CoreML export
# onnx>=1.9.0 # ONNX export
# onnx-simplifier>=0.3.6 # ONNX simplifier
# scikit-learn>=0.19.2 # CoreML quantization
# tensorflow>=2.4.1 # TFLite export
# tensorflowjs>=3.9.0 # TF.js export
# openvino-dev # OpenVINO export

# Extras -----
# alumentations>=1.0.3
# Cython # for pycocotools https://github.com/cocodataset/cocoapi/issues/172
# pycocotools>=2.0 # COCO mAP
# roboflow
thop # FLOPs computation
```

Figura 16. Instalación de instancias de requirements.txt

Paso 3: Ejecutar el entrenamiento

Antes de comenzar con el entrenamiento es necesario descargar el dataset que se creó en Roboflow y luego subirlo en la carpeta Yolo del drive.

Para comenzar con el entrenamiento del modelo con pesos pre-entrenados se debe de conocer los siguientes términos:

- `train.py`: contiene el código para ejecutar el entrenamiento.
- `img`: indica el tamaño que usa la imagen para el entrenamiento.
- `batch`: indica el número de imágenes que se usan por cada iteración de entrenamiento y se debe usar en múltiplos de 2 (4, 8, 16, 32, 64) hasta lograr que el entrenamiento sea el correcto.
- `epochs`: es el ciclo que ha completado el conjunto de datos utilizados para entrenar una red neuronal. Entre más épocas, mejor será el resultado del modelo, aunque tiene un límite y esto sucede cuando el modelo no puede seguir mejorando.
- `data`: hace el llamado al archivo `data.yaml` que contiene el directorio donde se encuentra la data para el entrenamiento.
- `weights`: hace llamado al peso pre-entrenados descargado.

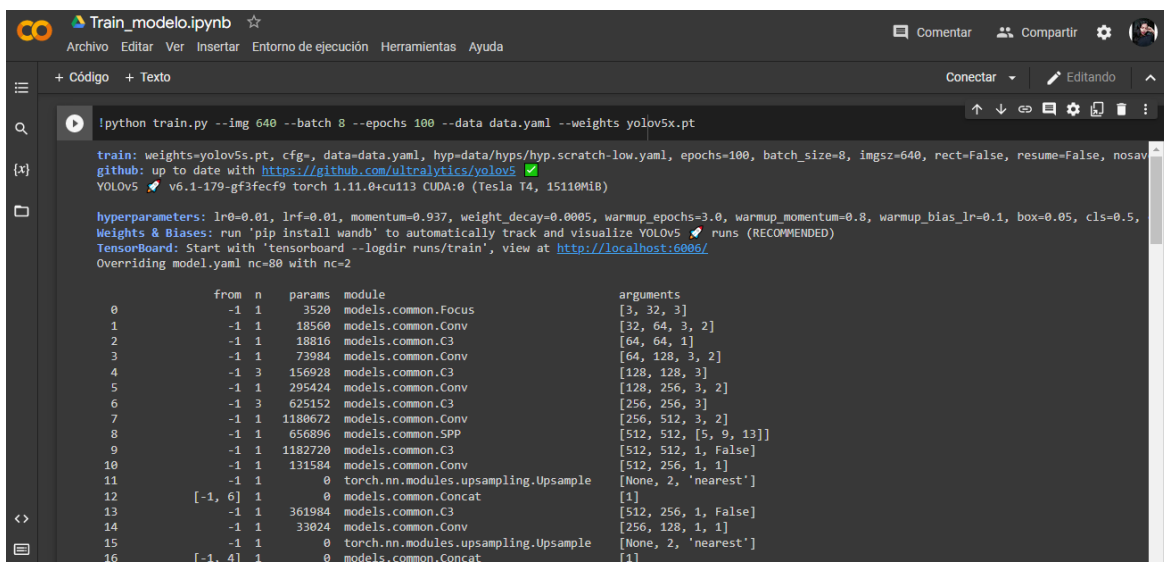
Con el siguiente comando se puede comenzar a entrenar el modelo con pesos pre-entrenados:

```
!python train.py --img 640 --batch 8 --epochs 100 --data data.yaml --weights yolov5x.pt
```

El tiempo de ejecución dependerá del tipo de tarjeta de video asignada.

En caso que exista algún problema durante el entrenamiento se puede utilizar el siguiente comando para retomar el entrenamiento desde el último punto de iteración.

```
!python train.py --resume
```



The screenshot shows a Jupyter Notebook interface with a terminal window. The terminal displays the execution of the command `!python train.py --img 640 --batch 8 --epochs 100 --data data.yaml --weights yolov5x.pt`. The output includes the training command, the path to the weights, and a list of hyperparameters. Below the hyperparameters, there is a table showing the model's architecture details, including the number of layers, parameters, and modules.

from	n	params	module	arguments	
0	-1	1	3520	models.common.Focus	[3, 32, 3]
1	-1	1	18560	models.common.Conv	[32, 64, 3, 2]
2	-1	1	18816	models.common.C3	[64, 64, 1]
3	-1	1	73984	models.common.Conv	[64, 128, 3, 2]
4	-1	3	156928	models.common.C3	[128, 128, 3]
5	-1	1	295424	models.common.Conv	[128, 256, 3, 2]
6	-1	3	625152	models.common.C3	[256, 256, 3]
7	-1	1	1188672	models.common.Conv	[256, 512, 3, 2]
8	-1	1	656896	models.common.SPP	[512, 512, [5, 9, 13]]
9	-1	1	1182720	models.common.C3	[512, 512, 1, False]
10	-1	1	131584	models.common.Conv	[512, 256, 1, 1]
11	-1	1	0	torch.nn.modules.upsampling.Upsample	[None, 2, 'nearest']
12	[-1,	6]	0	models.common.Concat	[1]
13	-1	1	361984	models.common.C3	[512, 256, 1, False]
14	-1	1	33024	models.common.Conv	[256, 128, 1, 1]
15	-1	1	0	torch.nn.modules.upsampling.Upsample	[None, 2, 'nearest']
16	[-1,	4]	1	models.common.Concat	[1]

Figura 17. Entrenamiento de la Red Neuronal

El proceso se realizó 3 veces con diferentes parámetros, hasta encontrar el modelo con un mejor valor de precisión para luego realizar las pruebas.

Los pasos: ejecución de pruebas del modelo, obtención de métricas de cada modelo, comparación de los modelos y selección del mejor modelo, correspondiente a la metodología para la construcción de la red neuronal, serán descritos en el capítulo 3, en el apartado 3.2. Resultados de la Evaluación.

2.3.2. Metodología Mobile-D para el desarrollo de la aplicación móvil

Mobile-D, es una metodología ágil orientada únicamente al desarrollo de las aplicaciones móviles, por tal motivo se eligió esta metodología para el desarrollo de este trabajo.

2.3.2.1. Fase de Exploración

Esta fase se enfoca en el establecimiento de los stakeholder, requisitos funcionales, requisitos no funcionales y el alcance del desarrollo de la aplicación móvil.

Establecimiento de los stakeholder

En la Tabla 9, se evidencia el equipo de stakeholder del trabajo de titulación con sus respectivas funciones establecidas:

Tabla 9. Establecimiento de los stakeholder

Responsables	Cargo	Funciones
Selena Aguilar, Andrés Carrión	Desarrolladores	Implementar las funciones a la aplicación mediante la programación de los módulos.
Selena Aguilar, Andrés Carrión	Analistas	Recolectar la información necesaria para el desarrollo de la aplicación.
Comunidad universitaria y/o externas	Usuario/cliente	Encargado de usar la aplicación final, en este caso personas que ingresen a la FIC.
Selena Aguilar, Andrés Carrión	Tester	Hacer las pruebas de funcionalidad correspondientes durante el desarrollo de la aplicación para evitar futuros errores.

Requisitos Funcionales

A continuación, en la Tabla 10 se especifican los requisitos funcionales que la aplicación será capaz de realizar.

Tabla 10. Requerimientos Funcionales

ID	Requisitos	Descripción	Responsable	Prioridad
RF001	Mostrar información al usuario	El usuario podrá conocer la función de la app y en qué estacionamiento se hará la detección.	Usuario	Media
RF002	Ejecutar detección	El usuario podrá ejecutar el modelo y realizar la detección de las clases (Libre y Ocupado) del estacionamiento de la FIC.	Usuario	Alta
RF003	Visualizar total de estacionamiento disponibles	El usuario podrá informarse del total de estacionamientos disponibles que hay antes de ingresar a visualizar el parqueadero.	Usuario	Alta
RF004	Visualizar la cámara IP	El usuario podrá abrir mediante la app la cámara IP que estará enfocando el estacionamiento de la FIC.	Usuario	Alta

Requisitos No Funcionales

En la Tabla 11, se evidencian los requisitos no funcionales que la aplicación no será capaz de realizar.

Tabla 11. Requerimientos No Funcionales

ID	Requisitos	Descripción	Prioridad
RNF001	Lenguaje de programación	La aplicación móvil será desarrollada con el lenguaje de programación Java	Alta
RNF002	Plataforma	La aplicación será desarrollada para dispositivos con sistema operativo Android	Alta
RNF003	Idioma	La aplicación tendrá habilitado el idioma por defecto (español)	Media
RNF004	Interfaz de usuario	La aplicación tendrá un diseño de interfaz de usuario amigable.	Alta

Definición del alcance

El alcance encierra la detección de plazas de estacionamiento disponibles y ocupadas, abriendo una cámara IP que estará enfocando el estacionamiento de la FIC, procesando el video y haciendo el debido reconocimiento al instante.

Por tal motivo, se desarrolla una aplicación móvil que estará conectada a un back-end, abriendo la cámara IP y procesando la información para mostrar como resultado la detección de las plazas de estacionamiento en un cuadro delimitador indicando si se encuentra libre u ocupado.

2.3.2.2. Fase Inicialización

Esta fase define el cronograma de actividades, recursos de hardware y software usados para el desarrollo de la aplicación móvil. Asimismo, se define la planificación de las fases y los bocetos de la interfaz que tendrá la app.

Cronograma de actividades

Para la elaboración del cronograma se utilizó el software Microsoft Project, donde se detallaron todas las actividades para llevar a cabo el proceso de desarrollo de la aplicación móvil.

En la Figura 18, se evidencia el cumplimiento del cronograma de actividades realizadas de lunes a viernes, con 5 horas diarias.

