Nombre y apellidos del estudiante
Título del Trabajo Fin de Estudios


Universidad Internacional de La Rioja
Escuela Superior de Ingeniería y Tecnología

Máster Universitario en Ingeniería de Software y Sistemas Informáticos

# Crowdsourcing y datos libres mediante la detección de vehículos y pedestres utilizando redes neuronales convolucionales en smartphones.

| | |
|---|---|
| Trabajo fin de estudio presentado por: | Carlos Andrés Heredia Álvarez |
| Tipo de trabajo: | Desarrollo práctico |
| Director/a: | GARCIA BORGOÑON LAURA |
| Fecha: | 15/11/2021 |

# Resumen

Hoy en día la propuesta y diseño de soluciones en cualquier área de estudio se basa en la premisa de la disponibilidad de datos para su análisis, en el caso de los Sistemas de Transporte Integrado (ITS), la disponibilidad de datos de libre acceso a la comunidad es escaso o se limita a ciertos grupos del sector privado, los cuales no publican repositorios con los datos para su libre consulta.

Sin el acceso a datos abiertos de movilidad muchos grupos de investigación o personas interesadas en la temática se ven imposibilitadas de efectuar estudios que permitan generar soluciones a los problemas de tráfico actuales, si bien hoy en día existen números métodos basados en dispositivos tecnológicos que permiten generar datos de movilidad en lo referente a conteos vehiculares y reportes de tráfico, el acceso a dichos elementos se ve limitado por su costo de adquisición, además dichos dispositivos son estáticos y únicamente reportan datos del lugar en el cual fueron instalados.

En este contexto el presente trabajo presenta la implementación de un Sistema de generación y consulta de datos de movilidad (conteos vehiculares) completo que comprende las etapas de generación, reporte y consulta de conteos georreferenciados en tiempo real mediante la creación de un aplicativo móvil que hace uso de un algoritmo de detección de objetos basado en redes neuronales convolucionales para la detección de vehículos (coches, motocicletas, peatones y trasporte pesado) en la carretera mediante la cámara del teléfono y la implementación de un aplicativos Web basado en mapas para la consulta de la información reportada. Con los datos generados se efectúa la implementación de un sistema “crodwsourcing” de datos y colaboración abierta distribuida como elemento esencial en la participación y cultura ciudadana. El crodwsourcing es un enfoque nuevo que facilita intercambiar ideas y diferentes puntos de vista entre expertos y la comunidad con la finalidad de dar solución a un problema. La idea principal es generar una plataforma de uso libre en la cual grupos de investigación o interesados puedan consultar y ser parte del proceso de generación de información en lo referente a conteos vehiculares.

**Palabras clave**: crodwsourcing, red neuronal, movilidad, conteos.

# Abstract

Today the proposal and design of solutions in any area of study is based on the premise of the availability of data for analysis, in the case of Integrated Transportation Systems (ITS), the availability of free accessible data to the community is scarce or is limited to certain groups in the private sector, which do not publish repositories with the data for free consultation.

Without access to open mobility data, many research groups or people interested in the subject are unable to carry out studies that allow generating solutions to current traffic problems, although today there are numerous methods based on technological devices that allow generating mobility data in relation to vehicle counts and traffic reports, access to these elements is limited by their acquisition cost, in addition, these devices are static and only report data of the place where they were installed.

In this context, the present work presents the implementation of a complete mobility data generation and consultation system (vehicle counts) that includes the generation, reporting and consultation stages of georeferenced counts in real time through the creation of a mobile application that makes use of an object detection algorithm based on convolutional neural networks for the detection of vehicles (cars, motorcycles, pedestrians and heavy transport) on the road through the camera of the phone and the implementation of web applications based on maps for the consultation of the information reported. With the data generated, the implementation of a "crodwsourcing" system of open data and distributed open collaboration is carried out as an essential element in citizen participation and culture. Crodwsourcing is a new approach that facilitates the exchange of ideas and different points of view between experts and the community in order to solve a problem. The main idea is to generate a platform for free use in which research groups or interested parties can consult and be part of the process of generating information regarding vehicle counts.

**Keywords**: crodwsourcing, neural network, mobility, counts

# Índice de contenidos

# Índice de figuras

# Índice de tablas

# 1. INTRODUCCIÓN

Actualmente existen numerosos estudios que incluyen a la movilidad, el transporte y uso de espacios públicos por parte de los peatones y vehículos. La información y datos de movilidad permiten solucionar problemas relacionados con la congestión de tráfico, el mal uso de espacios destinados a circulación vehicular o peatonal, uso incorrecto de las carreteras, gestión de espacios destinados a ciclistas, carencia de transporte público urbano, entre otros. En la gran mayoría de ciudades no existe información que detalle conteos de flujos vehiculares en diferentes puntos y si existe no se encuentra disponible o su acceso está restringido a determinadas entidades públicas o al sector privado que la utiliza con fines diversos. El no disponer de repositorios de acceso libre a información de movilidad limita en gran medida las tareas de diferentes grupos de investigación y personas interesadas en la generación de soluciones y propuestas que permitan alcanzar una movilidad sostenible dentro de las ciudades. Algunos estudios proveen información que muestra que el uso del espacio público en la mayoría de las ciudades se encuentra mal utilizado y representan espacios inseguros debido a su mala distribución y uso por parte de vehículos y peatones (Stevens, 2009).

Sin embargo, en el paso de los años ha surgido una corriente denominada Crowdsourcing (Raju, Harinishree, & Lavanya, 2017) que promueve la creación de repositorios con datos de libre acceso y cuya generación está a cargo de personas voluntarias e interesadas en una temática específica, el termino fue fundado en el año 2006 por el periodista Jeff Howe, editor de la revista electrónica Wired y abarca los palabras crowd, que significa multitud y sourcing cuyo significado es abastecimiento (Moreno, n.d.). Crowdsourcing brinda la posibilidad de que una tarea compleja y costosa que requiere ser realizada por un grupo especializado de personas, pueda ser llevada a cabo mediante una convocatoria abierta por un gran número de voluntarios denominados crowdworkers (Levannoza et al., 2020).

En este contexto las ciudades inteligentes y los sistemas de transporte integrados (ITS) requieren recopilar, procesar e integrar información sobre el estado de la infraestructura de la vialidad pública. Sin embargo, recopilar información que involucre conteos volumétricos de

vehículos y peatones en carreteras, intersecciones o espacios públicos de forma manual es un trabajo tedioso, costoso y que requiere demasiado tiempo por parte de los interesados.

La automatización de estos procesos de conteo requiere hardware y software específicos a un precio muy elevado, algunas guías como (Board, of Sciences Engineering, & Medicine, 2014) proveen información sobre métodos y sensores disponibles en el mercado para la recolección de datos y conteo de peatones, bicicletas y automóviles en carreteras o espacios públicos. La mayoría de estos sistemas centran esfuerzos en la tecnología para conteos automatizados en video debido a que esta técnica resulta menos intrusiva y requiere mínima intervención humana para su instalación en comparación con otras tecnologías de infraestructura como los bucles de inducción o los contadores de tubos neumáticos (Fernández, n.d.).

En la actualidad los smartphones cuentan con una gran capacidad para almacenar y procesar datos e incluyen una cámara para la captura de video, además disponen de conectividad a la red lo que permite enviar la información generada por los algoritmos de conteo a una API o repositorio de datos para su posterior consulta.

Al hablar de una iniciativa de crowdsourcing para el conteo de vehículos y peatones se debe descartar todo dispositivo que no sea de uso común por parte de los usuarios interesados en colaborar con la generación de datos, por lo tanto, los dispositivos de conteo comercializados por diversas empresas y cuyo uso está sujeto a pagos y limitaciones por parte del proveedor no pueden ser considerados. Otra ventaja del uso de smartphones es la posibilidad de efectuar el procesamiento de video en el mismo dispositivo en lugar de utilizar la nube o servidores intermedios, este procesamiento se denomina de borde o “edge computing” (Wei et al., 2017).

## 1.1. Justificación

El uso inadecuado de los espacios públicos en muchas ciudades representa un problema de gran relevancia en la actualidad, se han realizado estudios (Maitre, 2017), que demuestran que gran parte de los accidentes de tránsito y congestiones vehiculares en las ciudades se

deben a diseños viales inadecuados, aglomeración de vehículos o peatones en zonas concretas de intersecciones viales, poca o nula existencia de carriles dedicados a la circulación de vehículos pesados o ciclistas.

Muchas de las soluciones a estos problemas se basan en el análisis de flujo vehicular y peatonal dentro de las zonas de conflicto, para lo cual se requieren datos que permitan describir la afluencia vehicular en diferentes momentos del día, en este sentido el conteo de actores de movilidad realizado manualmente por personas demanda una cantidad de recursos elevada , la cual muchas de las veces las autoridades encargadas de la seguridad vial no están dispuestas pagar, es por ello que una alternativa de tecnológica y libre acceso puede generar datos de utilidad en este contexto.

Con los datos generados se propone la creación de un sistema “crodwsourcing” (Liang, Li, & Luo, 2016), de datos y colaboración abierta con la comunidad como parte de un proceso de participación ciudadana. El crodwsourcing es un enfoque nuevo que facilita intercambiar ideas y diferentes puntos de vista entre expertos y la comunidad con la finalidad de dar solución a un problema. Mediante la presente propuesta se busca generar la información necesaria para dar solución a la problemática de la congestión de tráfico, uso inadecuado de espacios públicos para bicicletas, peatones y la carencia de alternativas de transporte público urbano en una ciudad intermedia.

En este contexto el objetivo de este TFM consiste en implementar un aplicativo móvil que haga uso de algoritmos de detección de objetos basados en redes neuronales convolucionales para detectar y contar vehículos (coches, motocicletas, trasporte pesado) y pedestres en la carretera, el resultado de estos conteos será enviado a una plataforma Web (API), que almacenará los datos junto con su ubicación de captura en forma de series de tiempo los cuales podrán se accedidas para su consulta de forma libre dando lugar a la creación de un sistema “crodwsourcing” de datos abiertos y colaboración abierta distribuida.

## 1.2. Estructura del trabajo

En capitulo II se analizarán trabajos relacionados con la temática de movilidad sostenible describiendo los principales aportes en el área, se revisarán trabajos que describen técnicas de conteo de actores de movilidad, así como métodos y algoritmos utilizados para la detección de objetos en secuencias de video, se describen implementaciones existentes resaltando sus ventajas y desventajas en la solución de conteos de actores de movilidad.

Se detallan trabajos relacionados que describen técnicas clásicas de aprendizaje máquina utilizadas comúnmente para la clasificación|n de imágenes frente a técnicas actuales de aprendizaje máquina basadas en redes neuronales convoluciones. Se analizan trabajos que involucran iniciativas de crowdsourcing para la recolección de datos y generación de repositorios de colaboración abierta en la comunidad, finalmente en ese apartado se presenta una tabla de resumen con los trabajos más relevante detectados.

En el capítulo III se presenta el contexto en el cual se va a desarrollar el trabajo, el objetivo general y específico que se pretende alcanzar y finalmente la metodología de trabajo a realizar para lograr los objetivos.

El Capítulo IV proporciona el desarrollo específico de la contribución, que define, explica los requisitos y las tecnologías a utilizar en el proceso, se explica la organización planteada para el desarrollo, las técnicas empleadas y se efectúa una descripción del software que detalla los pasos y los hitos del proceso. Además, se exponen ilustraciones de la arquitectura y funcionamiento de la solución propuesta.

Finalmente, en el capítulo V se presenta el resumen final del trabajo destacando el alcance y relevancia del aporte realizado. En este apartado se describe las líneas de trabajo futuro que podrían aportar valor añadido al trabajo, se señalan las perspectivas de futuro que abre el trabajo desarrollado para el campo de estudio definido.

# 2. CONTEXTO Y ESTADO DEL ARTE

Este capítulo está dividido en tres secciones. En la sección I, se describen las nociones básicas y la evolución del reconocimiento de imágenes digitales haciendo énfasis en la descripción de diversos algoritmos de detección de objetos, se define la terminología básica en el ámbito de aprendizaje profundo y el empleo de redes neuronales con el fin de obtener información de una imagen. Finalmente, en la sección II, se estudian investigaciones destacadas relacionadas con este trabajo.

## 2.1. Evolución del reconocimiento de imágenes.

En la década pasada, los métodos de detección de objetos seguían el paradigma de extracción de características utilizando el concepto de clasificador. Con este enfoque se debe definir deliberadamente una característica específica para una categoría de objetos con el fin de representar el objeto con precisión. Después de extraer suficientes características del conjunto de datos de entrenamiento, el objeto se puede representar mediante un vector de características, que se usa para entrenar a un clasificador y también se puede usar para realizar la tarea de detección en el tiempo de la prueba. En caso de requerir un detector de múltiples objetos, se debe escoger una característica general de entrenamiento que pueda adaptarse a diferentes objetos. La desventaja es obvia ya que la definición de la característica es muy compleja y el modelo es difícil de extender cuando se agrega un nuevo objeto a la lista de detección. Además, la precisión de detección es insatisfactoria (Real et al., 2017).

Hoy en día el aprendizaje profundo ha superado los límites de lo que era posible en el dominio del procesamiento de imágenes digitales. Sin embargo, eso no quiere decir que las técnicas tradicionales de visión por computadora, que habían experimentado un desarrollo progresivo en los años anteriores al surgimiento del DL (Deep Learning), se hayan vuelto obsoletas. Este trabajo analizará los beneficios y los inconvenientes de cada enfoque.

### 2.1.1. Técnicas tradicionales para la detección de objetos.

La visión por computadora se puede describir como la técnica que permite encontrar características a partir de imágenes con la finalidad de discriminar objetos o clases de objetos.

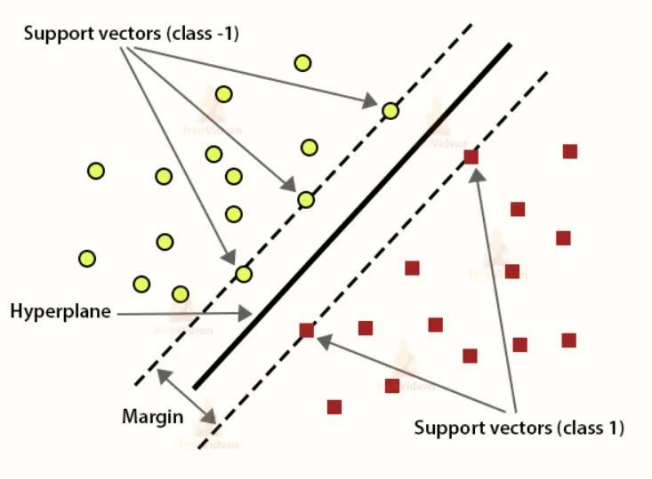
Los algoritmos de visión por computadora en general funcionan extrayendo vectores de características de imágenes y usando estos vectores de características para clasificar imágenes. En este sentido los enfoques tradicionales de visión por computadoras se remontan a los últimos 10 a 20 años y se destacan por extraer características de la ingeniería humana como: bordes, esquinas, colores, entre otras cualidades que añaden un valor significativo en las tareas de visión. Es decir, estas técnicas están impulsadas desde un enfoque humano.

A continuación, se describen algunas técnicas empleadas en la visión por computadora tradicional. El propósito de estas metodologías es formular distintas maneras de representar la imagen codificando varias características de la misma, como: esquinas, esquemas de colores, textura de la imagen, etc.

#### 2.1.1.1. SVM (Super Vector Machine)

SVM pertenece a la familia de clasificadores supervisados. Requiere muestras de entrenamiento dadas por un supervisor. A diferencia de una red neuronal artificial (ANN), es menos sensible al tamaño y la cantidad de datos de entrenamiento. Originalmente, SVM se desarrolló como clasificador binario donde asigna solo dos etiquetas posibles; -1 y 1 en una muestra de prueba determinada. Básicamente, el algoritmo de entrenamiento tiene como objetivo construir el hiperplano de separación (por ejemplo, límite de decisión) basado en las propiedades de las muestras de entrenamiento (o específicamente, su distribución en el espacio de características) (Yang, Wang, & Yang, 2012). En otras palabras, este hiperplano actúa como separador entre las dos clases de muestras. La ilustración del hiperplano se muestra en la Figura 1. El algoritmo funciona básicamente maximizando el ancho del margen del hiperplano de separación (entre la clase -1 y 1), por lo que se optimiza la distancia máxima entre las clases. En la construcción del hiperplano, no necesariamente todas las muestras están contribuyendo, sino solo un subconjunto de ellas que se elige como vector de soporte.