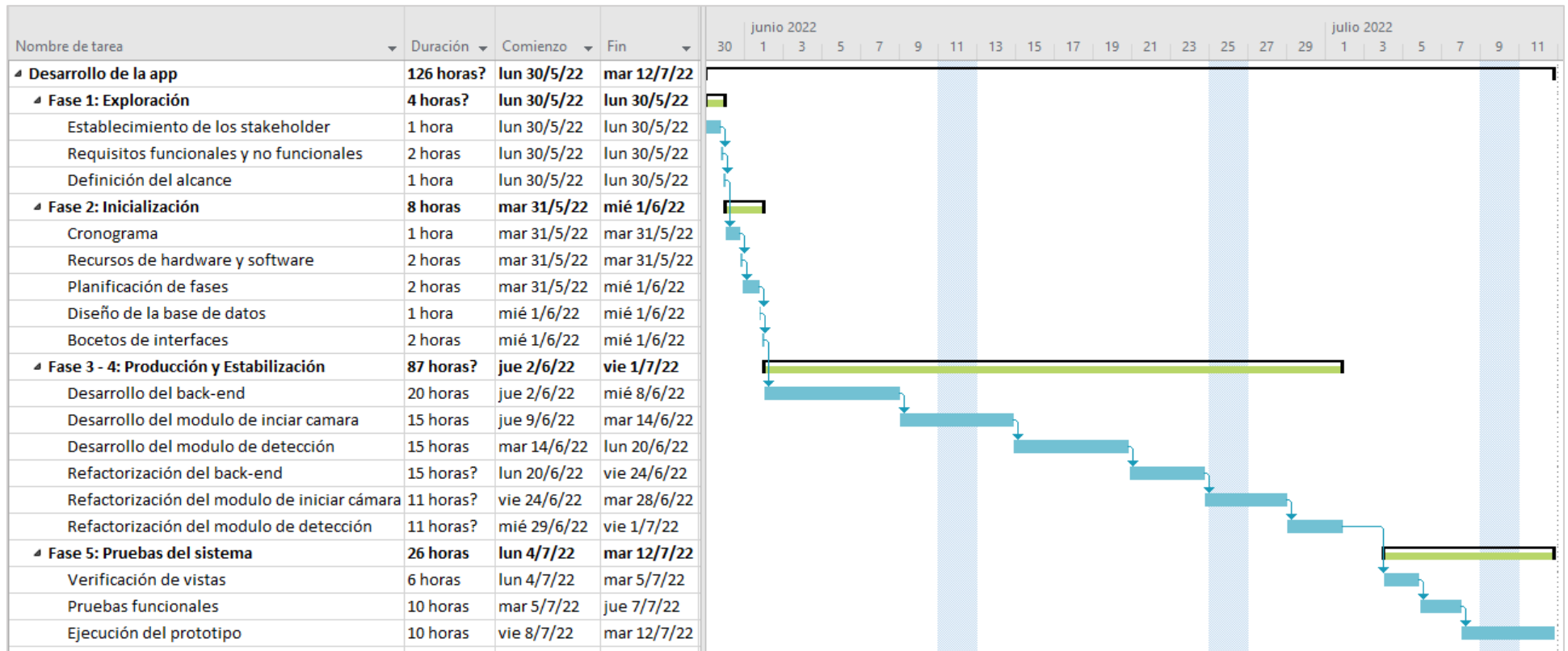


Figura 18. Cronogramas de actividades

Recursos de hardware

En la Tabla 12, se visualizan los recursos de hardware que se usaron para el desarrollo de la aplicación.

Tabla 12. Recursos de Hardware

		Recursos
Hardware	Ordenador	Procesador: Intel(R) Core (TM) i3-3240
		Memoria RAM: 8,00 GB
		Almacenamiento: 250gb
		Sistema Operativo: Windows 10 de 64bits
	Dispositivo móvil	Procesador: Mediatek Helio G70
		Memoria RAM: 4,00 GB
		Almacenamiento: 64GB
		Sistema Operativo: Android versión 10

Recursos de software

En la Tabla 13, se visualiza aquellos programas que fueron necesarios para el desarrollo de la aplicación.

Tabla 13. Recursos de Software

		Descripción	Versión
Software	Android Studio	Es un IDE (Entorno de desarrollo integrado), usando para el desarrollo de aplicaciones móviles para Android	2021.2
	Visual Studio Code	Es un editor de código fuente, usado normalmente para el desarrollo de aplicaciones web en diferentes lenguajes de programación.	1.69.2

Planificación de fases

En la Tabla 14, se evidencia la planificación de fases, donde se describe las actividades que se realizó en cada fase de la metodología.

Tabla 14. Planificación de fases.

Fase	Iteración	Descripción
Exploración	Iteración 0	Establecimiento de los stakeholder, definición de requisitos funcionales, requisitos no funcionales y alcance del trabajo.
Inicialización	Iteración 0	Definición del cronograma de actividades, recursos de hardware, software, planificación de fases y presentación de los bocetos de las interfaces.
Producción	Iteración 1	Desarrollo del back-end de la aplicación.
	Iteración 2	Desarrollo de las interfaces de las pantallas de inicio, informativas y menú principal.
	Iteración 3	Ejecución del funcionamiento de las interfaces de las pantallas de inicio, informativas y menú principal.
	Iteración 4	Desarrollo de la interfaz de detección
	Iteración 5	Ejecución del funcionamiento de la interfaz de detección
Estabilización	Iteración 6	Reestructuración del back-end de la aplicación.
	Iteración 7	Reestructuración de la funcionalidad de las interfaces de las pantallas inicio, informativas y menú principal.
	Iteración 8	Reestructuración de la funcionalidad de la interfaz de detección
Pruebas del sistema	Iteración 9	Pruebas y análisis de resultados de la aplicación, visualizando en tiempo real el parqueadero de la FIC.

Diseño de la base de datos

La base de datos utilizada para este trabajo fue Firebase. Es una base de datos NoSQL que guarda los datos en formato JSON. En la Figura 19, se observa la estructura NoSQL que se usó para guardar el total de estacionamiento libres en la base de datos Firebase.



Figura 19. Estructura NoSQL de Total de estacionamiento disponibles

Boceto de interfaces

A continuación, se visualizan los bocetos de la interfaz de la aplicación móvil, los mismos que fueron elaborados con la herramienta Balsamiq Mockups.

Pantalla de inicio

En la Figura 20, se muestra la pantalla de inicio, la cual contiene el nombre y logo de la aplicación.

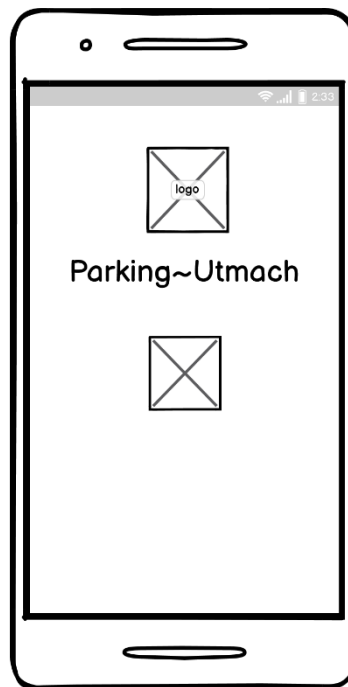


Figura 20. Boceto de la pantalla de inicio

Pantallas informativas

En esta sección se puede observar el funcionamiento de la aplicación de manera resumida, asimismo, informar del lugar donde realiza la detección. Además, cuenta con un botón que permite saltar estas pantallas informativas, esto se puede observar en la Figura 21.



Figura 21. Boceto informativo de la app

Módulo iniciar cámara

La Figura 22, muestra el menú principal, en donde se visualiza una interfaz sencilla que contiene el logo, nombre de la aplicación y un botón para acceder al módulo de detección. Además, antes de abrir la cámara IP para la detección en tiempo real muestra el total de espacios disponibles.

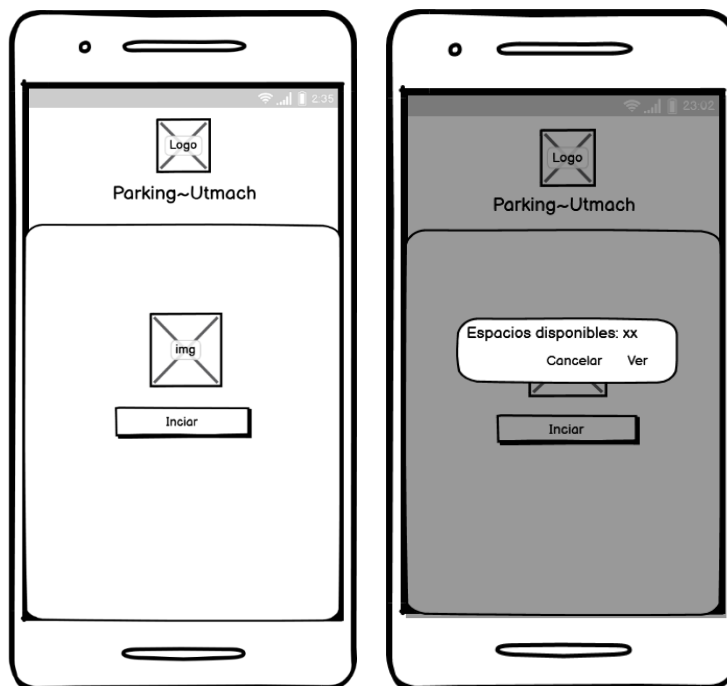


Figura 22. Boceto del menú principal

Módulo de detección

En la Figura 23, se muestra el boceto del módulo de detección, el mismo que se encuentra conformado por un componente de Android Studio que permite visualizar y detectar las plazas de estacionamiento de la FIC mediante la cámara IP.

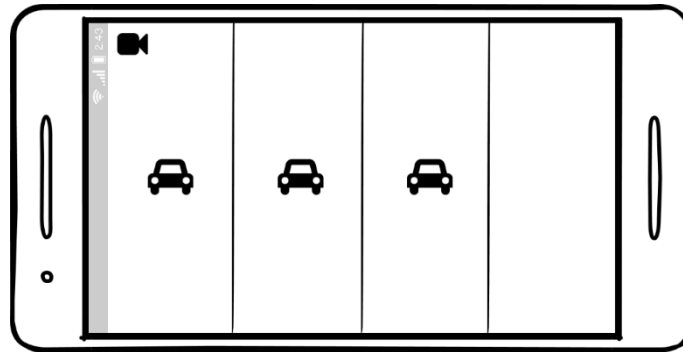


Figura 23. Boceto del módulo de detección

2.3.2.3. Fase de Producción y Fase de Estabilización

La fase de producción y la fase de estabilización se basa en el desarrollo de la parte funcional de la aplicación móvil, aplicando los requisitos que fueron reunidos en la fase de exploración.

Desarrollo del back-end

En la Figura 24, se puede observar la estructura del back-end conformado por algunas carpetas que fueron clonadas de un repositorio de GitHub. Además, consta de algunos paquetes instalados que facilitan la creación del back-end como, django, yolov5, torch, entre otros.

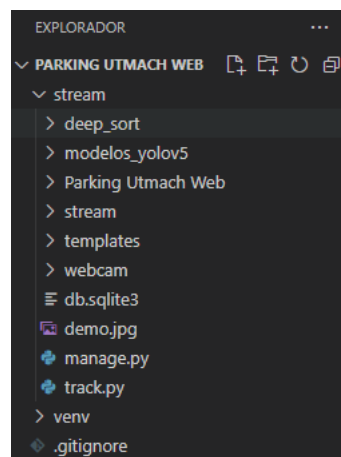


Figura 24. Estructura del Back-End

Para desarrollar el back-end se necesitó crear las URLs, las cuales permiten abrir la cámara IP para comenzar a visualizar la detección en tiempo real.

En la Figura 25, se puede observar parte de la codificación para la creación de las URLs.

```
stream > webcam > urls.py > ...
1  from django.contrib import admin
2  from django.urls import path, include
3
4
5  from webcam import views
6  urlpatterns = []
7      path('inicio/', views.index, name='index'),
8      path('video_feed', views.video_feed, name='video_feed'),
9      path('streaming/', views.streaming, name='streaming'),
10     path('streaming-alone/', views.streamingAlone, name='streamingA'),
11
```

Figura 25. Creación de las URLs del Back-End

En la Figura 26, se puede observar la codificación del módulo stream. Este módulo permite abrir la cámara y a la vez cargar el archivo del modelo entrenado en formato pt, para comenzar a detectar en tiempo real.

```
views.py X
stream > webcam > views.py > stream
67 def stream():
68     cap = cv2.VideoCapture('rtsp://admin:tesis2022@192.168.0.55:554/cam/realmonitor?channel=1&subtype=0') #Camara ip Dahua
69
70     #sum = 0
71     sum2 = 0
72     max_value2 = 0
73
74     while True:
75         ret, frame = cap.read()
76         if not ret:
77             print("Error: failed to capture image")
78             break
79
80         results = model(frame, augment=True)
81         # process
82         annotator = Annotator(frame, line_width=1, pil=not ascii)
83         det = results.pred[0]
84         if det is not None and len(det):
85             xywhs = xyxy2xywh(det[:, 0:4])
86             confs = det[:, 4]
87             cls = det[:, 5]
88             outputs = deepsort.update(xywhs.cpu(), confs.cpu(), cls.cpu(), frame)
89             if len(outputs) > 0:
90                 for j, (output, conf) in enumerate(zip(outputs, confs)):
91                     bboxes = output[0:4]
92                     id = output[4]
93                     cls = output[5]
94
95                     c = int(cls) # integer class
96                     #label = f'{id} {names[c]} (conf: {conf:.2f})'
97                     label = f'{names[c]}'
98
```

Figura 26. Función para la detección

Estructura del proyecto

En la Figura 27, se puede apreciar la estructura del proyecto, la cual consta de un solo directorio, el mismo que contiene todas las clases que fueron necesarias para el desarrollo de la aplicación.

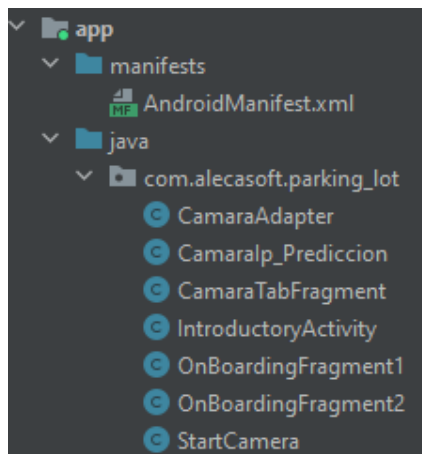


Figura 27. Estructura del proyecto - Android Studio

Módulo iniciar cámara

En la Figura 28, se muestra la función que se utilizó para abrir la cámara IP, en el mismo también se muestra cómo se extrae el dato de la cantidad de espacios libres que se encuentra en la base de datos de Firebase.

```
private void iniciarCamara() {
    databaseReference = FirebaseDatabase.getInstance().getReference();
    databaseReference.child("Libre").addListenerForSingleValueEvent(new ValueEventListener() {
        @Override
        public void onDataChange(@NonNull DataSnapshot snapshot) {
            if(snapshot.exists()){
                String cantidad = snapshot.child("cantidad").getValue().toString();

                new AlertDialog.Builder(getContext())
                    .setTitle("Espacios disponibles: " + cantidad)
                    .setCancelable(false)
                    .setNegativeButton("Cancelar", new DialogInterface.OnClickListener() {
                        @Override
                        public void onClick(DialogInterface dialogInterface, int i) {
                            btnIniciar.setEnabled(true);
                        }
                    })
                    .setPositiveButton("Ver", new DialogInterface.OnClickListener() {
                        @Override
                        public void onClick(DialogInterface dialog, int which) {
                            btnIniciar.setEnabled(true);
                            Intent intent = new Intent(getActivity(), CamaraIp_Prediccion.class);
                            startActivity(intent);
                        }
                    })
                    .show();
            }
        }
    });
}
```

Figura 28. Codificación del módulo Iniciar Cámara.

Módulo de detección

La Figura 29, hace referencia a la sección donde el usuario puede visualizar la detección de los espacios de estacionamiento libre y ocupados con la cámara IP. Para ello, primero se diseñó la interfaz que contiene los elementos tales como el WebView, este elemento hace el llamado al sitio web donde se encuentra el back-end y el elemento SwipeRefreshLayout se encarga de recargar el sitio web dentro de la aplicación móvil.

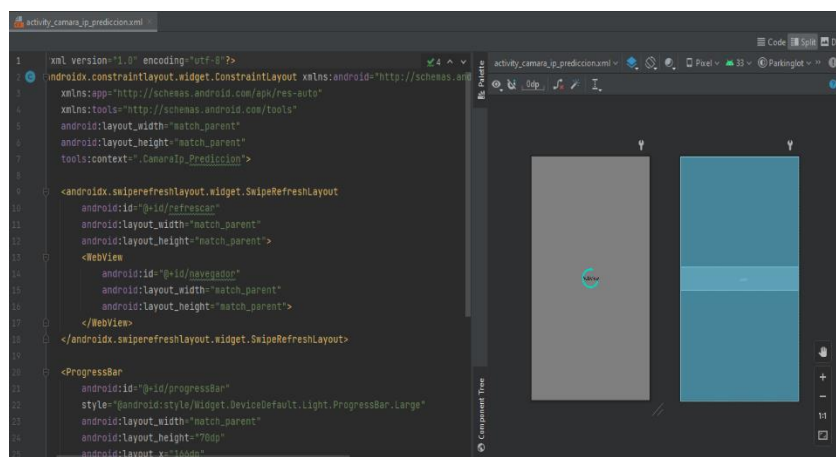


Figura 29. Diseño de la Interfaz del módulo de detección

En la clase llamada “CamaraIp_Prediccion”, se realizó la programación de diferentes funciones. En la Figura 30, se puede apreciar la codificación de la acción que permite refrescar el sitio web de la predicción.

```

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_camara_ip_prediccion);
    getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN, WindowManager.LayoutParams.FLAG_FULLSCREEN);
    relacionarVistas();
    cargarPagina();

    swipeRefreshLayout.setOnRefreshListener(new SwipeRefreshLayout.OnRefreshListener() {
        @Override
        public void onRefresh() {
            new Handler().postDelayed(new Runnable() {
                @Override
                public void run() { progressBar.setVisibility(View.VISIBLE); }
            }, delayMillis: 1000);
            cargarPagina();

            swipeRefreshLayout.setRefreshing(false);
        }
    });
}

```

Figura 30. Codificación de la acción refrescar

La Figura 31, contiene la función llamada “cargarPagina”. En esta función se especifica la URL del sitio web que contiene el Back-End, la misma que es colocada dentro del elemento WebView, el cual permite visualizar la detección en tiempo real.

```

public void cargarPagina(){
    String Url = "http://192.168.0.2:8000/streaming-alone/";
    WebSettings webSettings = myWebView.getSettings();
    myWebView.setWebViewClient(new WebViewClient(){
        public boolean shouldOverrideUrlLoading(WebView view, String url){
            return false;
        }
    });
    myWebView.loadUrl(Url);

    new Handler().postDelayed(new Runnable() {
        @Override
        public void run() { progressBar.setVisibility(View.INVISIBLE); }
    }, delayMillis: 12000);
}

```

Figura 31. Codificación de la función cargar página

2.3.2.4. Fase de Pruebas del sistema

Verificación de vistas

En la Tabla 15, se puede comprobar que la aplicación móvil fue desarrollada con los mismos bocetos elaborados con la herramienta Balsamiq Backup.

Tabla 15. Verificación de vistas

Boceto en Balsamiq Backup	Vista de la aplicación	Cumplimiento
Pantalla de inicio	Pantalla de inicio	Hecho
Pantallas informativas	Pantallas informativas	Hecho
Módulo iniciar cámara	Módulo iniciar cámara	Hecho
Módulo de detección	Módulo de detección	Hecho

Pruebas funcionales

La Tabla 16, se evidencia el cumplimiento total de los requisitos funcionales que se establecieron en la fase de exploración de la metodología.

Tabla 16. Verificación de pruebas funcionales

ID	Requerimientos Funcionales	Cumplimiento
RF001	Mostrar información al usuario	Hecho
RF002	Ejecutar detección	Hecho
RF003	Visualizar total de estacionamiento disponibles	Hecho
RF004	Visualizar la cámara IP	Hecho

2.4. Ejecución del prototipo

La ejecución del prototipo se basa en representar visualmente la funcionalidad e implementación de la aplicación, evidenciando el uso de herramientas tecnológicas que fueron elegidas para cumplir el objetivo del trabajo.