**Figura 1.** *Hiperplano en SVM*



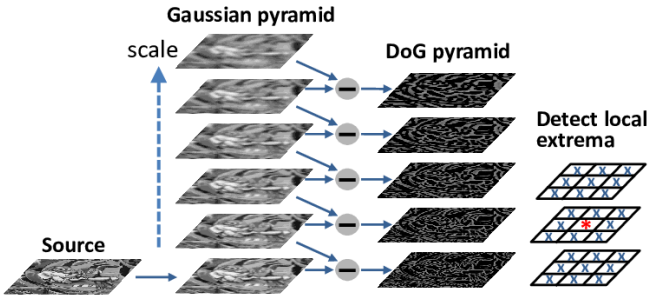

Fuente: (Tutorials, 2018)

#### 2.1.1.2. SIFT (Transformación de características invariantes a la escala)

El algoritmo SIFT resuelve el problema de SVM en el sentido de que ciertas características de la imagen, como los bordes y las esquinas, no son invariantes en la escala. En otras palabras, hay ocasiones en las que una esquina en una imagen parece una esquina con la imagen en tamaño real, pero se interpreta como un elemento completamente diferente cuando la imagen se ve agrandada por algunos factores. Este algoritmo hace uso de aproximaciones matemáticas que crean una imagen que no varía en escala, ajustando todas las imágenes a un mismo estándar, por ejemplo; si la imagen está agrandada, SIFT la encoge; o si la imagen está encogida, SIFT la agranda. Es decir, si alguna característica es capturada en una imagen médiate el uso de una ventana cuadrada de dimensión m en los pixeles, si esta imagen tuviera una escala más grande deberá usarse una dimensión mayor k x m para detectar la misma característica.

El algoritmo SIFT tiene como finalidad estandarizar la escala de la imagen con el objeto de detectar características importantes. Posteriormente estas características son codificadas por un vector empelado para personificar la imagen. La Figura 2 presenta el esquema general de trabajo de SIFT (G. Wang, Rister, & Cavallaro, 2013).

**Figura 2.** *Diagrama del algoritmo de detección SIFT con procesamiento Gaussinao.*



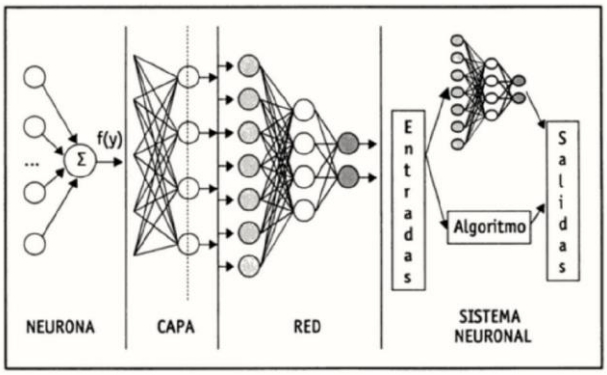

Fuente: (J. Wang, Jiang, Song, & Yang, 2019)

### 2.1.2. Técnicas basadas en redes neuronales para la detección de objetos

Según (Raquel Flórez López, Jose Miguel Fernandez, 2008) las redes neuronales artificiales son un modelo matemático formado por múltiples nodos que intenta imitar el sistema nervioso humano. Estas redes emulan las neuronas biológicas del cerebro humano que interactúan entre sí y se conectan por enlaces, mediante los nodos que toman los datos de entrada efectuando operaciones sencillas y transmitiendo el resultado a otras neuronas. La salida en cada nodo se denomina activación o valor de nodo.

Cada vínculo está asociado con un peso, permitiéndole a la red neuronal aprender, esto se lleva a cabo modificando los valores de los pesos asociados. La Figura 3, muestra la estructura de una ANN (Red Neuronal Artificial) simple:

**Figura 3**. *Estructura de una red neuronal artificial*




Fuente: (Flórez & Fernandez, 2008)

Las capas están formadas por nodos y un nodo es donde ocurre el proceso computacional. Los nodos se encargan de combinar los datos de entrada con un conjunto de coeficientes que amortiguan esa entrada, atribuyendo importancia a las entradas con respecto a la tarea que el algoritmo intenta aprender. Los resultados de la ponderación obtenidos en la entrada son sumados, después el producto de esta suma es transferido por medio de la llamada función de activación de nodos, esto permitirá determinar en qué medida la señal debe avanzar a través de la red para modificar el producto final.

#### 2.1.2.1. Deep Learning o aprendizaje profundo.

El Deep Learning es un subconjunto de aprendizajes automáticos. El Deep Learning se basa principalmente en redes neuronales artificiales (ANN), un paradigma informático inspirado en el funcionamiento del cerebro humano. El aprendizaje profundo tiene que ver con el aprendizaje en múltiples capas de una red neuronal de manera precisa, eficiente y sin supervisión. En comparación con las técnicas tradicionales de visión por computadora, Deep learning permite lograr una mayor precisión en tareas como clasificación de imágenes, segmentación semántica, detección de objetos, localización y mapeo simultáneo de patrones. Dado que las redes neuronales que se utilizan en Deep Learning están entrenadas en lugar de programadas, las aplicaciones que utilizan este enfoque a menudo requieren un análisis menos experto y un ajuste fino y explotan la enorme cantidad de datos de video disponibles en los sistemas actuales. Adicionalmente el aprendizaje profundo proporciona una flexibilidad superior porque los modelos y marcos de CNN (Convolutional Neural Network) se pueden volver a entrenar utilizando un conjunto de datos personalizado para cualquier caso de uso, a diferencia de los algoritmos de computer vision, que tienden a ser más específicos del dominio.

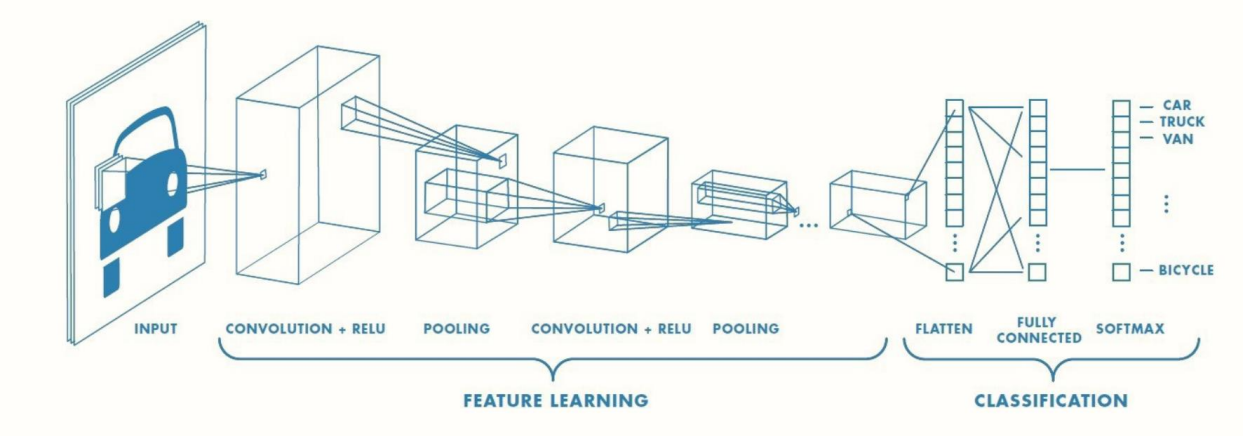
#### 2.1.2.2. Redes neuronales Convolucionales (CNN)

Una red neuronal convolucional (ConvNet o CNN) se define como un algoritmo de aprendizaje profundo que hace uso de una imagen de entrada, estableciendo valores a los diferentes objetos de la imagen, distinguiendo uno del otro. El procesamiento previo necesario en una ConvNet es más simple en comparación a otros algoritmos de clasificación. En las técnicas

primitivas los filtros son esquematizados de forma manual, en cambio, las ConvNets implementan funcionalidades que realizan estos procesamientos de forma automática.

La ConvNet fue creada basándose en la organización de la corteza visual y cuenta con una estructura similar a la del patrón de conectividad de las neuronas del cerebro. Las neuronas individuales reconocen los estímulos únicamente en una región específica del campo visual denominada como campo receptivo. Toda el área visual está cubierta por un conjunto de estos campos (Choi, Hong, & Kim, 2016). La Figura 4 presenta la distribución de una red neuronal convolucional simple.

**Figura 4.** *Estructura de una red neuronal Convolucional*

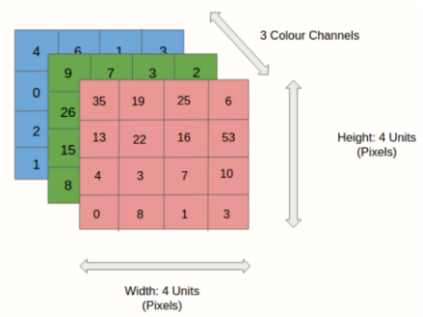



Fuente: (Saha, 2018)

- Imagen de entrada.

En la Figura 5, se presenta una imagen RGB separada en sus tres planos de color: rojo, verde y azul. La función del ConvNet es simplificar el proceso de reducción de imágenes evitando la perdida de sus características que son esenciales para la obtención de buenos resultados. Es importante crear una arquitectura funcional que permita aprender características y que sea escalable a conjuntos de datos masivos (Saha, 2018).

**Figura 5.** *Imagen RGB de entrada de 4x4x3 pixeles*




Fuente: (Saha, 2018)

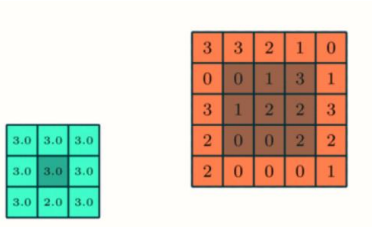
- Capa convolucional o kernel

Un kernel es solo una matriz de valores, llamados pesos, que están entrenados para detectar características específicas. Como su nombre lo indica, la idea principal detrás de las CNN es convolucionar espacialmente el kernel en una imagen de entrada determinada para verificar si la característica que debe detectar está presente (Raquel Flórez López, Jose Miguel Fernandez, 2008).

- Capa de Pooling

La capa de pooling se asemeja a una capa convolucional y es la encargada disminuir el tamaño espacial de la entidad convolucionada, al reducir el tamaño, puede reducir la potencia de procesamiento necesaria para procesar los datos. También, es de gran utilidad para obtener características dominantes que son fijas rotacionales y posicionales, conservando el proceso de adiestramiento efectivo del modelo (Saha, 2018). La Figura 6 presenta el proceso de pool aplicado a una función convolucionada de 5x5.

**Figura 6.** *Agrupación 3x3 sobre función convolucionada 5x5.*



Fuente: (Saha, 2018)

- Función de Activación ReLu

La función lineal de activación ReLU permite formar una entrada directamente si es positiva, en el caso de ser negativa generará cero. Actualmente es la función de activación predeterminada en varios tipos de redes neuronales debido a que es un modelo simple de entrenar y con frecuencia se obtiene un mejor rendimiento.

## 2.2. Estudios relacionados

Durante la última década se ha podido evidenciar un incremento en el uso de técnicas basadas en inteligencia artificial para el desarrollo de tareas antes echas por humanos, hoy en día en el mercado existen aplicaciones para desbloqueo facial en dispositivos móviles, computadoras e inclusive línea blanca. Entre las grandes empresas que han incrementado sus ganancias haciendo uso de estas técnicas se encuentra Google , la cual implementa algoritmos de inteligencia artificial en casi todos sus servicios, los cuales van desde el asistente de Gmail, Google Fotos, asistente de mapas, Google Lens, buscadores de texto, entre otros (Tecnologica, n.d.). En este contexto existen algoritmos de inteligencia artificial que permiten efectuar tareas de detección de objetos en imágenes estáticas o secuencia de videos, las aplicaciones de este tipo de tecnología son diversas, por ejemplo en (Shuai & Wu, 2020) los autores proponen un algoritmo para la identificación de objetos en tiempo real basado en el modelo de red neuronal convolucional denominado Single Shot MultiBox Detector (SSD), la detección se efectúa sobre objetos de uso cotidiano y tiene como finalidad medir la precisión en la detección por modelo neuronal.

Pero no solo las redes neuronales permiten efectuar tareas de detección de objetos, antes de las Neural Networks (NN) otros tipos de métodos de clasificación eran los más utilizados en el contexto de Machine Learning. Tomando como referencia, por ejemplo, Support Vector Machine (SVM) como se describe en (Cortes & Vapnik, 1995), este método utiliza vectores de tamaño estándar para realizar la clasificación. Un ejemplo práctico de implementación de esta técnica se efectúa en (Ma & Chia, 2007), donde los autores utilizan SVM en lugar de redes neuronales convolucionales para tareas de detección y clasificación de patrones a partir de

un conjunto de imágenes de sonda en un proceso de barrido lateral utilizado en tareas de contramedida automática de minas.

Las aplicaciones de detección de objetos son innumerables y pueden ir desde aplicaciones médicas para la detección de tumores en imágenes (Cen, Pan, Li, & Ding, 2019) hasta la detención de vegetación en imágenes satelitales (Wu & Zhang, 2018), el ámbito de interés del presente trabajo centra la atención en la aplicación de técnicas que se basan en redes neuronales para la detección de vehículos y peatones en la carretera.

### 2.2.1. Crowdsourcing para la generación de datos de uso libre.

Es importante destacar que dentro de una iniciativa de Crowdsourcing se pueden considerar diferentes tipos, a pesar de que no existe una clasificación universal, los autores clasifican los proyectos crowdsourcing de distintas maneras. En la Figura 7 se presenta una clasificación propuesta por Howe-Estellés (Howe & Jeff, 2006), en la cual se presenta categorías en función del tipo de aplicación a implementar.

**Figura 7.** *Comparativa de la clasificación de los tipos de crowdsourcing.*

<table>
<tr><th colspan="2">Howe</th><th colspan="2">Estellés</th></tr>
<tr><td colspan="2">Crowdfunding</td><td colspan="2">Crowdfunding</td></tr>
<tr><td colspan="2">Crowdcreation</td><td rowspan="3">Crowdcontent</td><td>Crowdproduction</td></tr>
<tr><td colspan="2" rowspan="2"></td><td>Crowdsearching</td></tr>
<tr><td>Crowdanalyzing</td></tr>
<tr><td rowspan="3">Crowdwisdom</td><td>Crowdstorming</td><td rowspan="2">Crowdcollaboration</td><td>Crowdsupport</td></tr>
<tr><td>Crowdcasting</td><td>Crowdstorming</td></tr>
<tr><td>Predicción de mercado</td><td colspan="2">Crowdcasting</td></tr>
<tr><td colspan="2">Crowdvoting</td><td colspan="2" rowspan="2">Crowdopinion</td></tr>
</table>

Fuente: (Howe & Jeff, 2006)

Acorde a la Figura 7, las definiciones a considerar en crowdsourcing son los siguientes:

1. Crowdvoting: proceso que toma la opinión de la comunidad para efectuar tareas de organización, filtrado y clasificación jerárquica de contenido en concursos de ideas o grandes encuestas.

2. Crowdwisdom: considera la opinión colectiva de una asociación de individuos en lugar de una única opinión de un experto para dar respuesta a un determinado problema o pregunta.

3. Crowdcasting: es una táctica de resolución de problemas y generación de ideas en la que una corporación difunde detalles de un problema o situación específica a un grupo de personas cuidadosamente elegido para posibles soluciones.

4. Crowdcollaboration: abarca todas las iniciativas en las que se efectúa un proceso de comunicación entre individuos en la comunidad, en tanto la corporación iniciadora del proceso permanece al margen. Los colaboradores aportarán conocimiento para resolver problemas de forma colaborativa, puede ser de dos tipos: crowdstorming que cosiste en una tormentas de ideas para dar solución a un problema y crowdsupport que engloba las iniciativas en las cuales los clientes solventan dudas o incidencias gracias a la colaboración activa de la comunidad sin necesidad de recurrir a la compañía involucrada (Díaz, 2014).

5. Crowdcontent: abarca todas las actividades en las cuales la comunidad aporta con mano de obra y conocimiento de forma activa para para la creación de contenido de diferente naturaleza. Se distingue de crowdcasting en que no consiste en un reto y se basa en el trabajo individual de los colaboradores remiendo al final el resultado de todos.