2.4.1. Pantalla de inicio

En la Figura 32, se evidencia la pantalla de inicio donde se muestra el nombre de la aplicación y el logo de la universidad.



Figura 32. Ejecución de la pantalla inicio

2.4.2. Pantallas informativas

En la Figura 33, se observan dos pantallas, la primera muestra información de la funcionalidad de la aplicación y la segunda muestra información del lugar de aplicación y la capacidad de parqueo.



Figura 33. Ejecución de Pantallas informativas

2.4.3. Módulo iniciar cámara

En la Figura 34, se muestra la pantalla del módulo iniciar cámara, el cual contiene el nombre de la aplicación y un botón que permite visualizar la cámara IP que está ejecutándose en el back-end.



Figura 34. Interfaz del módulo iniciar cámara

En la Figura 35, se puede observar la pantalla con la información de cuántos espacios libres existen. Esta información se muestra antes de ingresar a visualizar la cámara IP.



Figura 35. Ejecución de módulo iniciar cámara

2.4.4. Módulo de detección

La Figura 36, muestra la detección en tiempo real de los espacios de estacionamientos disponibles y ocupados en el parqueadero de la Facultad de Ingeniería Civil.



Figura 36. Ejecución de módulo detectar

CAPÍTULO III. EVALUACIÓN DEL PROTOTIPO

3.1. Plan de evaluación

En el plan de evaluación se utilizaron algunas métricas de clasificación para evaluar los resultados del rendimiento de la red neuronal entrenada y un instrumento de recolección de datos para comprobar el cumplimiento de las funcionalidades de la app.

Objetivo: Realizar el plan de evaluación del prototipo, mediante pruebas de rendimiento de la red neuronal y utilizando un instrumento de recopilación de datos, para la demostración del cumplimiento de la hipótesis del trabajo de titulación.

En las Tablas 17 y 18, se especifican las semanas de actividades de evaluación del prototipo que son correspondientes al periodo académico 2022-1.

Tabla 17. Actividades de la semana 10 de la evaluación del prototipo

Actividad: Aplicar métricas de clasificación para evaluar los resultados del rendimiento del modelo entrenado.		
Tipo de Métrica	Descripción	Semana
Precisión	Medir la precisión del modelo mediante la aplicación de su fórmula, con ayuda de los valores obtenidos en la matriz de confusión.	Semana 10
Sensibilidad	Obtener el valor de la sensibilidad a partir de la aplicación de su fórmula, con ayuda de los valores obtenidos en la matriz de confusión.	Semana 10
Valor de referencia	Calcular el valor de referencia a partir de la aplicación de su fórmula, usando los valores de precisión y sensibilidad.	Semana 10

Tabla 18. Actividades de la semana 11 de la evaluación del prototipo

Actividad: Plantear encuesta del cumplimiento de las funcionalidades de la app, en base a la encuesta de diagnóstico inicial para la elaboración del prototipo		
Instrumento de recopilación de datos	Descripción	Semana
Encuesta Final	Se realiza una encuesta a un grupo de personas que hayan utilizado la aplicación de forma local, para comprobar el cumplimiento de las funcionalidades de la app.	Semana 11

3.1.1. Pruebas de la Red Neuronal

3.1.1.1. Métricas

La salida de una red neuronal son resultados binarios, es decir, valores de 0 y 1 lo que indica si el objeto se encuentra o no en la imagen. En la Tabla 19, se explica la matriz de confusión y los 4 posibles resultados [50].

- **Verdadero positivo (VP):** el valor de la situación real es positivo y el valor de la predicción también es positivo.
- **Verdadero negativo (VN):** el valor de la situación real es negativo y el valor de la predicción también es negativo.
- **Falso negativo (FN):** el valor de la situación real es positivo y el valor de la predicción es negativo.

- **Falso positivo (FP):** el valor de la situación real es negativo y el valor de la predicción es positivo.

Tabla 19. Matriz de Confusión

		Valor de Predicción	
		Presencia del objeto	Sin presencia
Situación Real	Presencia del objeto	Verdadero Positivo (VP)	Falso Negativo (FN)
	Sin presencia	Falso Positivo (FP)	Verdadero Negativo (VN)

A continuación, se presentan las métricas de evaluación de calidad para detectar los espacios disponibles y ocupados de un estacionamiento.

Precisión (Precision)

La precisión consiste en mostrar que los objetos detectados sean relevantes o exactos a su predicción, es decir, nos indica cuántas de la cantidad de los casos predichos correctamente resultaron realmente positivos.

$$Precisión = \frac{Verdadero\ Positivo\ (VP)}{Verdadero\ Positivo\ (VP) + Falso\ Positivo\ (FP)}$$

Sensibilidad (Recall)

Esta métrica indica que tan capaz es el algoritmo de detectar los casos positivos.

$$Sensibilidad = \frac{Verdadero\ Positivo\ (VP)}{Verdadero\ Positivo\ (VP) + Falso\ Negativo\ (FN)}$$

Valor de Referencia (F-Score)

El valor de referencia (F-Score) indica que tan bien funciona el sistema, ya que muestra el valor de la precisión y sensibilidad en una sola métrica.

$$F - Score = 2 \times \frac{Precisión \times Sensibilidad}{Precisión + Sensibilidad}$$

3.1.2. Encuesta de cumplimiento de las funcionalidades de la app

Se planteó una encuesta a un grupo de personas, para comprobar el cumplimiento de las funcionalidades de la app de manera local (Anexo 3). Las funcionalidades fueron obtenidas en base a una encuesta de diagnóstico inicial, los resultados de esta encuesta se pueden apreciar en el Anexo 4.

3.2. Resultados de la evaluación

3.2.1. Resultados del entrenamiento de los modelos

Resultado del modelo 1

En el entrenamiento del modelo 1, se usó el peso pre-entrenado yolov5s y algunos parámetros como, tamaño de lotes, épocas, entre otros. En la Tabla 20, se evidencia los parámetros del entrenamiento.

Tabla 20. Indicadores del entrenamiento modelo 1

Parámetros	Valor
Conjuntos de datos	Propio
Total de imágenes del train	15
Total de imágenes del valid	3
Tasa de aprendizaje (Learning rate)	0.01
Pesos pre-entrenados	yolov5s.pt
Número de épocas (epoch)	100
Tamaño de lotes (batch)	2
Dimensiones de las imágenes (alto x ancho)	640 x 640

En la Tabla 21, se observa los resultados del entrenamiento del modelo 1. El valor de precisión es bajo, debido a la escasez de imágenes del dataset.

Tabla 21. Resultados de métricas del entrenamiento modelo 1

Métricas	Valor
Precisión (Precision)	0.18
Sensibilidad (Recall)	0.34
Valor de Referencia (F-Score)	0.23

Resultado de modelo 2

En el entrenamiento del modelo 2, se usó el peso pre-entrenado yolov5x. En la Tabla 22, se puede visualizar los parámetros usados.

Tabla 22. Indicadores del entrenamiento modelo 2

Parámetros	Valor
Conjuntos de datos	Propio
Total de imágenes del train	400
Total de imágenes del valid	100
Tasa de aprendizaje (Learning rate)	0.001
Pesos pre-entrenados	yolov5x.pt
Número de épocas (epoch)	80
Tamaño de lotes (batch)	8
Dimensiones de las imágenes (alto x ancho)	640 x 640

En la Tabla 23, se observa los resultados del entrenamiento del modelo 2. En este caso el valor de la precisión es aceptable, pero aun así la detección no es la correcta. Se necesita hacer más pruebas con más épocas, tamaño de lotes y aumentando imágenes del parqueadero de la FIC de la UTMACH.

Tabla 23. Resultados de métricas del entrenamiento modelo 2

Métricas	Valor
Precisión (Precision)	0.8374
Sensibilidad (Recall)	0.7971
Valor de Referencia (F-Score)	0.8168

Resultado de modelo 3

En la Tabla 24, se visualizan los parámetros utilizados para el entrenamiento del modelo 3. Se usó el peso pre-entrenado yolov5x con un tamaño de lotes de 8 y 500 épocas.

Tabla 24. Indicadores del entrenamiento modelo 3

Parámetros	Valor
Conjuntos de datos	Propio
Total de imágenes de train	800
Total de imágenes del valid	200
Tasa de aprendizaje (Learning rate)	0.001
Pesos pre-entrenados	yolov5x.pt
Número de épocas (epoch)	500
Tamaño de lotes (batch)	8
Dimensiones de las imágenes (alto x ancho)	640 x 640

En la Tabla 25, se observa los resultados del entrenamiento del modelo 3. Este modelo tiene mejores resultados de métricas, esto hace que detecte casi perfecto las plazas de estacionamiento.

Tabla 25. Resultados de métricas del entrenamiento modelo 3

Métricas	Valor
Precisión (Precision)	0.8755
Sensibilidad (Recall)	0.8158
Valor de Referencia (F-Score)	0.8446

3.2.2. Resultados del rendimiento

Resultado del Rendimiento - modelo 1

La siguiente prueba, se realizó utilizando la app móvil. En la Figura 37, se observa que el resultado de la predicción utilizando el modelo 1 es mala, el algoritmo no detecta ninguna plaza de estacionamiento.



Figura 37. Resultado del rendimiento - modelo 1

A continuación, en la Tabla 26, se observa la matriz de confusión de la prueba de rendimiento 1, teniendo resultados no favorables.

Tabla 26. Resultado del modelo 1 - Matriz de Confusión

Matriz de confusión			
		Resultado de predicción	
		Libre (+)	Ocupado (-)
Situación Real	Libre (+)	VP 0	FN 0
	Ocupado (-)	FP 0	VN 0

Resultado del Rendimiento - modelo 2

En la Figura 38, se puede observar la prueba de rendimiento 2. El algoritmo detecta algunos espacios disponibles y ocupados, pero no de manera correcta.



Figura 38. Resultado del rendimiento - modelo 2

En la Tabla 17, se observa la matriz de confusión de los resultados obtenidos de la prueba de rendimiento 2, teniendo resultados un poco favorables.

Tabla 27. Resultado del modelo 2 - Matriz de Confusión

Matriz de confusión			
		Resultado de predicción	
		Libre (+)	Ocupado (-)
Situación Real	Libre (+)	VP 6	FN 0
	Ocupado (-)	FP 2	VN 0

Resultado del Rendimiento - modelo 3

En la Figura 39, se visualiza que el algoritmo detecta los espacios disponibles y ocupados con mayor exactitud que los modelos anteriores.

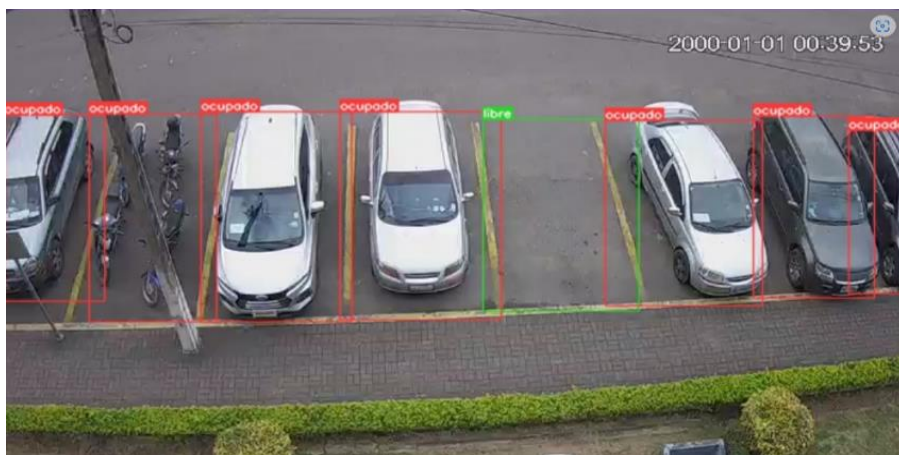


Figura 39. Resultado del rendimiento - modelo 3

En la Tabla 18, se observa la matriz de confusión de los resultados obtenidos de la prueba de rendimiento 3. El modelo propuesto ya muestra mejores resultados en la predicción.

Tabla 28. Resultado del modelo 3 - Matriz de Confusión

Matriz de confusión			
		Resultado de predicción	
		Libre (+)	Ocupado (-)
Situación Real	Libre (+)	VP 1	FN 0
	Ocupado (-)	FP 0	VN 7

En la Tabla 29, se observa las métricas de evaluación de cada prueba realizada anteriormente, en donde se comparan los resultados de cada uno de los modelos, para elegir el que muestre mejores resultados.

Tabla 29. Métricas de evaluación de las pruebas

	Métricas de evaluación		
	Precisión (Precision)	Sensibilidad (Recall)	Valor de Referencia (F-Score)
Prueba 1	0	0	0
Prueba 2	0.75	1	0.86
Prueba 3	1	1	1

Con los resultados obtenidos de cada métrica de evaluación, se ha seleccionado el modelo utilizado en la prueba 3, debido a que arrojó mejor resultado en la predicción.

3.2.3. Resultados de la encuesta del cumplimiento de las funcionalidades de la app

A continuación, se visualiza las tabulaciones de los resultados de la encuesta que se realizó para comprobar el cumplimiento de las funcionalidades de la app, que fueron identificadas en base a una encuesta inicial. Esta encuesta fue efectuada a un grupo reducido de 10 personas, porque las pruebas se hicieron de manera local.

1. ¿La aplicación presenta una pantalla inicial para su descripción, el lugar de aplicación y la capacidad de parqueo?

10 respuestas

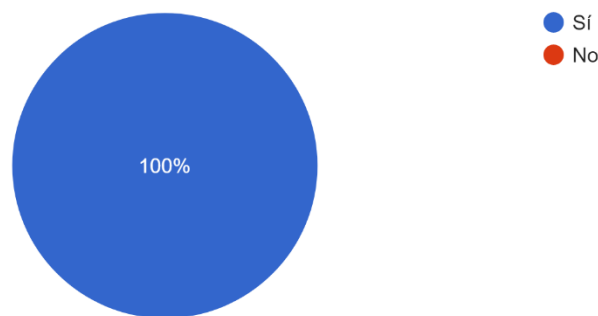


Figura 40. Pregunta 1 de la encuesta de cumplimiento de funcionalidades

2. ¿La aplicación tiene una opción para visualizar el área de parqueo mediante una cámara IP?

10 respuestas

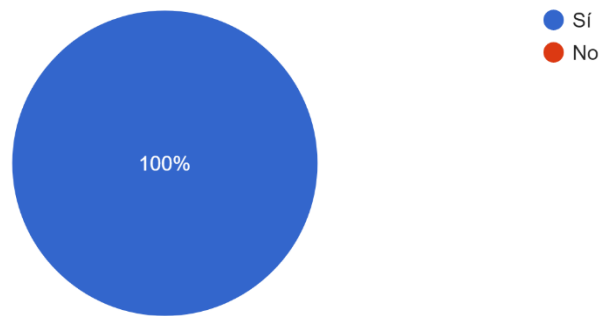


Figura 41. Pregunta 2 de la encuesta de cumplimiento de funcionalidades

3. ¿La aplicación muestra el total de espacios disponibles de parqueo antes de visualizar la cámara IP?

10 respuestas

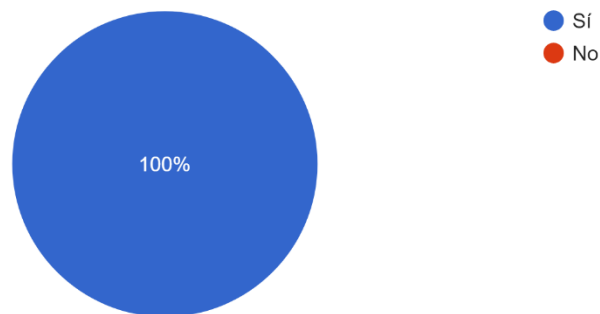


Figura 42. Pregunta 3 de la encuesta de cumplimiento de funcionalidades

4. ¿La aplicación visualiza los espacios disponibles y ocupados en el área de parqueo?

10 respuestas

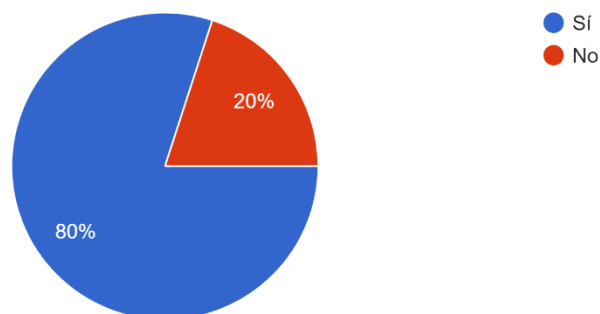


Figura 43. Pregunta 4 de la encuesta de cumplimiento de funcionalidades

5. ¿La aplicación permite recargar el Streaming?

10 respuestas

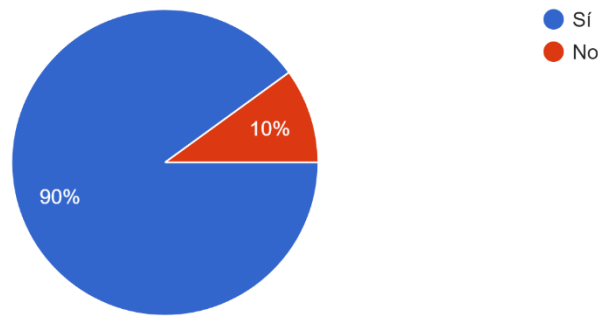


Figura 44. Pregunta 5 de la encuesta de cumplimiento de funcionalidades

6. La aplicación pregunta, “¿Desea cerrar la aplicación?”

10 respuestas

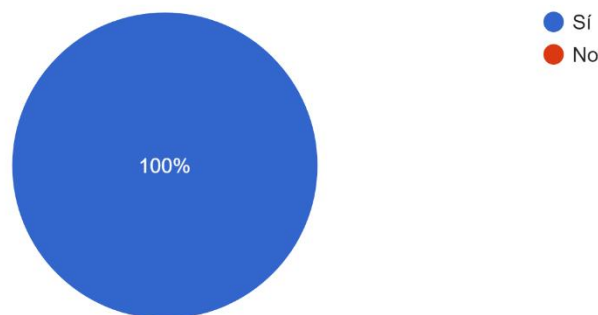


Figura 45. Pregunta 6 de la encuesta de cumplimiento de funcionalidades

En base a los resultados obtenidos en las tabulaciones, se puede observar que la app si cumple con todas las funciones. A continuación, se presenta la Tabla 30:

Tabla 30. Cumplimiento de las funcionalidades de la app

Preguntas	Cumplimiento	
	Si	No
1. ¿La aplicación presenta una pantalla inicial para su descripción, el lugar de aplicación y la capacidad de parqueo?	<input checked="" type="checkbox"/>	<input type="checkbox"/>
2. ¿La aplicación tiene una opción para visualizar el área de parqueo mediante una cámara IP?	<input checked="" type="checkbox"/>	<input type="checkbox"/>
3. ¿La aplicación muestra el total de espacios disponibles de parqueo antes de visualizar la cámara IP?	<input checked="" type="checkbox"/>	<input type="checkbox"/>
4. ¿La aplicación visualiza los espacios disponibles y ocupados en el área de parqueo?	<input checked="" type="checkbox"/>	<input type="checkbox"/>
5. ¿La aplicación permite recargar el Streaming?	<input checked="" type="checkbox"/>	<input type="checkbox"/>
6. La aplicación pregunta, “¿Desea cerrar la aplicación?”	<input checked="" type="checkbox"/>	<input type="checkbox"/>

CONCLUSIONES

Como resultado del desarrollo de la aplicación de detección de plazas de estacionamiento en el parqueadero de la Facultad de Ingeniería Civil de la UTMACH, en tiempo real, aplicando Aprendizaje Profundo, se concluye lo siguiente:

- Se realizó búsquedas bibliográficas en varias revistas científicas utilizando la metodología de Revisión Sistemática de Literatura, con lo cual se logró construir el marco teórico del presente proyecto.
- Se construyó un dataset con imágenes del estacionamiento de la FIC, que sirvió exitosamente para el entrenamiento del modelo con una red neuronal convolucional utilizando YOLOv5.