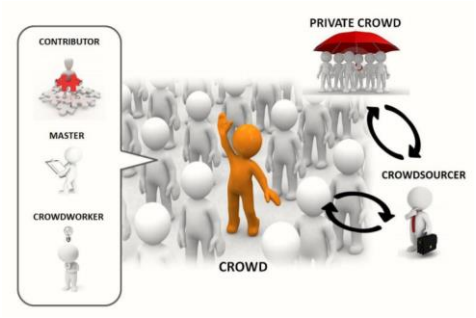
### 2.2.2. Agentes participantes en crowdsourcing

Del mismo modo que en cualquier tipo de proyecto está conformado por sus stakeholders, en las iniciativas y proyectos de crowdsourcing los agentes participantes desempeñan diferentes roles. En función de la tarea desarrollada se dispone de diferentes roles. A continuación, se describen los más relevantes a tener en consideración:

- **Crowdsourcer**: Es la organización encargada de poner en marcha el proyecto y, en comparación con un proyecto tradicional el trabajo no es ejecutado por agentes con asignación (empleado, contratista…), se lleva a cabo por la comunidad.

- **Crowd:** Se refiere a la comunidad y grupo de individuos a los que se les ofrecerá efectuar una tarea dentro de una iniciativa de crowdsourcing. Suele ser un grupo de carácter heterogéneo y generalmente está integrado por individuos que no se conocen entre sí, usualmente el número de personas en la comunidad variara en función de la iniciativa (Howe & Jeff, 2006).
- **Multitud privada o private crowd:** Conformado por grupos de trabajadores cualificados que desempeñan tareas para una empresa en la modalidad de subcontratados por lo general como trabajadores independientes, mitigando los costes y cargos económicos que envuelven a las grandes plataformas de crowdsourcing (Díaz, 2014).
- **Crowdworker:** Persona que provee cierta cantidad de mano de obra, experiencia, dinero o conocimiento a una iniciativa de crowdsourcing, a cambio de una recompensa por lo general de tipo económico, reconocimiento social o de desarrollo de capacidades personales.
- **Contributor:** Surge en proyectos complejos y de larga duración y participa en conocimiento del proyecto en la interna, además realiza tareas de control y supervisión. En otras implementaciones suele denominarse master, por ende un master es un crowdworker que ha efectuado con éxito una gran cantidad de tareas que demandan inteligencia humana para su resolución (Díaz, 2014).

**Figura 8.** *Agentes participantes en proyectos crowdsourcing*




Fuente: (Díaz, 2014)

La Figura 8 se resume la función desempeñada por los diferentes agentes de crowdsourcing.

### 2.2.3. Estudios relacionados a iniciativas de crowdsourcing aplicado a sistemas de transporte integrado

El número de trabajos que presentan aplicaciones en las cuales los resultados de conteos vehiculares formen parte de una iniciativa de Crowdsourcing para la colaboración libre de datos es reducida, la mayoría de los estudios abarcan otro de tipo de aplicaciones las cuales van desde la recolección de datos de sensores en teléfonos móviles para generar información útil en el análisis de la actividad humana (Mairittha & Inoue, 2019). Otros autores (El Alaoui El Abdallaoui, El Fazziki, Sadiq, Zohra, & Sadgal, 2016), proponen una novedosa aplicación de crowdsourcing mediante la cual la comunidad puede colaborar con información para encontrar a niños reportados como desaparecidos, se emplea una plataforma de datos abiertos en línea. Otro trabajo de interés es propuesto por Muhammad y Zaka (Jilani, Ur Rehman, & Abbas, 2019), los cuales describen un marco de aplicación para un sistema de notificación de eventos de emergencia. El enfoque propuesto se basa en el crowdsourcing que utiliza la detección participativa para aprovechar múltiples usuarios de teléfonos inteligentes para informar colectivamente eventos de emergencia. Los voluntarios que se encuentren cerca de un incidente pueden activar rápidamente una notificación de alerta a los servicios de rescate con información crucial, como las coordenadas de ubicación, el tipo de incidente y el número de víctimas. El sistema procesa esta información y envía una alerta a la ambulancia y otros servicios de rescate con detalles de ruta optimizados y la ubicación de los centros de emergencia más cercanos.

En resumen, las iniciativas de crowdsourcing se presentan como una alternativa de innovación para las empresas. Los crowdworkers al estar conformados por diferentes personas de todo el mundo integran un valioso grupo con un gran potencial que está dispuesto a contribuir constantemente en la mejora y simplificación de procesos, planteando ideas para la creación de productos y contribuyendo con soluciones a retos científicos importantes.

### 2.2.4. Estudio comparativo de Implementaciones

En esta sección se efectúa una comparativa de los instrumentos que dispone el mercado para el conteo de actores de movilidad y la herramienta propuesta en el presente trabajo, se consideraron alternativas de software de uso libre y comercial. En este contexto es importante describir en detalle las principales características del sistema de conteo de actores de movilidad propuesto, lo cual servirá de referencia al momento de compararlo con otro tipo de plataformas.

Para satisfacer los objetivos planteados, la herramienta propuesta se ha diseñado con la finalidad de cumplir los siguientes criterios de operación y funcionamiento:

- **Interfaz Simple**: El diseño del aplicativo es simple, intuitivo y de fácil uso, pretende que el usuario en pocos pasos sea capaz de generar datos de conteos vehiculares en tiempo real haciendo del procesamiento, detección y envió de información a la plataforma de visualización transparente al usuario.
- **Datos abiertos**: La finalidad de la propuesta es generar un repositorio con datos de libre acceso para investigadores, sector público o privado e interesados en temas de movilidad mediante una plataforma de visualización de los conteos gratuita y sin restricciones.
- **Plataforma Modular**: El sistema propuesto consta de tres módulos independientes para su operación: para el manejo de peticiones y datos se utiliza un servicio web RestFul, la interfaz Web de visualización de datos utiliza React JS en la parte del cliente y la aplicación móvil para la generación de conteos fue desarrollada para teléfonos basados en Android.
- **Clasificación de Vehículos**: La solución propuesta es capaz de clasificar los conteos en 4 tipos de categorías de movilidad: camiones, automóviles, motos/bicicletas y peatones.
- **Exactitud en los conteos**: La red neuronal empleada se basa en el modelo SSD-Mobilenet (TensorFlow Hub, 2021), el cual provee soporte para la detección de vehículos y peatones en la vía pública, como se detalla en (Heredia & Barros-

Gavilanes, 2019), en condiciones de tráfico normal la precisión en los conteos provista por modelo puede ir del 91% al 97%.

- **Desempeño adecuado en móviles de gama media-baja**: La aplicación de conteo vehicular se ejecuta de forma fluida en teléfono móviles de gama media-baja, puesto el modelo de red neuronal esta optimizado para este tipo de dispositivos.
- **Desarrollo distribuido basado en la comunidad:** El presente trabajo pretende fomentar las bases para la generación de datos abiertos como parte de una iniciativa de Crowdsourcing en la cual la comunidad se la encargada de suministrar y frenar la información en los puntos de interés.
- **Entorno ubicuo:** Con la implementación de este sistema se pretende vincular el uso de la tecnología móvil e influir en la vida de los usuarios fomentando una cultura de colaboración en el entorno de la movilidad.
- **Diversidad geográfica de datos:** Gracias a la implementación del sistema de conteo en dispositivos móviles y la iniciativa de colaboración abierta (CrowdSourcing) por parte de la comunidad, es posible recolectar datos de conteos vehiculares de cualquier parte del mundo y georreferenciarlos en la plataforma de visualización.
- **Cero costos:** Los datos generados por la comunidad en el contexto del presente trabajo se encontrarán disponibles y accesibles a cualquier persona o grupo de investigación interesado.

Actualmente en el mercado se pueden encontrar soluciones comerciales para la ejecución de conteos vehiculares en la carretera, tal es el caso de la empresa TRIGG (TRIGG, 2021), la cual provee sensores especializados en el conteo de vehículos en carreteras. La tecnología encargada de efectuar los conteos se basa principalmente en sensores magnéticos y piezoeléctricos, el costo de las implementaciones varía en función del sensor utilizado. Otra empresa que provee soluciones en lo referente a conteos vehiculares es All Traffic Solutions (Solutions, 2021), entre los sistemas promocionados se disponen de tres categorías: StatTrak, la cual incluye un sensor que posibilita la detección de la dirección del movimiento vehicular en autopsitas, QueTrak compuesta por un sensor de video y un software capaz de procesar conteos vehiculares sin efectuar la clasificación de los vehículos detectados y SpeedLane Pro,

que provee detección de alta precisión que incluye dirección, carril, velocidad y clase de vehículos individuales de hasta 16 carriles. Intu Vision es otra empresa del mercado que presta soluciones en el área de conteos vehiculares, dispone de un sistema completo de conteo que incluye sensores especializados con cámaras de video para la captura , procesamiento y conteo de vehículos en carreteras, una plataforma de almacenamiento y visualización de datos de conteo en tiempo real, de acuerdo al proveedor los servicios de conteo se facturan por horas de uso, es un solución completa aunque poco accesible para personas o grupos de investigación que desean efectuar estudios preliminares en el campo de la movilidad.

Productos similares a los descritos en esta sección existen una gran variedad, los cuales comparten una característica en común: todas las empresas basan sus productos en dispositivos electrónicos especializados y de uso propietario con cámaras y sensores de movimiento de alta calidad, lo cual involucra costos elevados para personas o grupos interesados en colaborar con soluciones en el campo de la movilidad, además la diversidad de los datos es demasiado reducida, es decir solo se obtendrán datos de conteos del sector en el cual el sensor fue instalado y por el cual se pagó el servicio a la empresa proveedora.

En este sentido la iniciativa propuesta en este trabajo presenta muchas ventajas, como ejemplo no demanda de ningún dispositivo electrónico especializado ni tampoco de cámaras de video de alta resolución, además la diversidad puede extenderse a nivel mundial puesto que un usuario con el aplicativo móvil puede reportar datos desde cualquier sitio.

No se encontraron alternativas comerciales que utilicen dispositivos móviles para la identificación y conteo de vehículos en carreteras, lo cual tiene sentido desde la perspectiva comercial y financiera de las compañías, puesto la mayoría de ellas busca vender una solución especializada que involucra un hardware y software propietario, el cual debe ser configurado en un sitio determinado para la captura de datos.

En cuanto a soluciones de código libre la mayoría de trabajos son estudios que ofrecen alternativas para la ejecución de conteos vehiculares utilizando técnicas de procesamiento de imágenes tradicionales o redes neuronales, las alternativas disponibles en su mayoría se

encuentran disponibles mediante repositorios abiertos y de libre acceso, entre los trabajos más destacados tenemos: la implementación efectuado por autores como Chen (Chen, Ellis, & Velastin, 2012) los cuales utilizan SVM para la detección de vehículos en la carretera en el contexto de un Sistema de transporte integrado (ITS), en este trabajo los autores presentan los diferentes pasos de un sistema de conteo y clasificación de vehículos a través de SVM incluyendo el procesamiento, extracción de background en el fondo de las imágenes a través de modelos Gaussianos mixtos, la técnica reporta más del 99% de precisión. Sin embargo, los videos sobre los que se aplica el algoritmo corresponden al mismo escenario en todos los casos, esto implica que se debe realizar un entrenamiento específico para cada ubicación con la finalidad de obtener la máxima precisión.

El estudio realizado por Peña y Nuño (Peña-González & Nuño-Maganda, 2014) propone un sistema para el conteo, detección y seguimiento de vehículos, en este trabajo los autores implementan un algoritmo para detectar la Región de interés (ROI) que involucra el proceso de agrupamiento, seguimiento, clasificación por frame y el conteo vehicular. El hardware de prueba fue una PC con procesador Intel Centrino Duo (T7250@2.00GHz), 4 GB de RAM DDR2 y sistema operativo Windows Vista 32-Bits. Esta configuración corresponde a una computadora de baja capacidad.

Otro trabajo relacionado es propuesto por Myung-Cheol Roh en (Roh & Lee, 2017), el cual efectúa la implementación del algoritmo Fast R-CNN con una mejora en la etapa de detección, la cual se argumenta tiene un desempeño pobre en la detección de objetos pequeños. El estudio demuestra que los resultados obtenidos con la modificación aportan mejoras en la detección de personas y vehículos al reducir la detección de falsos positivos (FP). Otro trabajo relevante en el área es descrito en (J. Wang et al., 2019) donde los autores hacen una comparación entre diferentes algoritmos para la detección de objetos. El estudio utiliza redes neuronales convolucionales (CNN) para detectar objetos y proporcionar métricas de precisión en la detección. Los autores consideran tres modelos diferentes: Single Shot Detector (SSD), Faster Region based Convolutional Neural Networks (FR-CNN) y Region based Fully Convolutional Networks (R-FCN). Estos modelos se combinan con diferentes extractores de funciones como VGG, Resnet-101 y Mobilenet para determinar la mejor combinación en

términos de rendimiento y velocidad. Los autores informan que la combinación de Faster R-CNN con Resnet-101 proporciona la mejor precisión para el detector de objetos con un 33,7% de precisión media media (mAP) y 396 ms de tiempo de GPU, mientras que el modelo SSD-MobileNet proporciona un mAP de 19% con un tiempo de GPU de 40ms. Las configuraciones de hardware incluyen una tarjeta GPU Nvidia Titan X que se ejecuta en un dispositivo RAM de 32 GB con un procesador Intel Xeon E5-1650 v2.

En relación al despliegue de una red neuronal convolucional (CNN) en dispositivos embebidos y teléfonos móviles el número de publicaciones que proponen un sistema unificado de detección, seguimiento y conteo es reducido. La mayoría de los estudios se limitan a un único campo de aplicación como la detección o la clasificación utilizando técnicas tradicionales como la extracción de background en imágenes o el uso de métodos como SVM (ver, por estudios, (Peña-González & Nuño-Maganda, 2014), (Chen et al., 2012), (Espinoza, Gabriel, & Barros, 2017)).

El soporte proporcionado para dispositivos embebidos (e, g. Raspberry Pi), teléfonos móviles y elementos electrónicos semejantes para el despliegue de redes neuronales convolucionales se ve afectado por la capacidad de procesamiento de estas plataformas. Para resolver este problema en (OTHMAN & AYDIN, 2018), los autores utilizan un elemento de procesamiento de imágenes auxiliar denominado “Movidius Neural Computer Stick”, el cual mejora el rendimiento en la incorporación de un modelo que identifica obejtos en tiempo real en un dispositivo integrado de bajo consumo (Raspberry Pi 3B). Los resultados muestran que Movidius alcanza 3,5 FPS frente a 0,5 FPS sin el dispositivo.

Los autores citados en la mayoría de los casos utilizan los modelos de detección en computadoras avanzadas que cuentan con GPUs, la razón principal es que los modelos como Faster R-CNN, R-FCN o GoogleLenet demandan un alto procesamiento que no puede ser satisfecho por un dispositivo embebido o teléfono móvil. En este contexto, el modelo más adecuado para el despliegue con dispositivos integrados y teléfonos móviles es SSD-Mobilenet, que está estrictamente diseñado para dispositivos con bajas capacidades de procesamiento (OTHMAN & AYDIN, 2018).

A pesar de las restricciones que existen para el despliegue de redes neuronales de alto desempeño en teléfonos móviles, la capacidad de procesamiento de los mismos ha evolucionado de forma gradual en los últimos años y hoy en día las tarjetas de procesamiento de video y procesadores en los teléfonos inteligentes se asemejan con los de una computadora personal, lo cual facilita su uso conjunto con este tipo de modelos.

En este contexto la mayoría trabajos que efectúan implementaciones de redes neuronales en teléfonos móviles es reciente, entre la bibliografía más relevante se encuentre el estudio realizado por (Cai et al., 2020) donde se utiliza el modelo YOLO, los autores proponen la detección de objetos en tiempo real en dispositivos móviles a través del diseño de un esquema adaptativo de kernel. El esquema propuesto se basa en la colaboración GPU-CPU junto con optimizaciones avanzadas asistidos por compiladores. Los resultados experimentales indican que el esquema propuesto alcanza una tasa de compresión 14× en YOLOv4 con 49.0 mAP. Se reporta una velocidad de inferencia de 17 FPS utilizando las GPU del teléfono Samsung Galaxy S20 y con el uso del esquema colaborativo GPU-CPU propuesto, la velocidad de inferencia aumenta a 19.1 FPS y supera al YOLOv4 original con una aceleración 5 veces mayor.