- Se planificó el desarrollo de la aplicación móvil y se elaboró correctamente su cronograma de acuerdo a la metodología Mobile-D.
- Se desarrolló una aplicación móvil integrando aprendizaje profundo para la detección en tiempo real de las plazas disponibles y ocupadas en el parqueadero de la Facultad de Ingeniería Civil de la UTMACH.
- Se realizaron las pruebas finales de la detección de plazas de estacionamiento, por medio de la aplicación móvil de manera local, las cuales fueron exitosas. Por lo tanto, la hipótesis fue demostrada porque el uso de Deep Learning si permitió detectar espacios disponibles y ocupados en el parqueadero de la FIC.

RECOMENDACIONES

De acuerdo a la experiencia que se obtuvo desarrollando una aplicación móvil que integre el aprendizaje profundo, mediante la tecnología YOLOv5, es necesario recomendar lo siguiente:

- Para desarrollar una aplicación móvil que integre aprendizaje profundo se debe otorgar ciertos permisos para la conexión con el back-end y así realizar la detección en tiempo real de las plazas disponibles y ocupadas.
- Implementar la metodología de Revisión Sistemática de Literatura, para obtener material de calidad necesario para elaborar el marco teórico de trabajos similares.
- Utilizar software y metodologías que permitan realizar planificaciones de cronograma de actividades en un tiempo determinado, para tener un adecuado control durante el desarrollo de una aplicación.

- Las imágenes para el dataset deben cumplir con algunos requisitos como, buena calidad, resolución, entre otros., para el entrenamiento del modelo de red neuronal convolucional y deben ser etiquetados correctamente con alguna herramienta.
- Se debe realizar pruebas finales de la aplicación, para comprobar si cumple con las funciones establecidas en los requerimientos. Además, permite obtener observaciones útiles para realizar mejoras a futuro.
- Para el modelo, si se desea mejorar la predicción se debe seguir experimentando con otros parámetros para entrenar una nueva red y también agregar imágenes nuevas para mejorar la data.
- Este trabajo de titulación puede ser mejorado e implementado en un hosting, solo hay que tener en cuenta los recursos para la ejecución del back-end y mejorar la latencia del procesamiento del video para la detección en tiempo real.

REFERENCIAS BIBLIOGRÁFICAS

- [1] nedapidentification.com, “Detección de vehículos.”
<https://www.nedapidentification.com/es/soluciones/deteccion-de-vehiculos/> (accessed Dec. 18, 2021).
- [2] elcomercio.com, “El Comercio,” Nov. 01, 2019.
<https://www.elcomercio.com/actualidad/negocios/parque-automotor-ecuador-crecimiento-decada.html> (accessed Dec. 18, 2021).
- [3] M. Mahmud, M. S. Kaiser, A. Hussain, and S. Vassanelli, “Applications of deep learning and reinforcement learning to biological data,” *IEEE Trans Neural Netw Learn Syst*, vol. 29, no. 6, pp. 2063–2079, Jun. 2018, doi: 10.1109/TNNLS.2018.2790388.
- [4] A. Farley, H. Ham, and Hendra, “Real time IP camera parking occupancy detection using deep learning,” *Procedia Comput Sci*, vol. 179, pp. 606–614, 2021, doi: 10.1016/j.procs.2021.01.046.
- [5] J. Vilela, J. Castro, L. E. G. Martins, and T. Gorschek, “Integration between requirements engineering and safety analysis: a systematic literature review,” *Journal of Systems and Software*, vol. 125, pp. 68–92, Mar. 2017, doi: 10.1016/j.jss.2016.11.031.

- [6] J. I. Victor Aguilar-Alvarado, W. I. Braulio Rivas-Asanza, and K. Elizabeth Garcia-Galarza III, “Visión artificial para control de estacionamiento vehicular,” vol. 6, no. 9, pp. 2134–2154, 2021, doi: 10.23857/pc.v6i9.3159.
- [7] L. N. Rosales Lindao, “Diseño e implementacion de un parqueo inteligente utilizando arduino yun basado en internet de las cosas (IoT),” Universidad Politécnica Salesiana, Guayaquil, 2016. Accessed: Dec. 18, 2021. [Online]. Available: <https://dspace.ups.edu.ec/bitstream/123456789/13461/1/UPS-GT001798.pdf>
- [8] M. A. Avendaño Quinde and C. A. Garzón Velásquez, “Desarrollo de un prototipo de control de espacios de estacionamientos libres mediante el uso de visión artificial en el edificio de la carrera de ingeniería en sistemas computacionales de la universidad de Guayaquil,,” Carrera de Ingeniería en Sistemas Computacionales, Universidad de Guayaquil, Guayaquil, 2019. Accessed: Dec. 18, 2021. [Online]. Available: <http://repositorio.ug.edu.ec/handle/redug/48872>
- [9] G. Rodríguez-Miranda, R. Santos-Osorio, C. S. Ordaz-Banda, and J. A. Lopez-Rivera, “Estacionamiento inteligente,” *Revista de Ingenieria Innovativa*, pp. 34–39, Mar. 2019, doi: 10.35429/joie.2019.9.3.34.39.
- [10] G. Amato, F. Carrara, F. Falchi, C. Gennaro, and C. Vairo, “Car parking occupancy detection using smart camera networks and deep learning,” *2016 IEEE Symposium on Computers and Communication (ISCC)*, 2016, doi: 10.1109/ISCC.2016.7543901.
- [11] P. Gupta, B. Pareek, G. Singal, and D. V. Rao, “Edge device based Military Vehicle Detection and Classification from UAV,” *Multimed Tools Appl*, vol. 81, no. 14, pp. 19813–19834, Jun. 2022, doi: 10.1007/s11042-021-11242-y.
- [12] V. Kaul, S. Enslin, and S. A. Gross, “History of artificial intelligence in medicine,” *Gastrointest Endosc*, vol. 92, no. 4, pp. 807–812, Oct. 2020, doi: 10.1016/j.gie.2020.06.040.
- [13] X. Ren and Y. Chen, “How Can Artificial Intelligence Help with Space Missions - A Case Study: Computational Intelligence-Assisted Design of Space Tether for Payload Orbital Transfer under Uncertainties,” *IEEE Access*, vol. 7, pp. 161449–161458, 2019, doi: 10.1109/ACCESS.2019.2951136.

- [14] I. Erkmen and A. K. Topalli, “Four methods for short-term load forecasting using the benefits of artificial intelligence,” *Electrical Engineering (Archiv fur Elektrotechnik)*, vol. 85, no. 4, pp. 229–233, Sep. 2003, doi: 10.1007/s00202-003-0163-9.
- [15] F. Piccialli, F. Giampaolo, E. Prezioso, D. Crisci, and S. Cuomo, “Predictive analytics for smart parking: a deep learning approach in forecasting of IoT data,” *ACM Trans Internet Technol*, vol. 21, no. 3, pp. 1–21, 2021, doi: <https://doi.org/10.1145/3412842>.
- [16] G. Amato, F. Carrara, F. Falchi, C. Gennaro, C. Meghini, and C. Vairo, “Deep learning for decentralized parking lot occupancy detection,” *Expert Syst Appl*, vol. 72, pp. 327–334, Apr. 2017, doi: 10.1016/j.eswa.2016.10.055.
- [17] E. Grossi and M. Buscema, “Introduction to artificial neural networks,” *Eur J Gastroenterol Hepatol*, vol. 19, no. 12, pp. 1046–1054, Dec. 2007, doi: 10.1097/MEG.0b013e3282f198a0.
- [18] F. M. Awan, Y. Saleem, R. Minerva, and N. Crespi, “A comparative analysis of machine/deep learning models for parking space availability prediction,” *Sensors*, vol. 20, no. 1, p. 322, Jan. 2020, doi: 10.3390/s20010322.
- [19] S. Park, “D-PARK: user-centric smart parking system over BLE-beacon based internet of things,” *Electronics (Basel)*, vol. 10, no. 5, p. 541, Feb. 2021, doi: 10.3390/electronics10050541.
- [20] N. Hurst-Tarrab, L. Chang, M. Gonzalez-Mendoza, and N. Hernandez-Gress, “Robust parking block segmentation from a surveillance camera perspective,” *Applied Sciences*, vol. 10, no. 15, p. 5364, Aug. 2020, doi: 10.3390/app10155364.
- [21] C. Badii, P. Nesi, and I. Paoli, “Predicting available parking slots on critical and regular services by exploiting a range of open data,” *IEEE Access*, vol. 6, pp. 44059–44071, Aug. 2018, doi: 10.1109/ACCESS.2018.2864157.
- [22] S. Yoon, H. Jeon, and D. Kum, “Predictive cruise control using radial basis function network-based vehicle motion prediction and chance constrained model predictive control,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 10, pp. 3832–3843, Oct. 2019, doi: 10.1109/TITS.2019.2928217.

- [23] K. v. Sakhare, T. Tewari, and V. Vyas, “Review of vehicle detection systems in advanced driver assistant systems,” *Archives of Computational Methods in Engineering*, vol. 27, no. 2, pp. 591–610, Apr. 2020, doi: 10.1007/s11831-019-09321-3.
- [24] S. Vítek and P. Melničuk, “A distributed wireless camera system for the management of parking spaces,” *Sensors*, vol. 18, no. 1, p. 69, Dec. 2017, doi: 10.3390/s18010069.
- [25] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015, doi: 10.1038/nature14539.
- [26] Y. Xin *et al.*, “Machine learning and deep learning methods for cybersecurity,” *IEEE Access*, vol. 6, pp. 35365–35381, May 2018, doi: 10.1109/ACCESS.2018.2836950.
- [27] C. Jang and M. Sunwoo, “Semantic segmentation-based parking space detection with standalone around view monitoring system,” *Mach Vis Appl*, vol. 30, no. 2, pp. 309–319, Mar. 2019, doi: 10.1007/s00138-018-0986-z.
- [28] S. Jiang, H. Jiang, S. Ma, and Z. Jiang, “Detection of parking slots based on mask R-CNN,” *Applied Sciences*, vol. 10, no. 12, p. 4295, Jun. 2020, doi: 10.3390/app10124295.
- [29] Roboflow, “Roboflow.” <https://roboflow.com/> (accessed Apr. 11, 2022).
- [30] S. Shariar and K. M. Azharul Hasan, “GPU accelerated indexing for high order tensors in google colab,” *2020 IEEE Region 10 Symposium (TENSymp)*, 2020, doi: 10.1109/TENSymp50017.2020.9230789.