Otro trabajo referente en el área es el efectuado por Chan & Kemao (Chan, Lee, & Kemao, 2018) los cuales presentan una técnica para el reconocimiento y seguimiento de objetos sin textura en tiempo real en dispositivos móviles. El algoritmo propuesto es una modificación del detector de objetos BIND (Binary Integrated Net Descriptor), personalizado para aplicaciones en dispositivos móviles. Esta modificación denominada BINDLite emplea técnicas para superar el bajo poder computacional de los dispositivos móviles actuales, conservando la robustez de detección de objetos del modelo BIND original.

La Tabla 1 efectúa una comparativa entre la solución propuesta, las soluciones comerciales, de código libre y otras similares tomado como referencia los criterios de operación y funcionamiento antes descritos:

**Tabla 1.** *Comparativa entre la solución propuesta, soluciones comerciales y de código libre.*

| Autor | Interfaz Simple | Datos abiertos | Plataforma Modular | Clasificación de Vehículos | Exactitud en los conteos | Soporte en teléfonos móviles | Crowd Sourcing | Entorno Ubicuo | Diversidad geográfica de datos | Sin Costo |
|---|---|---|---|---|---|---|---|---|---|---|
| Implementación propia de TFM | ✓ | ✓ | ✓ | ✓ | 93-97% | ✓ | ✓ | ✓ | ✓ | ✓ |
| TRIGG (TRIGG, 2021) | X | X | X | X | NE | X | X | X | X | X |
| All Trafic Solutions (Solutions, 2021) | NE | X | X | ✓ | 97% | X | X | X | X | X |
| Intu Visison (IntuVision, 2021) | ✓ | X | ✓ | ✓ | 99% | X | X | ✓ | X | X |
| Axiomtek (Axiomtek, 2021) | NE | X | ✓ | ✓ | NE | X | X | X | X | X |
| HMI (HMI, 2021) | ✓ | X | X | ✓ | NE | X | X | X | X | X |
| ICOMS (ICOMS, 2021) | NE | X | ✓ | ✓ | 99% | X | X | ✓ | X | X |
| (Peña-González & Nuño-Maganda, 2014) | X | ✓ | X | ✓ | 87% | X | X | X | X | X |
| (Shan & Li, 2018) | X | ✓ | X | ✓ | 97% | X | X | X | X | X |
| (OTHMAN & AYDIN, 2018) | X | X | X | ✓ | NE | X | X | X | X | X |

Fuente: Elaboración propia.

# 3. OBJETIVOS CONCRETOS Y METODOLOGÍA DE TRABAJO

## 3.1. Objetivo general

Implementar un conjunto de herramientas que combinen una iniciativa de Crowdsourcing con las ventajas y capacidad de procesamiento de los dispositivos móviles Android actuales para desplegar una red neural convolucional que permita procesar y generar conteos de vehículos y peatones en la carretera.

## 3.2. Objetivos específicos

1. Desarrollar un aplicativo móvil para dispositivos Android que permita la captura, procesamiento y detección de vehículos en secuencias de video en tiempo real.

2. Implementar un aplicativo Web que permita almacenar y visualizar datos de conteos vehiculares.

3. Generar la información necesaria para dar solución a la problemática de la congestión de tráfico, uso inadecuado de espacios públicos para bicicletas, peatones y la falta de opciones de transporte público urbano en una ciudad intermedia

La Tabla 2 describe los criterios de aceptación que debe cumplir cada uno de los objetivos, los cuales se verifican en el Capítulo V.

**Tabla 2.** *Criterios de aceptación conforme a los objetivos.*

| **Objetivos** | **Criterios** |
|---|---|
| Objetivo General | - Plataforma para la visualización de conteos vehiculares.<br>-Aplicativo Android para la captura, detección y conteo de vehículos y peatones en la carretera. |
| Objetivo Específico 1 | -Captura, procesamiento y generación de conteos vehiculares en tiempo real. |

| | |
|---|---|
| Objetivo Específico 2 | -Plataforma con Frontend y Backend para visualización y almacenamiento de conteos |
| Objetivo Específico 3 | -Repositorio de datos abierto. |

Fuente: Elaboración propia

## 3.3. Metodología del trabajo

En esta sección se describe la metodología seleccionada para la elaboración del trabajo planteado, se tomó como referencia la metodología agile SCRUM. Se consideró SCRUM puesto es una metodología muy versátil, se adapta a los cambios, se adecua al trabajo en grupos reducidos, integra al cliente al equipo de trabajo y posee ciclos de entrega cortos e incrementales, lo cual se adecua a las necesidades del proyecto planteado.

## 3.2. Metodología Scrum

La metodología considerada para el desarrollo de este TFM toma como referencia a SCRUM, puesto SCRUM es una metodología orientada al trabajo en equipos, se ha considerado modificar algunos procesos definidos por este Framework para adaptarlo a una metodología unipersonal. En los referentes a los roles SCRUM implementa tres tipos: el Product owner, Development team y Scrum master, los cuales para este trabajo se han reducido a un único rol denominado “manager-development”, el cual asumirá las responsabilidades de la creación, desarrollo e implementación de requisitos e historias de usuario. Al igual que la metodología SCRUM tradicional se considera la creación y manejo de los incrementos mediante el uso del producto backlog, la creación de los objetivos de Sprint y la generación del sprint backlog.

Para este caso se ha decido suprimir las reuniones de tipo “Dayli Scrum” y considerar únicamente las planificaciones de inicio de Sprint (Sprint planning) y las revisiones al finalizar un Sprint. Otro aspecto a destacar es la duración de lo Sprints, para este caso se consideraron cuatro Sprints, el primero y segundo con una duración de 3 semanas, el tercero dos semas y el cuarto 1 semana. El detalle cada Sprint se verá reflejado en el capítulo 4.

En las siguientes secciones se aborda y describe a detalle las características y fases de la metodología SCRUM aplicada.

SCRUM es un modelo de desarrollo de software que permite configurar un plan paso a paso con éxito, su agilidad es solo una de las muchas características que destaca a esta metodología, brinda la posibilidad de trabajo en equipo o de forma individual como una alternativa para auto organizar el trabajo, un equipo en SCRUM consiste de un conjunto de personas que trabajan juntas para entregar los incrementos de producto solicitado, consta de tres roles :

### 3.2.1. Product owner

El product owner se encarga de potenciar el valor del trabajo del equipo de desarrollo y del producto, de distintas maneras dependiendo de la organización, equipos scrum o individuos. El Product owner administra la cartera de productos que incluye:

- Detallar correctamente los elementos que componen la Pila de Producto;
- Organizar los elementos en el Product Backlog para alcanzar una mejor funcionalidad en los objetivos y misiones
- Fortalecer el valor del trabajo que desempeña el Equipo de Desarrollo
- Garantizar que el Product Backlog sea sencillo, funcional y claro para todos
- Cerciorarse de que el equipo entienda los elementos de la cartera de productos al nivel necesario.

### 3.2.2. Scrum Master.

El Scrum Master es el individuo encargado de cerciorarse que la metodología Scrum sea compresible, aplicable y funcional, verificando que todo el equipo Scrum logre dominar los elementos necesarios de Scrum. Además, el individuo encargado de llevar a cargo este rol deberá asistir a personas ajenas al equipo de Scrum a comprender cuáles de sus interacciones son útiles y cuáles no.

El Scrum Master colabora con el product owner con:

- Contribuir con metodologías funcionales para un manejo eficiente de la cartera de productos.
- Colaborar con el equipo Scrum a entender los elementos que componen la Pila de Productos de forma sencilla y eficaz.
- Conocer la planificación de productos en un medio empírico.
- Entender y practicar la agilidad, ligereza y eficiencia.
- Favorecer y agilitar los eventos de Scrum según las necesidades.

### 3.2.3. Equipo de desarrollo

Este equipo se compone por desarrolladores capacitados en un área propia de especialización. Además, se ocupan de la ejecución real del software, incremento y entregables que se aportaran al finalizar cada Sprint.

El equipo de desarrollo debe estar integrado de personas con habilidades especializadas como desarrolladores front-end, desarrolladores backend, Dev-Ops, analistas de negocios, expertos en control de calidad, encargados de bases de datos, etc., pero todos se denominan desarrolladores; no se permiten otros títulos.

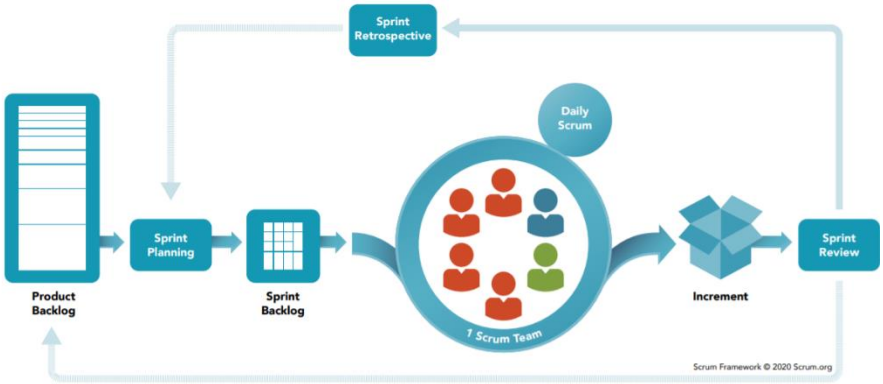
### 3.2.4. Artefactos Scrum

- **Product Backlog:** El Product Backlog es una con la versión inicial que enumera los requisitos preliminares y conocidos, esta lista de requisitos evoluciona según los cambios realizados en entorno de desarrollo y producto.
- **Objetivo de Sprint:** Un objetivo de Sprint muestra el resultado deseado de una iteración que proporciona un objetivo compartido al equipo, cuya finalidad debe definirse antes de que el equipo comience el Sprint para poder concentrarse en conseguirlo**.**
- **Sprint Backlog:** Sprint Backlog está compuesto por elementos del Product Backlog utilizados para el Sprint actual, además de planes para entregar incrementos de producto para lograr los objetivos de Sprint

### 3.2.5. Ciclo de vida de scrum

El ciclo de vida de scrum da inicio con la creación de un backlog de requisitos primarios, sin proveer ninguna orientación sobre cómo desarrollarlo o priorizarlo, el backlog consiste en un conjunto de sprints que generan un resultado final con un incremento de productos potencialmente entregables como se muestra en la Figura 9.

**Figura 9.** *Ciclo de vida de SCRUM*




Fuente: Elaboración propia

A continuación, se resumen los procesos clave en el ciclo de vida de scrum:

- En primer lugar, se define el Product Backlog (los requisitos principales del producto priorizando su orden), esta tarea la lleva cargo el Product Owner.
- El equipo Scrum se encarga de elaborar las estimaciones y cambios para la carga de trabajo de acuerdo a la lista de Backlog del producto en la reunión de refinamiento del Backlog del producto.
- Al contar con la lista de Product Backlog, se planifica una junta para determinar el objetivo de Sprint de esta iteración.
- Después, se escoge una lista de historias de usuario para crear el Sprint Backlog que servirá para el próximo sprint que podría lograr el objetivo.
- El Sprint Backlog se completa por el equipo Scrum, cada integrante mejora las tareas pequeñas de acuerdo a las necesidades del Sprint Backlog.

- El Sprint precisa de una reunión al día de Scrum y estas reuniones podrán tener una duración hasta de 15 minutos. Aquí los miembros del equipo intercambian sus avances con los demás miembros informando las actividades realizadas el día anterior y comprometiéndose con lo que aspiran alcanzar hoy.
- Cuando el Sprint se completa, es decir, el Sprint Backlog se encuentra listo con todas las historias de usuario, se debe organizar una reunión para su revisión. En esta reunión es importante que participe el propietario del producto junto con el cliente. Aquí los integrantes del Scrum Team enseñaran el software de trabajo que han completado.
- Por último, la Retrospectiva del Sprint se la realiza después de la verificación del Sprint al final de cada Sprint. En la retrospectiva, los integrantes identifican los elementos del proceso que funcionaron o no durante el sprint, y analizan las posibles soluciones. Las retrospectivas duran aproximadamente 90 minutos e interviene en la mejora constante de la cultura del equipo y en la cadencia de Sprint.

Una vez definida la metodología de trabajo, se procede a implementarla de acuerdo a los pasos descritos en el capítulo para lo cual en el Capítulo IV se detalla el product Backlog que describe los requerimientos del proyecto y la carga de trabajo asignado a cada sprint, en función de estas tareas se desarrolla el capítulo IV.

Ser resume a continuación las principales fases que forman parte de la implementación propuesta:

1. Diseño de la Arquitectura general del Sistema.
2. Recopilación, selección y análisis de modelos para la detección de objetos en imágenes.
3. Selección del Framework de trabajo para la implementación del modelo seleccionado.
4. Aplicación del modelo de detección de objetos y evaluación de la precisión y velocidad en la detección.
5. Selección del Framework para el desarrollo del aplicativo Web de almacenamiento y visualización de conteos.
6. Evaluación del Sistema y Análisis de Resultados.

### 3.2.6. Diseño de la Arquitectura general del Sistema

En esta fase se define de forma general y especifica la arquitectura del sistema, se describe la integración de todas las partes que involucran la iniciativa desde la captura, procesamiento y envío de datos generados por la aplicación y el modelo de red neuronal seleccionado hacia a plataforma Web de almacenamiento y visualización de datos.

### 3.2.7. Recopilación, selección y análisis de modelos para la detección de objetos en imágenes.

Se efectúa el análisis comparativo de diferentes modelos de redes neuronales, con el objeto de obtener el más adecuado para su implementación en dispositivos móviles, se identifican y describen los requisitos asociados y se consideran dos factores claves para la selección del modelo: la velocidad de detección de referencia y la precisión media (mAP).

### 3.2.8. Selección del Framework de trabajo para la implementación del modelo seleccionado

Considerando la variedad de tecnologías que existen en el mercado para el desollado de aplicaciones móviles, en esta se sección se selecciona un framework que se adapte al manejo de librerías basadas en técnicas de redes neuronales convoluciones y en especial sea capaz de trabajar con el modelo seleccionado en la sección 3.3.2.

### 3.2.9. Aplicación del modelo de detección de objetos y evaluación de la precisión y velocidad en la detección.

En esta etapa se realiza el establecimiento del modelo de detección de objetos, y se analiza el rendimiento del mismo en dispositivos de diferentes gamas, con la finalidad de elaborar una comparativa que describa los tiempos de procesamiento, precisión en la detección y conteos del algoritmo en teléfonos con capacidades de procesamiento diferentes.

### 3.2.10. Selección del Framework para el desarrollo de la plataforma de almacenamiento y visualización de datos.

Se efectúa un pequeño análisis de los requisitos del sistema y en base a ellos se seleccionan las tecnologías asociadas al Backend y Frontend para el almacenamiento y visualización de los conteos enviados desde el aplicativo móvil en tiempo real, en esta sección se describe con más detalle la arquitectura de la plataforma de visualización de datos y sus diferentes componentes.

### 3.3.6 Evaluación del Sistema y Análisis de Resultados.

Por último, en la etapa final se obtiene el rendimiento del sistema en general con cualquier imagen de entrada y se puede medir el rendimiento de los sistemas de detección de objetos y presentación de datos trabajando de manera conjunta. En esta etapa se podrá evaluar si se ha alcanzado el objetivo de implementar un sistema de colaboración de datos libre para la mejora de la movilidad en espacios públicos.

# 4. DESARROLLO ESPECÍFICO DE LA CONTRIBUCIÓN

En este capítulo se desarrolla la metodología de manera técnica considerando el marco de trabajo SCRUM el cual fue descrito en el capítulo III, se detallando la arquitectura general y especifica en cada una de las etapas del sistema, se describen los principales casos de uso y las tecnologías involucradas en el desarrollo de la solución, así como la arquitectura del modelo de red neuronal utilizado para el proceso de detección de objetos.