- [31] T. Carneiro, R. V. Medeiros Da Nobrega, T. Nepomuceno, G.-B. Bian, V. H. C. de Albuquerque, and P. P. R. Filho, “Performance analysis of google colaboratory as a tool for accelerating deep learning applications,” *IEEE Access*, vol. 6, pp. 61677–61685, 2018, doi: 10.1109/ACCESS.2018.2874767.
- [32] [research.google.com](https://research.google.com/colaboratory/intl/es/faq.html), “Google colaboratory.” <https://research.google.com/colaboratory/intl/es/faq.html> (accessed Dec. 18, 2021).
- [33] M. F. Wofford, B. M. Boscoe, C. L. Borgman, I. v. Pasquetto, and M. S. Golshan, “Jupyter notebooks as discovery mechanisms for open science: citation practices in the astronomy community,” *Comput Sci Eng*, vol. 22, no. 1, pp. 5–15, Jan. 2020, doi: 10.1109/MCSE.2019.2932067.
- [34] Glenn Jocher, “YOLOv5,” May 2020.

- [35] J. Yao, J. Qi, J. Zhang, H. Shao, J. Yang, and X. Li, “A real-time detection algorithm for kiwifruit defects based on yolov5,” *Electronics (Switzerland)*, vol. 10, no. 14, Jul. 2021, doi: 10.3390/electronics10141711.
- [36] tensorflow.org, “Plataforma de extremo a extremo de código abierto para el aprendizaje automático.” <https://www.tensorflow.org/?hl=es-419> (accessed Dec. 18, 2021).
- [37] keras.io, “Keras: the python deep learning API.” <https://keras.io/> (accessed Dec. 18, 2021).
- [38] Microsoft, “Visual studio code.” <https://code.visualstudio.com/docs> (accessed Jul. 30, 2022).
- [39] E. W. Anderson, G. A. Preston, and C. T. Silva, “Using python for signal processing and visualization,” *Comput Sci Eng*, vol. 12, no. 4, pp. 90–95, Jul. 2010, doi: 10.1109/MCSE.2010.91.
- [40] developer.android.com, “Introducción a android studio.” <https://developer.android.com/studio/intro?hl=es-419> (accessed Dec. 18, 2021).
- [41] java.com, “Java.” https://www.java.com/es/download/help/whatis_java.html (accessed Dec. 18, 2021).
- [42] Wu-Jeng Li, Chiaming Yen, You-Sheng Lin, Shu-Chu Tung, and ShihMiao Huang, “Just IoT internet of things based on the firebase real-time database,” 2018. doi: 10.1109/SMILE.2018.8353979.
- [43] firebase.google.com, “Firebase por plataforma.” <https://firebase.google.com/docs?hl=es-419> (accessed Dec. 18, 2021).
- [44] utmachala.edu.ec, “Portal web de la universidad técnica de machala.” <https://www.utmachala.edu.ec/portalwp> (accessed Dec. 16, 2021).
- [45] surveymonkey.com, “SurveyMonkey.” <https://es.surveymonkey.com/mp/sample-size-calculator/> (accessed Dec. 19, 2021).
- [46] E. D. Montesdeoca Ordoñez, “Implementación de un sistema de reconocimiento del uso de mascarillas como medida de precaución contra el covid19 usando deep learning,” Universidad Técnica de Machala, Machala, 2020. Accessed: Jan. 24, 2022. [Online]. Available: <http://repositorio.utmachala.edu.ec/handle/48000/15890>

- [47] J. P. Cruzado, T. L. Cespedes, G. P. Bohorquez Coria, and J. L. Herrera Salazar, "Geobook: mobile app with augmented reality for Learning geometry," in *2020 IEEE Engineering International Research Conference (EIRCON)*, Oct. 2020, pp. 1–4. doi: 10.1109/EIRCON51178.2020.9253759.
- [48] A. Sanchez, R. Tinco, and J. L. Herrera, "A mobile application based on expert judgment and their influence on investigations," Oct. 2020. doi: 10.1109/EIRCON51178.2020.9254012.
- [49] Glenn Jocher, "YoloV5." <https://github.com/ultralytics/yolov5> (accessed Apr. 11, 2022).
- [50] M. Massiris, C. Delrieux, and J. Álvaro Fernández, "Detección de equipos de protección personal mediante red neuronal convolucional yolo."

ANEXOS

Anexo 1. Matriz de consistencia

Problema, objeto y campo	Objetivo	Marco Teórico	Hipótesis	Variables	Metodología
<p>Problema: ¿Cómo implementar un sistema de detección de plazas disponibles y ocupadas utilizando aprendizaje profundo en los parqueaderos de vehículos de la Facultad de Ingeniería Civil de la Universidad Técnica de Machala, durante el periodo 2021-2022?</p> <p>Problemas específicos:</p> <ul style="list-style-type: none"> • ¿Cómo obtener imágenes útiles para detectar espacios de estacionamiento disponibles? • ¿Cuál es la técnica que se utilizará para la detección de plazas de estacionamiento? 	<p>Objetivo General: Desarrollar una aplicación móvil utilizando un modelo de red neuronal de Aprendizaje Profundo y la metodología Mobile-D, para la detección de plazas disponibles y ocupadas en el estacionamiento de la Facultad de Ingeniería Civil de la Universidad Técnica de Machala.</p> <p>Objetivos Específicos:</p> <ul style="list-style-type: none"> • Realizar una búsqueda bibliográfica, aplicando la Metodología de Revisión Sistemática de Literatura, para la elaboración del estado del arte y marco teórico. 	<p>Antecedentes históricos a nivel internacional y nacional del objeto, campo: 1.2.2.1. Historia de Inteligencia Artificial</p> <p>Fundamentos Teóricos de objeto, campo y variables: Generalidades de la IA Historia de IA Definición de IA Beneficios del uso de IA</p> <p>Deep learning Definición de Aprendizaje Profundo</p> <p>Redes neuronales Definición de red neuronal Tipos de red neuronal Red neuronal multicapa Red neuronal convolucional red neuronal recurrente Redes de base radial</p> <p>Aprendizaje o entrenamiento Definición de aprendizaje Tipos de aprendizaje Aprendizaje supervisado</p> <p>Herramientas y/o materiales para la construcción de la red neuronal</p>	<p>Hipótesis General: El desarrollo de una aplicación móvil que integra Aprendizaje Profundo o Deep Learning permitirá la detección de espacios disponibles y ocupados en el parqueadero de vehículos de la Facultad de Ingeniería Civil de la Universidad Técnica de Machala.</p>	<p>Variable 1 / Independiente: Desarrollo de una Aplicación móvil que integra Aprendizaje Profundo</p> <p>Dimensiones o categorías:</p> <ol style="list-style-type: none"> 1. Metodologías de desarrollo de aplicación móvil con IA 2. Deep Learning 3. Sistema de detección de parqueo <p>Variable 2/ Dependiente: Detectar plazas disponibles y ocupadas en el estacionamiento de vehículos</p> <p>Dimensiones o categorías:</p>	<p>Enfoque: El enfoque de investigación de este trabajo fue cuantitativo, ya que se realizó una recolección de datos (imágenes) para crear y entrenar un dataset y además se utilizó la red convolucional YOLOv5, que permitió para detectar plazas de estacionamiento disponibles y ocupadas.</p> <p>Alcance: El alcance es exploratorio y descriptivo, debido a que al principio se realizó una indagación bibliográfica para tener más información correspondiente al tema de Aprendizaje Profundo aplicados en estacionamiento de vehículos, para luego</p>

<ul style="list-style-type: none"> • ¿Cómo vincular el aprendizaje profundo a la aplicación móvil, para la detección de plazas de estacionamiento? <p>Objeto de estudio: Sistema de detección de plazas de parqueo en un estacionamiento.</p> <p>Campo de Acción: Desarrollo de una aplicación móvil integrando aprendizaje profundo.</p>	<ul style="list-style-type: none"> • Planificar el cronograma y actividades de acuerdo a la metodología Mobile-D, para tener un adecuado seguimiento durante el desarrollo de la aplicación. • Crear un dataset con imágenes del estacionamiento de la FIC (disponibles y ocupadas), para el entrenamiento del modelo con una red neuronal convolucional utilizando YOLOv5. • Comprobar el funcionamiento de la detección de las plazas de estacionamiento disponibles y ocupadas, mediante la aplicación móvil. 	<p>Dataset Roboflow Google Colab Jupyter notebook YOLOv5 Tensorflow Keras Visual Studio Code Python Android Studio Java Firebase</p>		<ul style="list-style-type: none"> • Satisfacción del usuario en base a la funcionalidad de la app • Eficacia 	<p>mostrar las especificaciones más importantes encontradas. El diseño de la red neuronal y la aplicación móvil fue de mayor prioridad, lo que corresponde a la latencia que se presentó al momento de procesar el video en tiempo real (Streaming) quedará como estudio a futuro para el mejoramiento del rendimiento. Además, la ejecución y pruebas finales de la aplicación móvil fueron realizadas de manera local.</p> <p>Diseño: El diseño de Investigación fue Cuasi-experimental porque el trabajo está destinado a un grupo de usuarios, los cuales permitieron realizar las pruebas finales de la app móvil.</p> <p>Unidades de análisis: Población:</p>
--	---	--	--	---	--

					<p>En la Tabla 5, se especifica el total de los miembros de la Facultad de Ingeniería Civil clasificados por: estudiantes, docentes, trabajadores administrativos /servicios y personas externas que hacen uso de la institución educativa.</p> <p>Muestra: Aplicando la muestra aleatoria simple y con ayuda de la herramienta web SurveyMonkey [45], se obtuvo como resultado la muestra de 90 personas a partir de la población de 1274 perteneciente a la FIC y personas externas de la Universidad Técnica de Machala, se utilizó un nivel de confianza del 95% con un margen de error del 10%.</p> <p>También se tomó en cuenta como muestra las imágenes, para la construcción del</p>
--	--	--	--	--	--

					<p>dataset y entrenamiento del modelo, teniendo un total de 1000 imágenes correspondiente al 100%. Para el entrenamiento se tomó en cuenta el 80% (800 imágenes) y para la validación el 20% (200 imágenes). A continuación, en la Tabla 6 se desglosa la información.</p> <p>Técnicas e instrumentos de recolección de datos: Para este trabajo, se ha detallado en la Tabla 7 la técnica e instrumentos de recopilación de datos que fueron usados.</p> <p>Técnicas de procesamiento de datos: Se usó Deep Learning, con el cuál se entrenó un modelo de red neuronal convolucional, para ello se necesitó un dataset de imágenes que estén etiquetadas</p>
--	--	--	--	--	---

					<p>por clases, la primera clase fue de imágenes de estacionamientos disponibles y la otra clase de estacionamientos ocupados. El modelo fue evaluado usando la matriz de confusión y métricas derivadas (precisión, sensibilidad y valor de referencia).</p> <p>Como segunda técnica de recopilación de datos, se realizó una encuesta del cumplimiento de las funcionalidades de la app, en base a la encuesta de diagnóstico inicial para la elaboración del prototipo. Estos resultados obtenidos permitieron realizar las respectivas tabulaciones.</p>
--	--	--	--	--	---

Anexo 2. Encuesta de diagnóstico inicial para la elaboración del prototipo
ENCUESTA DE DIAGNÓSTICO INICIAL PARA LA ELABORACIÓN DEL
PROTOTIPO

- 1. Seleccione a qué grupo pertenece.**
 - a. Comunidad Universitaria
 - b. Persona Externa de Comunidad Universitaria
- 2. ¿Con qué frecuencia usted utiliza aplicaciones móviles?**
 - a. Muy frecuentemente
 - b. Frecuentemente
 - c. Ocasionalmente
 - d. Raramente
 - e. Nunca
- 3. ¿Es buena idea el desarrollo de una aplicación móvil que permita detectar los espacios de estacionamiento disponibles y ocupados en la Facultad de Ingeniería Civil de la UTMACH?**
 - a. Sí
 - b. No

En caso de que se llegue a desarrollar la aplicación móvil:

- 4. ¿Cree usted que la app debería presentar una pantalla inicial para su descripción, el lugar de aplicación y la capacidad de parqueo?**
 - a. Sí
 - b. No
- 5. ¿Cree usted que la app debería tener una opción para visualizar el área de parqueo mediante una cámara IP?**
 - a. Sí
 - b. No
- 6. ¿Le gustaría que la app muestre el total de espacios disponibles de parqueo antes de visualizar la cámara IP?**
 - a. Sí
 - b. No
- 7. ¿Le gustaría que la app visualice los espacios disponibles y ocupados en el área de parqueo?**

- a. Sí
- b. No

8. ¿Le gustaría que la app permita recargar el Streaming?