## 4.1. Definición de requisitos e historias de usuario.

Tomado como referencia la metodología SCRUM, el primer paso en sus etapas iniciales es la definición y creación de las historias de usuario que debe satisfacer el sistema, para lo cual se procedió con la creación del "product backlog", descrito en la Tabla 3:

**Tabla 3**. *Product Backlog de las historias de usuario del Sistema.*

| | Enunciado de la historia | | | | Criterios de aceptación | |
|---|---|---|---|---|---|---|
| ID de la historia | Rol | Característica-Funcionalidad | Razón/Resultado | # de escenario | Criterio de aceptación | Evento |
| 1 | Como un usuario | Se requiere la implementacion de una aplicacion movil para dispositivos Android que permita detectar objetos mediante la camara del dispositivo | Requerido para la deteccion de vehiculos livianos, pesados, personas, motocicletas, bicicletas y autobuses | 1 | La cámara del móvil se activara después de abrir la aplicación | Al iniciar la aplicacion |
| | | | | 2 | En la pantallla del movil debe existir un espacio para la vizualizacion de los objetos detectados por la camara y un boton para enviar y efectuar el proceso de conteo de vehiculos | Al abrir la aplicacion en el dispositivo |
| 2 | Como usuario | Se requiere que el algoritmo de detección de objetos utilice una red neuronal pre-entrenada. | Para evitar el proceso de etiquetado y entrenamiento requerido para la generación de una red neuronal | 1 | Uso de la red neuronal SSD Mobilenet | Al efectuar la deteccion de vehiculos |
| 3 | Como usuario | Se requiere que una vez efectuado el proceso de detección de vehículos se proceda a contar y clasificarlos por categorías (vehículos, motocicletas, peatones, autobuses y bicicletas). | Para generar un conjunto de datos de movilidad mediante conteos en tiempo real. | 1 | El aplicativo móvil detecta y clasifica los vehículos (por categorias) capturados por la cámara del teléfono móvil. | Una vez los objetos son detectados por la camara del dispositivo, inicia el proceso de deteccion. |
| 4 | Como usuario | Se requiere que adicional al resultado de los conteos la aplicación reporte las coordenadas geográficas desde las cuales los datos fueron generados. | Para geo-referenciar y asociar los conteos vehiculares con calles y lugares. | 1 | El aplicativo pide acceso a la ubicación del usuario para reportar el resultado de los conteos | Al iniciar la aplicación |

| | | | | | | |
|---|---|---|---|---|---|---|
| **5** | **Como usuario** | Se necesita enviar el resultado de los conteos vehiculares a una API externa mediante un botón implementado en la aplicación. | Para facilitar el envio de los datos de conteos a una API externa. | **1** | El aplicativo dispone de un botón para enviar el resultado de los conteos a una base de datos externa | Al presionar el boton Enviar Conteos |
| | | | | | | |
| **6** | **Como usuario** | Se necesita implementar una API externa (con base de datos propia) para almacenar los conteos enviados por los usuarios mediante la aplicación. | Para almacenar el resultado de los conteos. | **1** | Conexion correcta entre la API y una base de datos externa. | Activa cuando se soliciten o ingresen datos |
| | | | | | | |
| **7** | **Como usuario** | Se necesita implementar los métodos GET y POST en la API para el almacenamiento y consulta de la información de conteos. | Para almacenamiento y consulta de datos | **1** | El almacenamiento y consulta de datos funciona de forma correcta | Disponibles para responder a las peticiones |
| | | | | | | |
| 8 | **Como usuario** | Se necesita disponer de un aplicativo Web amigable que permita visualizar el resultado de los conteos. | Para facilitar el acceso de investigadores e interesados en movilidad a los datos. | **1** | Página Web sencilla para visualizar el resultado de los conteos mediante el uso de mapas. | Para visualizar y descargar los conteos |
| | | | | | | |
| **9** | **Como usuario** | Se necesita geo-referenciar en el mapa la ubicación desde la cual los datos fueron reportados. | Para asociar la ubicación de los conteos a un mapa en el aplicativo Web. | **1** | Etiquetas en las ubicaciones del mapa, al dar click en las mismas se desplegará un menú con el resultado de los conteos | Al seleccionar un item con los conteos. |
| | | | | | | |
| **10** | **Como usuario** | Se necesita descargar un cvs con el resultado de todos los conteos y su ubicación geográfica. | Para facilitar el estudio y análisis de so datos generados | **1** | Archivo csv con el resultado de todos los conteos de la plataforma | Al seleccionar la opcion Descargar datos en le aplicativo Web |

Fuente: Elaboración propia

La Tabla 3 describe de forma general los requisitos mínimos funcionales que el sistema debe cumplir para satisfacer los objetivos planteados en el presente TFM. El product backlog es de utilidad al momento de planear cada sprint en las fases de desarrollo, para el presente TFM se ha considerado el desarrollo del proyecto en cuatro sprints, dentro de los cuales se asignaron las diferentes historias de usuario. La asignación de tareas se describe de forma detallada en la Tabla 4.

**Tabla 4.** *Distribución de las historias de usuario en los sprints.*

| SPRINT | Historia de Usuario | Prioridad |
|---|---|---|
| **SPRINT 1** | Diseño de la topología general del Sistema | 1 |
| | Análisis de las tecnologías y Frameworks a utilizar. | 1 |

| | | |
|---|---|---|
| | Creación de la aplicación móvil para dispositivos Android que permita detectar objetos mediante la cámara del dispositivo. | 2 |
| | Se requiere que el algoritmo de detección de objetos utilice una red neuronal pre-entrenada. | 2 |
| | Se requiere que una vez efectuado el proceso de detección de vehículos se proceda a contar y clasificarlos por categorías (vehículos, motocicletas, peatones, autobuses y bicicletas). | 2 |
| **SPRINT 2** | Se requiere que adicional al resultado de los conteos la aplicación reporte las coordenadas geográficas desde las cuales los datos fueron generados. | 1 |
| | Se necesita enviar el resultado de los conteos vehiculares a una API externa mediante un botón implementado en la aplicación. | 1 |
| | Se necesita implementar una API externa (con base de datos propia) para almacenar los conteos enviados por los usuarios mediante la aplicación. | 1 |
| **SPRINT 3** | Se necesita implementar los métodos GET y POST en la API para el almacenamiento y consulta de la información de conteos. | 1 |
| | Se necesita disponer de un aplicativo Web amigable que permita visualizar el resultado de los conteos. | 1 |
| | Se necesita geo-referenciar en el mapa la ubicación desde la cual los datos fueron reportados. | 1 |
| | Se necesita descargar un cvs con el resultado de todos los conteos y su ubicación geográfica. | 1 |
| **SPRINT 4** | Análisis y validación de resultados | 1 |
| | Análisis de precisión en el proceso de detección de vehículos | 1 |
| | Validación del sistema y requisitos | 1 |
| | Validación externa de usuarios | 1 |

Fuente: Elaboración propia

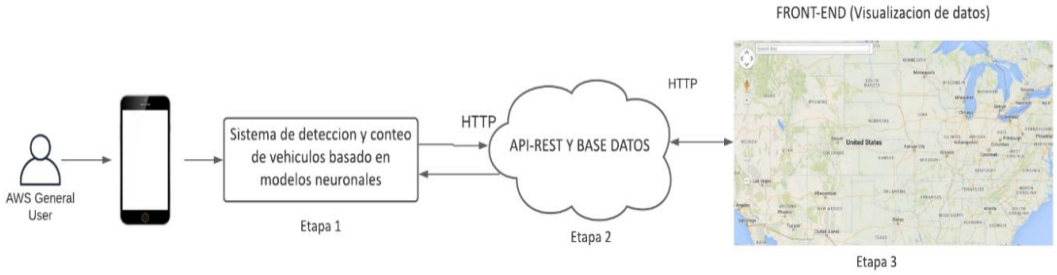
## 4.2. Desarrollo de la metodología

En esta sección se consideran las historias de usuario definidas en el Product Backlog y asignadas en los sprints de la Tabla 4. Se detalla y describe la metodología con la cual se desarrollaron las historias para la generación de la aplicación móvil y web en sus diferentes etapas.

### 4.2.1. Topología general del sistema (Sprint 1)

La arquitectura general propuesta para el sistema de conteo vehicular se describe en la Figura 10:

**Figura 10.** *Topología general del Sistema.*




Fuente: Elaboración propia

El sistema consta de tres etapas: en la primera fase el usuario mediante un teléfono móvil efectúa la captura de imágenes, la cuales son pre-procesadas por el sub-sistema de detección y conteo. Es en esta parte del sistema donde se implementa el modelo de red neuronal encargado de llevar a cabo la detección y conteo de vehículos, posteriormente los conteos se envían al servicio Web para su almacenamiento y consulta.

En la fase 2 del sistema se implementa la lógica de backend, en esta fase se optó por implementar una lógica de micro servicios basada en serverless. Se consideró este tipo de tecnología puesto que hoy en día la informática sin servidor ofrece una serie de ventajas sobre la infraestructura tradicional basada en la nube o centrada en el servidor, entre las que

destacan una mayor escalabilidad, más flexibilidad y un tiempo de lanzamiento más rápido, todo a un costo reducido.

Una vez definida la arquitectura general del sistema se procede a efectuar un análisis general de las tecnologías a utilizar para la implementación de las diferentes historias de usuario. Para el desarrollo de la aplicación móvil se consideró el uso de Android Studio en conjunto con la librería Tensorflow Lite, la cual posee soporte para la implementación de redes neuronales convoluciones en dispositivos móviles, para manejo arlo logia de backend en la API se consideró utilizar las funciones lambda de AWS en conjunto con la base de datos DynamoDB y finalmente para la creación del Frontend de la aplicación web se emplea React JS .

### 4.2.2. Descripción de tecnologías usadas. (Sprint 1)

#### 4.2.2.1. Serverless

La tecnología serverless como AWS Lambda permiten implementar arquitecturas backend sin servidor, los desarrolladores no necesitan preocuparse por la compra, el aprovisionamiento y la administración de servidores, es una arquitectura en la que un proveedor proporciona servicios de backend a medida que se necesitan. Entre las principales ventajas que ofrece el usar este tipo de servicio destacan:

- No es necesaria la gestión del servidor

Aunque la informática “sin servidor” en realidad se lleva a cabo en servidores, los desarrolladores nunca tienen que lidiar con los servidores. Son administrados por el proveedor, lo que reduce los gastos y también libera a los desarrolladores para crear y expandir sus aplicaciones sin verse limitados por la capacidad del servidor.

- Solo se cobra por el espacio del servidor utilizado, lo que reduce el costo.

El código solo se ejecuta cuando la aplicación sin servidor necesita funciones de backend, y el código se escala automáticamente según sea necesario. El aprovisionamiento es dinámico, preciso y en tiempo real. Algunos servicios son tan exactos que dividen sus cargos en incrementos de 100 milisegundos. En contraste, en una arquitectura tradicional de 'servidor

completo', los desarrolladores tienen que proyectar de antemano cuánta capacidad de servidor necesitarán y luego comprar esa capacidad, ya sea que terminen usándola o no.

- Las arquitecturas sin servidor son inherentemente escalables

Las aplicaciones creadas con una infraestructura sin servidor se escalarán automáticamente a medida que la base de usuarios crezca o aumente el uso. Si una función debe ejecutarse en varias instancias, los servidores del proveedor las iniciarán, ejecutarán y finalizarán según sea necesario, a menudo utilizando contenedores.

- Es posible realizar implementaciones y actualizaciones rápidas

Al utilizar una infraestructura sin servidor, no es necesario cargar código en los servidores ni realizar ninguna configuración de backend para lanzar una versión funcional de una aplicación. Los desarrolladores pueden cargar rápidamente fragmentos de código y lanzar un nuevo producto.

- El código puede ejecutarse más cerca del usuario final, lo que reduce la latencia.

Debido a que la aplicación no está alojada en un servidor de origen, el código se puede ejecutar desde cualquier lugar. Por lo tanto, es posible, dependiendo del proveedor utilizado, ejecutar funciones de la aplicación en servidores cercanos al usuario final. Esto reduce la latencia porque las solicitudes del usuario ya no tienen que viajar hasta un servidor de origen.

#### 4.2.2.2. Tensorflow Lite

El Framework de desarrollo Tensorflow Lite proporciona una colección de herramientas que permiten implementar un modelo de aprendizaje automático para una variedad de propósitos y entornos. TensorFlow Lite permite correr modelos del framework TensorFlow en dispositivos embebidos o teléfonos móviles, llevando a cabo el proceso de inferencia de aprendizaje automático de forma rápida y con latencia baja en mediante la transformación del modelo de red a un objeto binario pequeño.

Los componentes que forman parte de TensorFlow Lite se describen a continuación:

- **Iintérprete de TensorFlow Lite:** es el módulo encargado de ejecutar los modelos optimizados en diferentes tipos de hardware, para el caso de estudio en teléfonos celulares.
- **Conversor de TensorFlow Lite:** modulo encargado de convertir los modelos de TensorFlow en un formato adecuado para que sean utilizados por el intérprete.

Para el caso de estudio del presente TFM se siguió el flujo de trabajo recomendado por TensorFlow Lite, lo cual implica llevar a cabo los siguientes pasos:

- **Selección del modelo de red neuronal:** Se debe seleccionar un modelo de red neuronal soportado por TensorFlow, para nuestro caso de estudio se seleccionó SSD Mobilenet.
- **Conversión del modelo**: Implica utilizar el conversor de TensorFlow Lite para convertir el modelo a un formato soportado.
- **Implementación y despliegue en el dispositivo**: En esta etapa se ejecuta el modelo en el dispositivo móvil mediante el intérprete de TensorFlow Lite.
- **Optimización del modelo**: Consiste en reducir el tamaño del modelo neuronal y aumentar la eficiencia con un impacto mínimo en la exactitud de los conteos.

#### 4.2.2.3. DynamoDB

DynamoDB es una base de datos NoSQL basada en el paradigma de clave-valor, se utiliza en una gran variedad de aplicaciones semiestructuradas basadas en datos que prevalecen en casos de uso modernos y emergentes más allá de las bases de datos tradicionales, desde Internet de las cosas (IoT) hasta aplicaciones sociales o juegos multijugador-masivos.

Cuando se configura DynamoDB en AWS no se aprovisiona servidores específicos ni asigna cantidades establecidas de disco. En su lugar AWS aprovisiona el rendimiento: define la base de datos en función de la capacidad aprovisionada, es decir se determina cuántas transacciones y cuántos kilobytes de tráfico desea admitir por segundo. Los usuarios

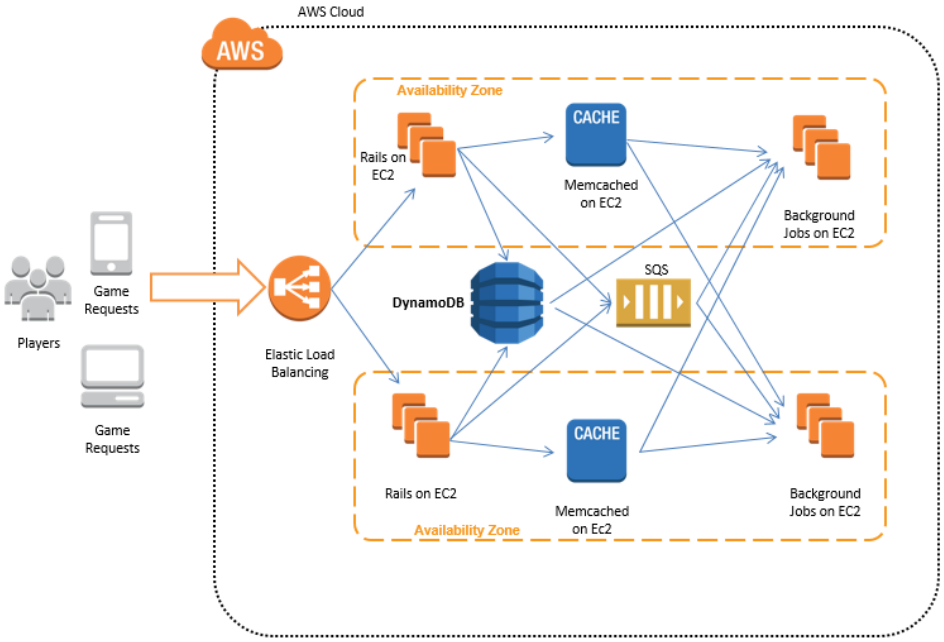
especifican un nivel de servicio de unidades de capacidad de lectura (RCU) y unidades de capacidad de escritura (WCU).

En el diagrama de la Figura 11 se presenta un esquema de conexión típico de DynamoDB en AWS.

#### 4.2.2.4. React JS

React JS es una biblioteca de JavaScript utilizada en la creación de interfaces de usuario o UI. En términos de sitios web y aplicaciones web, las IU son colecciones de menús en pantalla, barras de búsqueda, botones y cualquier otra interfaz con la que un usuario interactúa para utilizar un sitio web o una aplicación. React promueve el uso y reutilización de código mediante el uso de componentes, ahorra tiempo de desarrollo y reduce los errores de codificación.

**Figura 11.** *Esquema de conexión típico de DynamoDB en AWS.*

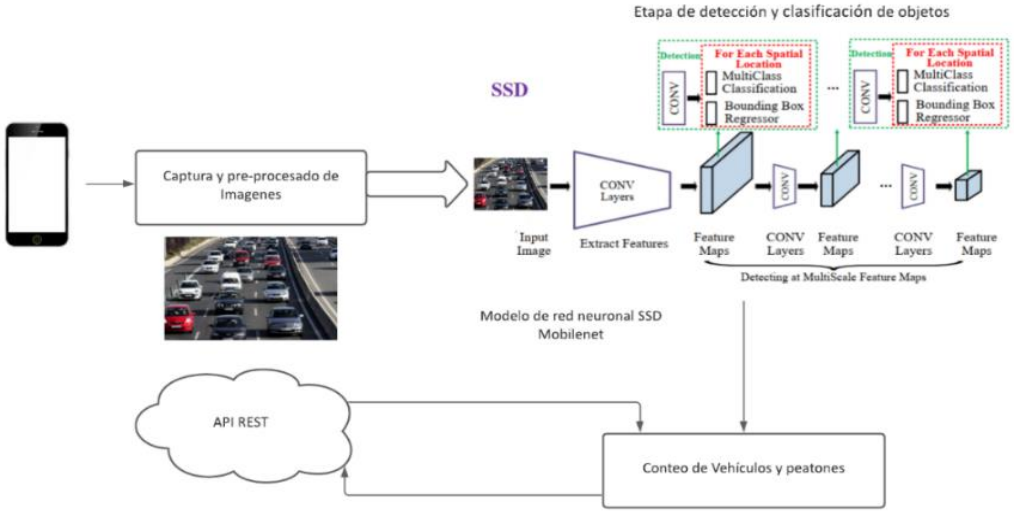



Fuente: (AWS, 2019)

### 4.2.3. Topología y descripción de la etapa de detección y conteo de objetos (Sprint 1)

Esta sección describe la topología del sub-sistema de detección, clasificación y conteo de vehículos (ver Figura 12).

**Figura 12.** *Topología del sub-sistema de captura, detección y conteo de vehículos en Smartphones.*




Fuente: Elaboración propia

Esta etapa consta de 4 fases, en una primera instancia el usuario efectúa la captura de imágenes a las cuales se aplica un pre-procesamiento que consiste en la redimensión y conversión de datos de la imagen a un formato adecuado al modelo neuronal utilizado, en nuestro caso como se describió en el capítulo II se optó por utilizar el modelo de red neuronal denominado SSD Mobilenet.

Para efectuar el pre-procesamiento de imágenes se emplearon métodos provistos por Tensorflow Lite, en la Figura 13 se presenta un extracto de código implementado en Android Studio para efectuar el pre-procesameinto de las imágenes de acuerdo a la topología de la Figura 12.

**Figura 13.** *Pre-procesamiento de imágenes (Tesorflow Lite).*

```
public void processImage() {
  ++timestamp;
  final long currTimestamp = timestamp;
  trackingOverlay.postInvalidate();

  // No mutex needed as this method is not reentrant.
  if (computingDetection) {
    readyForNextImage();
  }
  computingDetection = true;
  LOGGER.i("Preparing image " + currTimestamp + " for detection in bg thread.");

  rgbFrameBitmap.setPixels(getRgbBytes(), offset: 0, previewWidth, x: 0, y: 0, previewWidth, previewHeight);

  readyForNextImage();
  final Canvas canvas = new Canvas(croppedBitmap);
  canvas.drawBitmap(rgbFrameBitmap, frameToCropTransform, paint: null);
  // For examining the actual TF input.
  if (SAVE_PREVIEW_BITMAP) {
    ImageUtils.saveBitmap(croppedBitmap);
  }
```

Fuente: Elaboración propia

**Figura 14.** *Configuración del módulo TFLiteObjectDetectionAPIModel.*

```
public void onPreviewSizeChosen(final Size size, final int rotation) {
  final float textSizePx =
      TypedValue.applyDimension(
          TypedValue.COMPLEX_UNIT_DIP, TEXT_SIZE_DIP, getResources().getDisplayMetrics());
  borderedText = new BorderedText(textSizePx);
  borderedText.setTypeface(Typeface.MONOSPACE);

  tracker = new MultiBoxTracker( context: this);

  int cropSize = TF_OD_API_INPUT_SIZE;

  try {
    detector =
        TFLiteObjectDetectionAPIModel.create(
            context: this,
            TF_OD_API_MODEL_FILE,
            TF_OD_API_LABELS_FILE,
            TF_OD_API_INPUT_SIZE,
            TF_OD_API_IS_QUANTIZED);
    cropSize = TF_OD_API_INPUT_SIZE;
  } catch (final IOException e) {
    e.printStackTrace();
    LOGGER.e(e, format: "Exception initializing Detector!");
    Toast toast =
        Toast.makeText(
            getApplicationContext(), text: "Detector could not be initialized", Toast.LENGTH_SHORT);
    toast.show();
    finish();
  }

  previewWidth = size.getWidth();
  previewHeight = size.getHeight();
```

Fuente: Elaboración propia

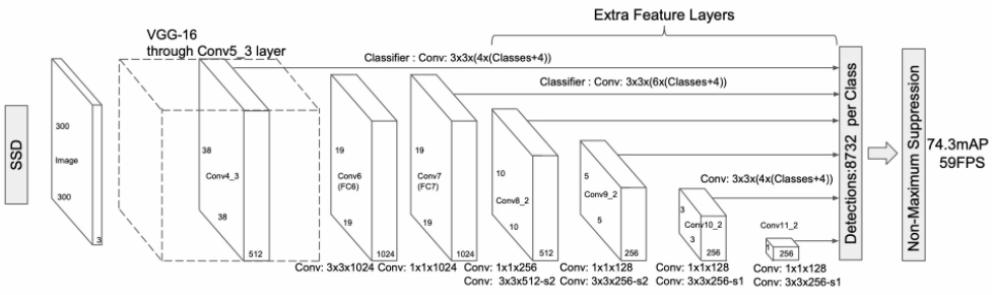
En tanto la Figura 14 presenta el uso del módulo TFLiteObjectDetectionAPIModel utilizado en la configuración básica del detector de objetos, el cual recibe como parámetros de entrada el modelo neuronal, las etiquetas con los objetos a detectar y un conjunto de parámetros extra como la configuración del tamaño de imagen a procesar y la cuantización de la imagen.

Una vez los datos ingresan al modelo neuronal, el mismo se encarga de efectuar la detección y clasificación de los objetos de interés devolviendo los datos de detección en un array con etiquetas las cuales identifican mediante un número el objeto de interés. Esta información es procesada y se efectúa el conteo del número de incidencias por frame detectado en la escena, lo cual representa el número de vehículos, personas, camiones o motos/bicicletas en el instante de tiempo de la captura de la imagen, dichos conteos se envían al servicio Web para su almacenamiento y posterior consulta. En la siguiente sección se describe de forma breve el modo de operación del modelo neuronal SSD Mobilenet descrito en la topología de la Figura 12.

#### 4.2.3.1. SSD Mobilenet

MobileNet es una arquitectura de red neuronal profunda y liviana diseñada para móviles y aplicaciones de visión integradas.

**Figura 15.** *Arquitectura del modelo neuronal SSD.*



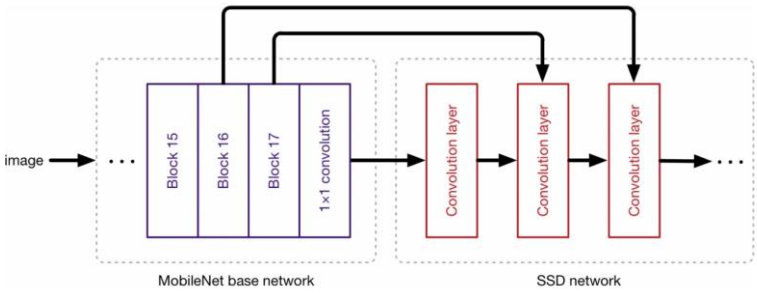

Fuente: (Sanjay Kumar, Subramani, Thangavel, & Parameswaran, 2021).

Las capas centrales de MobileNet se basan en filtros separables en profundidad. Como se describe en la Figura 15, la primera capa efectúa una convolución completa y la detección de objetos SSD realiza una transición única para detectar varios objetos dentro de la imagen. El enfoque SSD están basadas en una red convolucional de retroalimentación que provoca una

recopilación de tamaño fijo de cuadros delimitadores y puntuaciones para la presencia de instancias de clases de objetos en esos cuadros. Está compuesto por dos partes: Mapas de extracción de características y filtro convolucional para la detección de objetos.

SSD está diseñado para ser independiente de la red base, por lo que puede ejecutarse sobre cualquier red neuronal como VGG, YOLO, MobileNet. Con la finalidad de ejecutar redes neuronales de alto consumo de recursos y energía en dispositivos de gama baja en tiempo real MobileNet se integró en el marco SSD, estableciendo a MobileNet como red base para SSD, dando origen a MobileNet SSD. La topología del modelo se presenta en la Figura 16.

**Figura 16.** *Topología de Mobilenet SSD.*



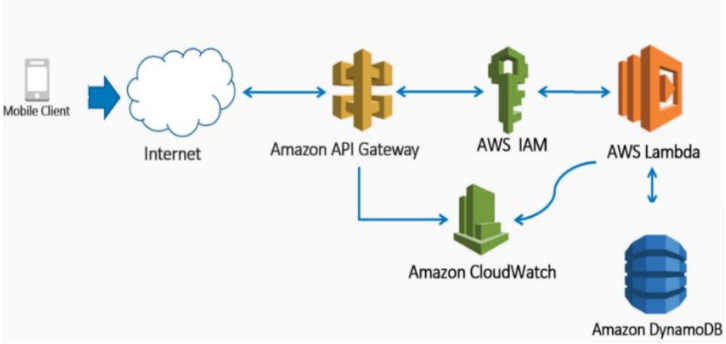

Fuente: (Sanjay Kumar et al., 2021)

### 4.2.4. Topología y descripción del sub-sistema API-RESTful para el manejo de peticiones (Sprint 2).

Para el desarrollo del backend se optó por utilizar los servicios de Amazon Web services, AWS Serverless Computing Architecture es una forma de crear y ejecutar servicios y aplicaciones sin tener que administrar la infraestructura. La aplicación de los usuarios aún se ejecuta en servidores, pero los servidores son administrados por AWS. Es un método que proporciona soporte para servicios de back-end. A los usuarios se les cobra en función de un cálculo establecido, y no se cobra nada más solo por los servicios utilizados. Aunque se denominan servidores sin “servidor”, se siguen utilizando servidores físicos, pero los desarrolladores no tienen que conocerlos. En la Figura 17 se describe la topología del sistema implementado para

la creación de la API encargada del manejo de la lógica de peticiones y almacenamiento de datos.

**Figura 17.** *Topología y configuración de la API REST para el manejo de peticiones.*




Fuente: Elaboración propia

Los servicios AWS Serverless Computing utilizados en la topología de la Figura 14 se describen a continuación:

Servicios de computación sin servidor (AWS Lambda): AWS Serverless proporciona AWS Lambda que permite al usuario ejecutar código sin administrar servidores, y el usuario solo paga por los cálculos utilizados.

Servicios de almacenamiento sin servidor: Se dispone deAmazon S3 proporciona a un equipo de desarrollo un acopio de objetos durable, certero y en gran medida escalable. Es sencillo de usar y portátil. También está la opción de Amazon EFS, la cual proporciona almacenamiento de archivos elástico, escalable y puro

Servicios de almacenamiento de datos sin servidor: AWS proporciona Amazon DynamoDB, que es un servicio de base de datos No SQL agil y veloz para todas las aplicaciones; necesita una latencia de milisegundos en n todas las escalas.

Amazon Aurora Serverless: es una configuración de escalado automatizada bajo demanda (sistema compatible con MySQL). Esta base de datos se inicia y apaga automáticamente según su uso.

Servicios de proxy de API: Se dispone de Amazon API Gateway, brinda una asistencia totalmente administrada que posibilita a los desarrolladores la tarea de crear, mantener, publicar, monitorear y proteger las API a cualquier escala. Las puertas de enlace le permiten procesar miles de llamadas API simultáneas y ayudan al usuario a manejar la gestión del tráfico, la autorización, el control de acceso, la supervisión y la gestión de la versión API.

Para efectuar el despliegue de los end points y base de datos DynamoDB en la nube de AWS es necesaria la creación de un archivo de configuración que describe los parámetros de la instancia. La configuración utilizada en su forma más básica se puede apreciar en la Figura 18, en la cual se describe que la API servirá dos puntos de acceso: un POST para el envio de datos y un GET para su consulta.

**Figura 18.** *Configuración básica de la API y base de datos Dynamo DB.*

```
service:
  name: apitesis
provider:
  name: aws
  runtime: nodejs12.x
  region: us-west-2
  iamRoleStatements:
    - Effect: "Allow"
      Action:
        - "dynamodb:*"
      Resource:
        - "*"
functions:
  saveToDoItem:
    handler: handler.saveConteos
    events:
      - http:
          method: post
          path: save-to-do-item
          cors: true
  getToDoItem:
    handler: handler.getconteos
    events:
      - http:
          method: get
          path: to-do-items/{id}
          cors: true
```

```
resources:
  Resources:
  ToDoListTable:
    Type: AWS::DynamoDB::Table
    Properties:
      TableName: to-do-list
      AttributeDefinitions:
        - AttributeName: id
          AttributeType: S
```

Fuente: Elaboración propia

Cada punto de acceso dispone de su propio código de ejecución en la instancia creada en la nube para el caso del método de envío de conteos (POST), la Figura 19 describe la función básica de captura y procesamiento de datos en el end-point.

**Figura 19.** *Función para el envío de datos a la API (método POST).*

```
//const uuid = require('uuid');
const AWS = require('aws-sdk');
const { v4: uuidv4 } = require('uuid');

AWS.config.setPromisesDependency(require('bluebird'));

const dynamoDb = new AWS.DynamoDB.DocumentClient();

module.exports.submit = (event, context, callback) => {
  const httpMethod = event.httpMethod;
  if (httpMethod) {
    switch (httpMethod) {
      case "POST": {
        const requestBody = event.body;
        let preProc = requestBody.toString().replace('=', '').replace('&', '');
        let conteosCoordenadas = preProc.replace(/%2C/g, ',').split(',');
        submitCandidateP(candidateInfo(conteosCoordenadas))
          .then(res => {
            callback(null, {
              statusCode: 200,
              body: JSON.stringify({
                message: `Conteo almacenado con exito`,
                candidateId: res
              })
            });
          })
          .catch(err => {
            console.log(err);
            callback(null, {
              statusCode: 500,
              body: JSON.stringify({
                message: `Unable to save vehicular count`
              })
            })
          });
      }
    }
  }
}
```

Fuente: Elaboración propia

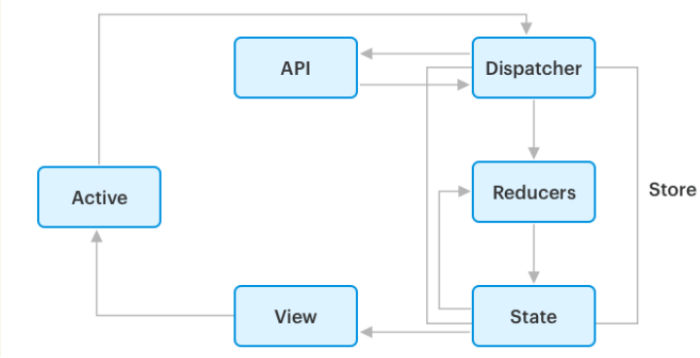
### 4.2.5. Topología y descripción de la aplicación Web para la visualización y descarga de resultados (Sprint 3).

Para la consulta y visualización de los datos de conteos se hizo uso del Framwork React JS en conjunto con la librería “google-maps-loader” (NPM, 2021), la cual permite renderizar componentes de React en el mapa provisto por Google, la librería es completamente isomórfica y puede renderizarse del lado del cliente o servidor. Además, puede representar componentes de mapas en el navegador incluso si la API de Google Maps no está cargada.