- a. Sí
- b. No

9. ¿Le gustaría que la app pregunte si desea cerrar la aplicación?

- a. Sí
- b. No

Anexo 3. Encuesta final para la evaluación del prototipo

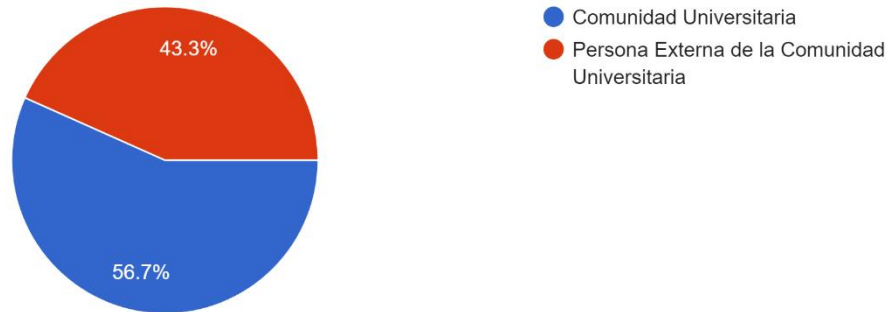
ENCUESTA FINAL PARA LA EVALUACIÓN DEL PROTOTIPO

- 1. ¿La aplicación presenta una pantalla inicial para su descripción, el lugar de aplicación y la capacidad de parqueo?**
 - a. Sí
 - b. No
- 2. ¿La aplicación tiene una opción para visualizar el área de parqueo mediante una cámara IP?**
 - a. Sí
 - b. No
- 3. ¿La aplicación muestra el total de espacios disponibles de parqueo antes de visualizar la cámara IP?**
 - a. Sí
 - b. No
- 4. ¿La aplicación visualiza los espacios disponibles y ocupados en el área de parqueo?**
 - a. Sí
 - b. No
- 5. ¿La aplicación permite recargar el Streaming?**
 - a. Sí
 - b. No
- 6. La aplicación pregunta, “¿Desea cerrar la aplicación?”**
 - a. Sí
 - b. No

Anexo 4. Tabulación de la Encuesta de diagnóstico inicial para la elaboración del prototipo

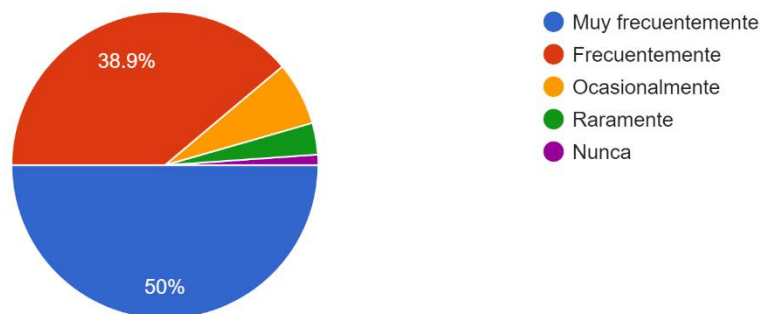
1. Seleccione a que grupo pertenece

90 respuestas



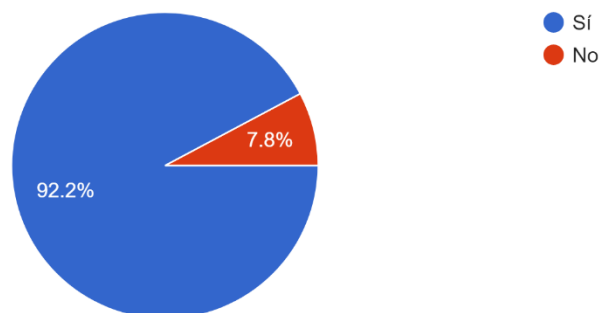
2. ¿Con qué frecuencia usted utiliza aplicaciones móviles?

90 respuestas



3. ¿Es buena idea el desarrollo de una aplicación móvil que permita detectar los espacios de estacionamiento libres y ocupados en la Facultad de Ingeniería Civil de la UTMACH?

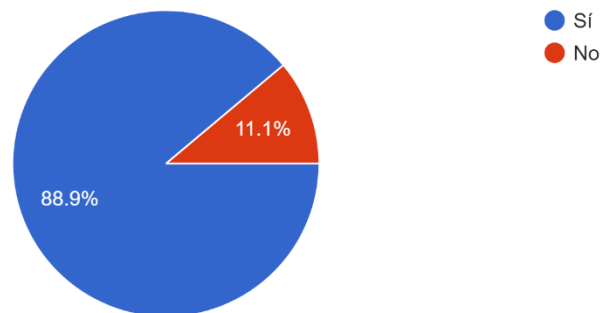
90 respuestas



En caso de que se llegue a desarrollar la aplicación móvil:

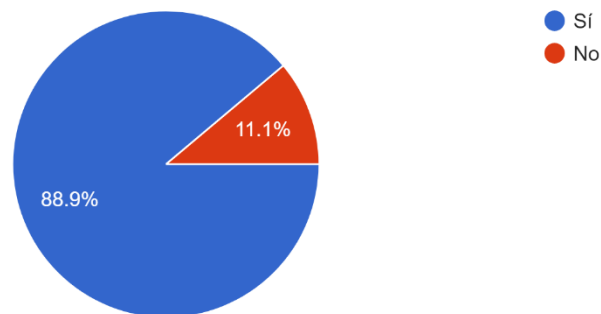
4. ¿Cree usted que la app debería presentar una pantalla inicial para su descripción, el lugar de aplicación y la capacidad de parqueo?

90 respuestas



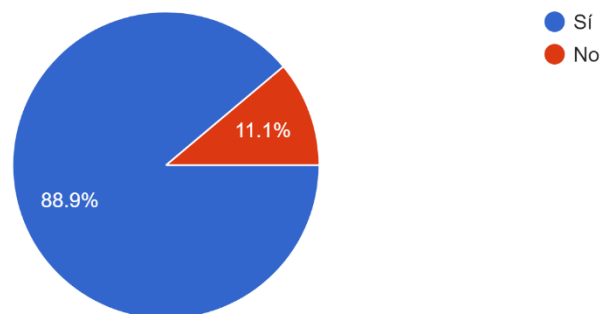
5. ¿Cree usted que la app debería tener una opción para visualizar el área de parqueo mediante una cámara IP?

90 respuestas

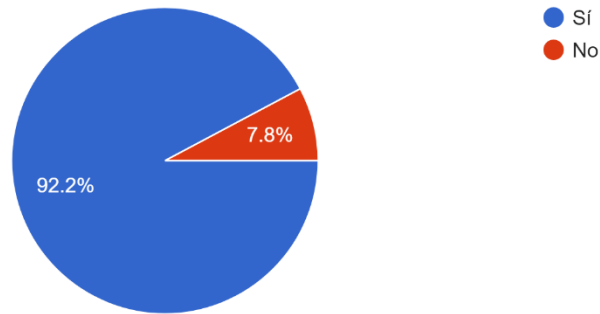


6. ¿Le gustaría que la app muestre el total de espacios disponibles de parqueo antes de visualizar la cámara IP?

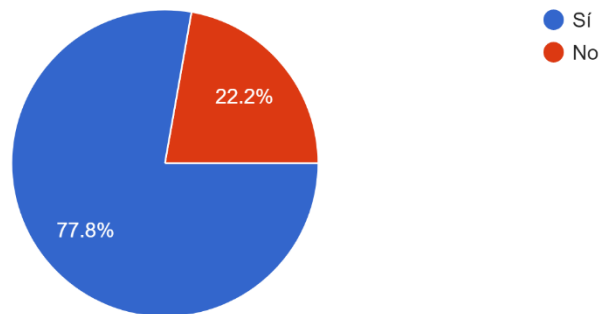
90 respuestas



7. ¿Le gustaría que la app visualice los espacios disponibles y ocupados en el área de parqueo?
90 respuestas



8. ¿Le gustaría que la app permita recargar el Streaming?
90 respuestas



9. ¿Le gustaría que la app pregunte si desea cerrar la aplicación?
90 respuestas