En tanto la librería Redux es una herramienta de gestión de estado ligera para aplicaciones JavaScript que permite administrar el estado de aplicaciones escritas en Reacts Js u otros lenguajes con soporte, facilitando la escritura de aplicaciones fáciles de probar y que se pueden ejecutar en diferentes entornos (cliente, servidor, nativo). Una de las formas clave en

que Redux hace esto es haciendo uso de redux-store, de modo que toda la aplicación es manejada por un objeto de estado. La Figura 20 describe la arquitectura con la cual se diseñó la aplicación Web de consulta de datos, se puede observar que se utiliza los módulos distpacher y reducer, los cuales sirve para disparar acciones que permiten alterar el estado de la aplicación ya sea para traer o enviar datos de la API, el reducer se utiliza para guardar los datos al state o estado general de la aplicación, en nuestro caso el state almacena toda la información de los conteos vehiculares provenientes de la API.

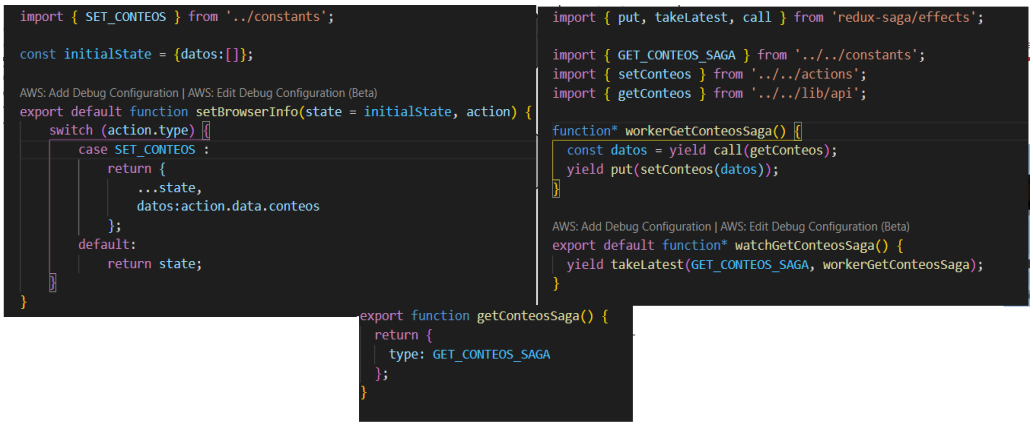
**Figura 20.** *Topología del manejo de estado de la aplicación Web.*




Fuente: Elaboración propia

En la Figura 21 se presentan una captura de pantalla de la implementación en código de la topología descrita en la Figura 20, la cual describe la forma en la cual se interconecta el reducer, state y las acciones de consulta de datos a la API.

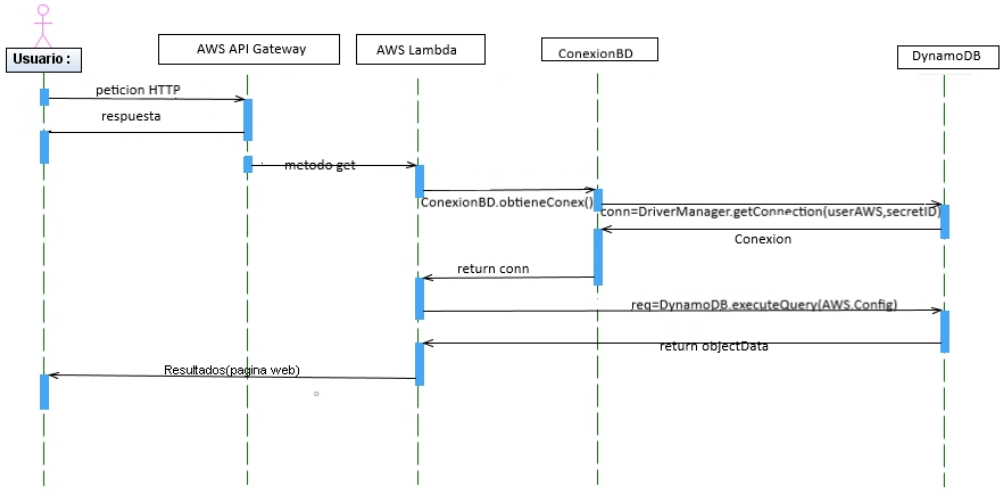
**Figura 21.** *Reducer, action y conexión a la API usando Redux.*




Fuente: Elaboración propia

El flujo de consulta de datos y visualización de conteos se describe en la Figura 22, en donde se aprecia que una petición debe ser procesada por la API de AWS y enviada a la función Lambda encargada de manejar las peticiones GET, en esta función se efectúa el proceso de conexión y consulta de datos con la base de datos DynamoDB.

**Figura 22.** *Diagrama de secuencia para la visualización de datos.*




Fuente: Elaboración propia

#### 4.2.5.1. Diseño e interfaz grafica

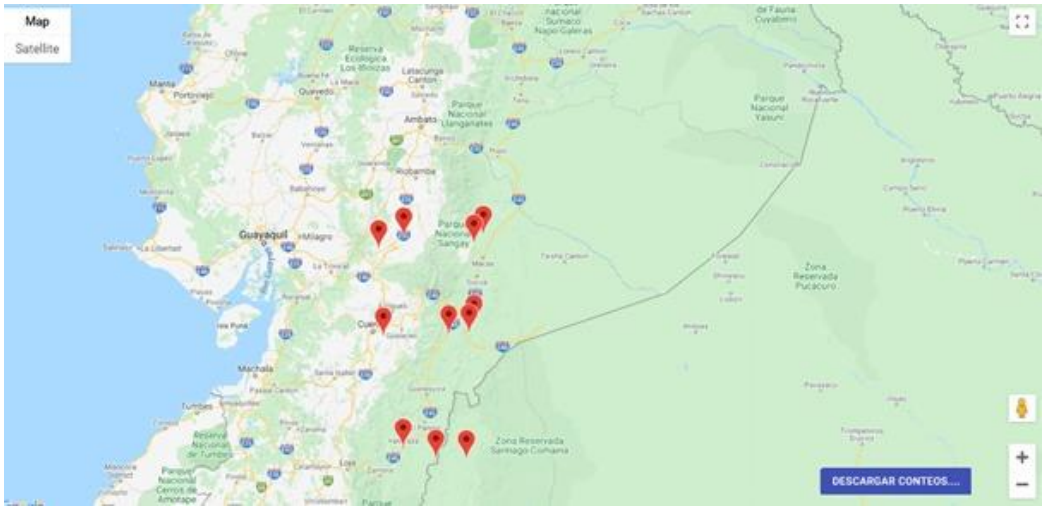
La interfaz de usuario (UI) representa la parte grafica que está diseñada en un dispositivo de información con el que un ser humano puede interactuar, incluida la pantalla, el teclado, el mouse, el lápiz óptico o la apariencia de la pantalla en un sitio web y se utiliza para la interacción con la página. Considerando la definición precedente se ha creado un diseño sencillo y funcional que permita a los usuarios efectuar la consulta, visualización y descarga de contenido (conteos vehiculares) en una única página principal la cual consta únicamente de tres compones principales:

1. **Mapa de Visualización**: Área en la cual se puede visualizar el resultado de los conteos.
2. **Marcador de conteos:** Componente ubicado sobre el mapa y cuya funcionalidad es desplegar un cuadro de texto que indica el número de conteos vehiculares en el sitio.

3. **Botón de descarga:** Proporciona la funcionalidad de descargar el resultado de todos los conteos en formato csv.

El diseño propuesto se presenta en la Figura 23:

**Figura 23.** *Diseño de la interfaz de la aplicación web.*




Fuente: Elaboración propia

Finalmente, el proceso de validación de requisitos y presentación de resultados se efectúa en el capítulo V, este capítulo se corresponde con las actividades definidas en el Sprint 4 de la Tabla 4.

# 5. ANÁLISIS DE RESULTADOS Y VALIDACIÓN DE REQUISITOS

Este capítulo define un conjunto de casos de prueba que permiten validar el correcto funcionamiento del sistema y verificar si se satisface cada uno de las requisitos e historias de usuario planteadas en capítulos precedentes, se presentan los resultados finales y algunas métricas de funcionamiento del sistema en general, también se considera un apartado para validar el uso del sistema y cada uno de sus componentes con usuarios reales mediante un conjunto de pruebas que permiten medir el nivel de satisfacción del sistema en general.

## 5.1. Validación de requisitos e historia de usuario

Para validar los requisitos se definió una plantilla de casos de prueba en la cual se detalla el tipo de prueba y el resultado generado por la misma, cada uno de los casos de prueba tiene relación directa con las historias de usuario definidas en el capítulo IV.

**Tabla 5**. *Caso prueba 1.*

| **Funcionamiento básico de la aplicación móvil** | **1** | |
|---|---|---|
| | **¿Prueba de despliegue?** | **Si** |
| **Descripción:**<br>Esta prueba tiene la finalidad de comprobar que el aplicativo móvil se instala y funciona de forma correcta en un dispositivo Android, sirve para verificar que la cámara y la geolocalización son activadas al abrir el aplicativo. | | |
| **Prerrequisitos**<br>Teléfono móvil con Sistema operático Android, cámara y geolocalización disponibles. | | |
| **Pasos:**<br>1. Instalar el aplicativo móvil<br>2. Abrir la aplicación móvil.<br>3. Verificar que la aplicación móvil solicita permiso para accederá la cámara y geolocalización del dispositivo | | |

| |
|---|
| 4. Comprobar que la cámara y geolocalización han sido actuadas en el teléfono.<br>5. Cerrar la aplicación |
| **Resultado esperado:**<br>Al ejecutar los pasos descritos el dispositivo debe solicitar los permisos al usuario para acceder a la cámara y la geolocalización, posterior a este paso la cámara del teléfono debe estar activa lo cual debe ser verificado en la pantalla del teléfono. |
| **Resultado obtenido:**<br>Se cumple con el resultado esperado en todas sus fases |

Fuente: Elaboración propia

**Tabla 6.** *Caso prueba 2.*

| **Detección de objetos en tiempo Real** | **2** | |
|---|---|---|
| | **¿Prueba de despliegue?** | **Si** |
| **Descripción:**<br>Esta prueba tiene como finalidad verificar la detección de vehículos y peatones en la carretera en tiempo real. | | |
| **Prerrequisitos**<br>Teléfono móvil con Sistema operático Android, con la aplicación de prueba instalada, cámara y geolocalización disponibles. | | |
| **Pasos:**<br>1. Abrir la aplicación móvil.<br>2. Aceptar los permisos para acceso a la cámara y geolocalización.<br>3. Enfocar la cámara del teléfono a un espacio público con flujo vehicular | | |

| |
|---|
| **Resultado esperado:**<br>1. Al enfocar la cámara del dispositivo móvil a un espacio público con flujo de vehículos o peatones automatícenle se dibujará en la pantalla del teléfono rectángulos enmarcando los vehículos detectados en el entorno, adicionalmente se debe presentar una etiqueta junto al rectángulo la cual indica el porcentaje de precisión en la detección. |
| **Resultado obtenido:**<br>Se cumple con el resultado esperado en todas sus fases |

Fuente: Elaboración propia

**Tabla 7.** *Caso prueba 3.*

| **Detección de objetos en tiempo Real** | **2** | |
|---|---|---|
| | **¿Prueba de despliegue?** | **Si** |
| **Descripción:**<br>Esta prueba tiene como finalidad verificar la disponibilidad y correcto funcionamiento del botón de envió de conteos en el aplicativo móvil. | | |
| **Prerrequisitos**<br>Teléfono móvil con Sistema operático Android, con la aplicación de prueba instalada, cámara y geolocalización disponibles. | | |
| **Pasos:**<br>4. Abrir la aplicación móvil.<br>5. Aceptar los permisos para acceso a la cámara y geolocalización.<br>6. Enfocar la cámara del teléfono a un espacio público con flujo vehicular.<br>7. Una vez los rectángulos de detección han sido dibujados en la mayoría de los vehículos detectados en el entorno presionar el botón “Enviar Conteos”. | | |

| |
|---|
| **Resultado esperado:**<br>2. Luego de presionar el botón “Enviar Conteos”, se debe cargar un loader en la pantalla principal indicando que los conteos están siendo efectuados y enviados a la API, luego se debe visualizar un mensaje de conformación que verifica que los conteos fueron enviados y almacenados correctamente. |
| **Resultado obtenido:**<br>Se cumple con el resultado esperado en todas sus fases |

Fuente: Elaboración propia

**Tabla 8.** *Caso prueba 4.*

| **Pruebas de funcionamiento de la API (Método POST)** | **4** | |
|---|---|---|
| | **¿Prueba de despliegue?** | **Si** |
| **Descripción:**<br>Esta prueba tiene como finalidad verificar que los datos generados y enviados por el usuario fueron almacenados de forma correcta en la base de datos DynamoDB | | |
| **Prerrequisitos**<br>8. Teléfono móvil con Sistema operático Android, con la aplicación de prueba instalada, cámara y geolocalización disponibles.<br>9. API configurada y desplegada en AWS con conexión a la base de datos DynamoDB. | | |
| **Pasos:**<br>10. Abrir la aplicación móvil.<br>11. Aceptar los permisos para acceso a la cámara y geolocalización.<br>12. Enfocar la cámara del teléfono a un espacio público con flujo vehicular.<br>13. Una vez los rectángulos de detección han sido dibujados en la mayoría de los vehículos detectados en el entorno presionar el botón “Enviar Conteos”. | | |

| |
|---|
| 14. Abrir el dashboard de AWS y acceder a la consola de DynamoDB.<br>15. Acceder a la tabla que almacena los conteos vehiculares. |
| **Resultado esperado:**<br>Luego de acceder a la tabla que almacena los conteos vehiculares y coordenadas geográficas del usuario, los datos enviados estar registrados. |
| **Resultado obtenido:**<br>Se cumple con el resultado esperado en todas sus fases |

Fuente: Elaboración propia

**Tabla 9.** *Caso prueba 5.*

| **Pruebas de funcionamiento de la API (Método GET)** | **4** | |
|---|---|---|
| | **¿Prueba de despliegue?** | **Si** |
| **Descripción:**<br>Esta prueba tiene como finalidad verificar el funcionamiento de la API y el endpoint del método GET que permite que los datos almacenados sean accesibles a consultas y peticiones externas. | | |
| **Prerrequisitos**<br>Programa POSTMAN, API configurada y desplegada en AWS con conexión a la base de datos DynamoDB. | | |
| **Pasos:**<br>16. Abrir la aplicación POST-MAN.<br>17. En el apartado de petición seleccionar tipo GET y en url colocar la dirección del endpoint de la API configurada para retornar los datos de conteos.<br>18. Presionar el botón enviar | | |

| |
|---|
| **Resultado esperado:**<br>Luego de presionar el botón enviar se debe recibir como respuesta un array de objetos con los conteos y coordenadas disponibles en la base de datos. |
| **Resultado obtenido:**<br>Se cumple con el resultado esperado en todas sus fases |

Fuente: Elaboración propia

**Tabla 10.** *Caso prueba 6.*

| **Pruebas de la aplicación Web** | **6** | |
|---|---|---|
| | **¿Prueba de despliegue?** | **Si** |
| **Descripción:**<br>Esta prueba tiene como finalidad verificar el acceso y funcionamiento de la página Web de consulta de datos. | | |
| **Prerrequisitos**<br>Dispositivo móvil o pc con acceso a internet. | | |
| **Pasos:**<br>1. Abrir la url de la aplicación web de consulta de datos | | |
| **Resultado esperado:**<br>Luego de abrir la url, se debe visualizar en la pantalla dos componentes básicos:<br>• Debe estar presenta un mapa para la navegación y consulta de los conteos.<br>• Debe estar presente un botón para descargar los datos de los conteos vehiculares. | | |
| **Resultado obtenido:**<br>Se cumple con el resultado esperado en todas sus fases | | |

Fuente: Elaboración propia

**Tabla 11.** *Caso prueba 7.*

| **Pruebas en la aplicación Web** | **7** | |
|---|---|---|
| | **¿Prueba de despliegue?** | **Si** |
| **Descripción:**<br>Esta prueba tiene como finalidad verificar el acceso y funcionamiento de la página Web de consulta de datos. | | |
| **Prerrequisitos**<br>Dispositivo móvil o pc con acceso a internet. | | |
| **Pasos:**<br>2. Abrir la url de la aplicación web de consulta de datos | | |
| **Resultado esperado:**<br>Luego de abrir la url, se debe visualizar en la pantalla dos componentes básicos:<br>• Debe estar presenta un mapa para la navegación y consulta de los conteos.<br>• Debe estar presente un botón para descargar los datos de los conteos. | | |
| **Resultado obtenido:**<br>Se cumple con el resultado esperado en todas sus fases | | |

Fuente: Elaboración propia

**Tabla 12.** *Caso prueba 8.*

| **Pruebas en la aplicación Web** | **6** | |
|---|---|---|
| | **¿Prueba de despliegue?** | **Si** |
| **Descripción:**<br>Esta prueba tiene como finalidad verificar que los datos de conteos vehiculares son visualizados | | |

| |
|---|
| mediante arcadores en el mapa de la página. |
| **Prerrequisitos**<br>Dispositivo móvil o pc con acceso a internet. |
| **Pasos:**<br>1. Abrir la url de la aplicación web de consulta de datos<br>2. Navegar por el mapa de la aplicación y seleccionar la ubicación de un lugar desde el cual se envió un conteo vehicular en las pruebas anteriores haciendo uso del teléfono móvil, el conteo debe ser ubicado por un marcador de color en el sitio desde el cual se reportó la información<br>3. Dar clic en el marcador. |
| **Resultado esperado:**<br>Una vez seleccionado un marcador debe desplegarse un modal con un texto que indique el número de conteos de cada tipo de vehículo reportado en la zona. |
| **Resultado obtenido:**<br>Se cumple con el resultado esperado en todas sus fases |

Fuente: Elaboración propia

**Tabla 13.** *Caso prueba 9*.

| **Descarga de datos de la aplicación Web** | **9** | |
|---|---|---|
| | **¿Prueba de despliegue?** | **Si** |
| **Descripción:**<br>Esta prueba tiene como finalidad verificar que los datos de conteos vehiculares son descargados de forma correcta desde la aplicación Web. | | |
| **Prerrequisitos** | | |

| |
|---|
| Dispositivo móvil o pc con acceso a internet. |
| **Pasos:**<br>4. Abrir la url de la aplicación web de consulta de datos.<br>5. Buscar en a la pantalla principal la opción descargar datos.<br>6. Dar clic en el botón. |
| **Resultado esperado:**<br>Una vez seleccionada la opción de descarga de datos, automáticamente el sistema deberá descargar un archivo en formato cvs que al abrir contendrá la información de las coordenadas y conteos vehiculares almacenados en la base de datos. |
| **Resultado obtenido:**<br>Se cumple con el resultado esperado en todas sus fases |

Fuente: Elaboración propia

## 5.2. Pruebas de satisfacción y usabilidad

Considerando que el presente TFM tienen como finalidad desarrollar un sistema de generación de datos abiertos que involucra la participación de la comunidad e interesados en el ámbito de la movilidad resulta de gran importancia que el sistema se adecue a las necesidades de los usuarios resultando intuitivo y fácil de utilizar al momento de generar y reportar datos. Considerando esta premisa se ha efectuado un conjunto de pruebas de usabilidad que consideran las siguientes métricas para el análisis (Wikipedia, 2020):

**Exactitud:** Considera la cantidad errores generados por los usuarios o la aplicación durante la etapa de prueba y considera si la ampliación se recuperó de los mismos.

**Tiempo** necesario para completar las pruebas.

**Recuerdo:** establece en porcentaje que el usuario recuerda acerca del uso de la aplicación después de un periodo de haberla utilizado.

**Respuesta emocional:** El estado de ánimo de los usuarios al culminar las pruebas.

**Tabla 14.** *Cuestionario de usabilidad y satisfacción de uso del Sistema.*

| | Nivel de Satisfacción | | | | | | |
|---|---|---|---|---|---|---|---|
| Pregunta | 1 | 2 | 3 | 4 | 5 | sí/ no | Especifique en caso de requerirlo |
| ¿Existieron problemas al instalar la aplicación en su dispositivo móvil? | | | | | | | |
| ¿Qué tan intuitiva le pareció la interfaz de aplicativo móvil? | | | | | | | |
| ¿Le resulto fácil detectar y reportar conteos vehiculares? | | | | | | | |
| ¿El desempeño de la aplicación fue fluido y rápido en su dispositivo móvil? | | | | | | | |
| ¿Se detectaron vehículos y peatones en la escena de prueba en la que se probó la aplicación? | | | | | | | |
| ¿Le pareció amigable la aplicación web de consulta de datos? | | | | | | | |
| ¿Tuvo problemas para visualizar o encontrar datos de conteos vehiculares en la aplicación web? | | | | | | | |
| ¿Le resulto fácil localizar conteos en el mapa de la aplicación web? | | | | | | | |
| ¿Tuvo problemas para descargar los datos desde el aplicativo web? | | | | | | | |
| ¿Cree que una iniciativa de recolección de datos de movilidad abierta puede aportar con soluciones a los problemas de movilidad actuales de su ciudad? | | | | | | | |
| ¿Recomendaría el uso del aplicativo móvil o web a otra persona? | | | | | | | |
| ¿Qué aspectos cree que se deberían mejorar en la aplicación móvil? | | | | | | | |
| ¿Qué aspectos cree que se deberían mejorar en la aplicación web? | | | | | | | |
| ¿En términos generales como le pareció el funcionamiento global del sistema? | | | | | | | |
| ¿En qué tipo de dispositivo probó la aplicación móvil? | | | | | | | |

Fuente: Elaboración propia

Para evaluar las métricas antes descritas se considera una muestra de 20 personas voluntarias, la cuales colaboraron en el proceso de prueba de la aplicación móvil y Web., los usuarios seleccionados procedieron a instalar la aplicación y reportar datos de conteos vehiculares, de igual modo consultaron la información generada en la plataforma Web. Al finalizar las pruebas cada uno de los participantes aportó con su criterio acerca el suso de los aplicativos mediante la encuesta de la Tabla 14.

Los resultados obtenidos del proceso de pruebas y reportados en el cuestionario de satisfacción se clasificaron y mapearon dentro de las métricas de calidad descritas en al Tabla 15 la cual resume el porcentaje y nivel de satisfacción de los usuarios:

**Tabla 15.** *Métricas de usabilidad del sistema de detección y conteo vehicular.*

| Exactitud app móvil (% Errores) | Exactitud app web (% Errores) | Tiempo Promedio | Recuerdo | % satisfacción global | Tipo de dispositivo | Apoyo a la iniciativa |
|---|---|---|---|---|---|---|
| 10% | 5% | 40 seg (app movil) 3.5min (app web) | 90% | 95% | 60% gama alta<br>20% gama baja<br>20% gama media | 95% |

Fuente: Elaboración propia

Se obtuvieron resultados favorables en el conjunto de pruebas realizadas, tan solo dos de cada 20 usuarios presentaron problemas al iniciar a la aplicación en su teléfono, los errores estuvieron relacionados al tipo de dispositivo utilizado en estos casos los dispositivos empleados fueron teléfonos de gama baja lo cual afecto la velocidad del procesamiento, detección y conteo de vehículos en los dispositivos. Otro aspecto importante que se analizó fue el tiempo promedio que le toma a un usuario ejecutar el ciclo de conteo y envió de datos desde la aplicación a la API, para las pruebas realizadas no se superó los 40 seg en promedio, lo cual es un claro indicador de que el aplicativo resulta intuitivo y fácil de utilizar.

En general un 95% de los usuarios al final de las pruebas reportaron en el cuestionario de satisfacción que la aplicación móvil es intuitiva y fácil de utilizar. Para el caso de la aplicación Web el tiempo que le tomo a aun usuario consultar los datos generados por el aplicativo móvil fue en promedio de 3.5min, lo cual es un tiempo bueno considerando que los usuarios deben buscar en el mapa del aplicativo el lugar desde el cual reportaron los datos. Otro aspecto importante es la métrica denominada “recuerdo” que permite determinar el porcentaje de usuarios que pasado un tiempo recordaron el proceso de uso del aplicativo, para este caso se preguntó a los voluntarios luego de 5 días si recordaban el proceso de generación y consulta de datos en el sistema obteniendo como resultado que el 90 % de los participantes recordaba aun los pasos a seguir.

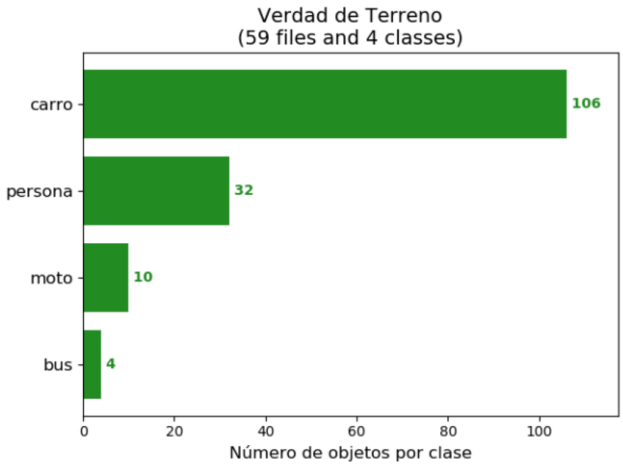
## 5.3. Análisis de precisión en la detección y conteo de vehículos

Para evaluar la precisión del proceso de conteo de vehículos se emplearon las métricas precisión (PRE), recall (REC) y F1 score (F1) las cuales se definen con detalle en (Chen et al., 2012) y consideran la cantidad falsos positivos, verdaderos positivos y falsos negativos involucrados en pruebas donde se tiene un conjunto de datos de referencia y un conjunto de datos de prueba. Estas métricas se definen en las ecuaciones 1-3, donde TP representa la cantidad de verdaderos positivos, FP los falsos positivos y FN representa a los falsos negativos.

Para efectuar las pruebas se generó un conjunto de datos de referencia (ground truth) para lo cual se generó una grabación de video de un escenario de prueba con vehículos y peatones de 15min de duración, luego se procedió a efectuar el conteo manual de los vehículos y peatones en la secuencia de video, con ello se obtuvo la cantidad real de vehículos y peatones en la escena, los resultados de este proceso se resumen en la Figura 24.

Posterior se procesó la misma secuencia de video en un teléfono móvil de gama media (xiaomi note 9) utilizando la aplicación con el algoritmo de detección y conteo vehicular, lo cual se generó un conjunto de datos con las detecciones del algoritmo. Con esta información se determinó los TP, FP y FN generados por el algoritmo de detección, los cuales se utilizaron en el cálculo de PRE, REC y F1 score. Los resultados se presentan en la Figura 25.

**Figura 24.** *Conteos reales en la escena de prueba.*

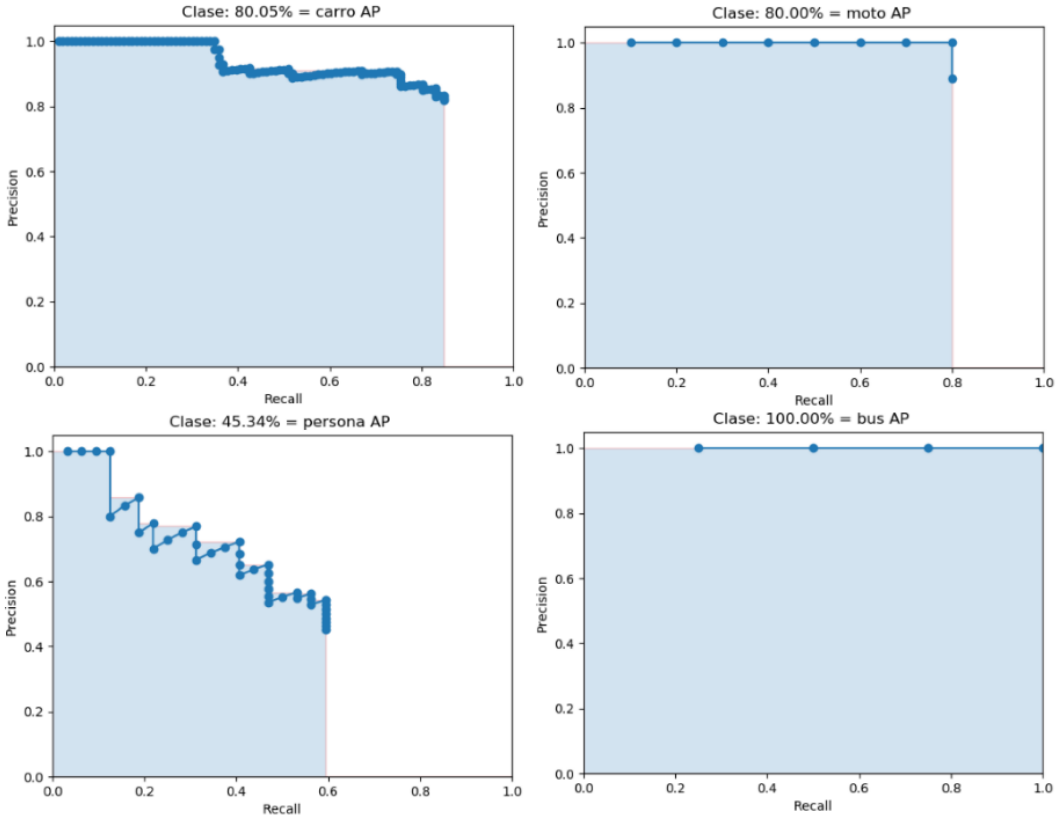



Fuente: Elaboración propia

$$PRE = \frac{TP}{TP+FP} \ (1) \qquad REC = \frac{TP}{TP+FN} \ (2)$$

$$F1 = \frac{2 * REC * PRE}{REC + PRE} (3)$$

**Figura 25**. *Métricas PRE, REC y AP del sistema.*




Fuente: Elaboración propia

En la Figura 25 se observan las gráficas resultantes del cálculo de PRE y REC considerando que en un escenario de detección perfecto estos valores serán 1, adicionalmente se calcula la métrica "Average Precision" (AP) la cual es representada por el área bajo la curva de las gráficas, AP es una medida que da una idea de la exactitud del proceso de detección en el escenario de prueba analizado.

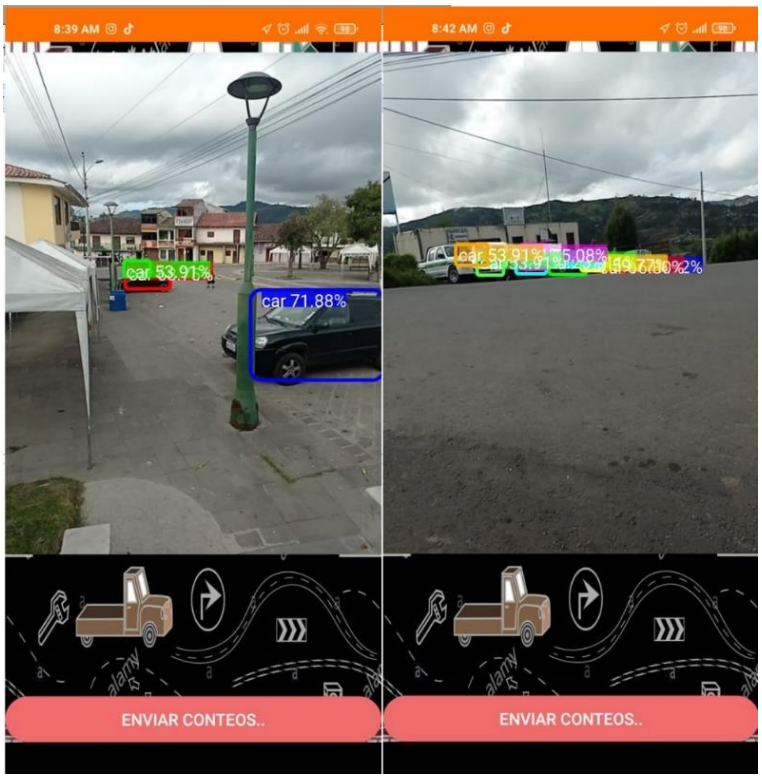
Se aprecia que el algoritmo de detección y conteo SSD-Mobileent genera métricas de PRE y REC muy cercanas a 1 (para carros, motos y buses), el mejor de los casos se dio en la detección de autobuses alcanzando un 100% de precaución en la detección y conteo, el caso más crítico

fue la detección de personas, en la cual se alcanzó un AP de 45%. En términos generales el algoritmo de detección SSD-Mobilent reportó un 77% de precisión para el escenario de prueba utilizado.

### 5.3.1. Presentación de resultados

En esta sección se presentan los resultados del Sistema de conteo vehicular bajo condiciones de prueba en un escenario real, la Figura 26 presenta la interfaz de la aplicación móvil en sus distintas etapas de trabajo: proceso de detección, conteo y envió de datos a la API.

**Figura 26.** *Interfaz de detección y envío de datos de la aplicación.*

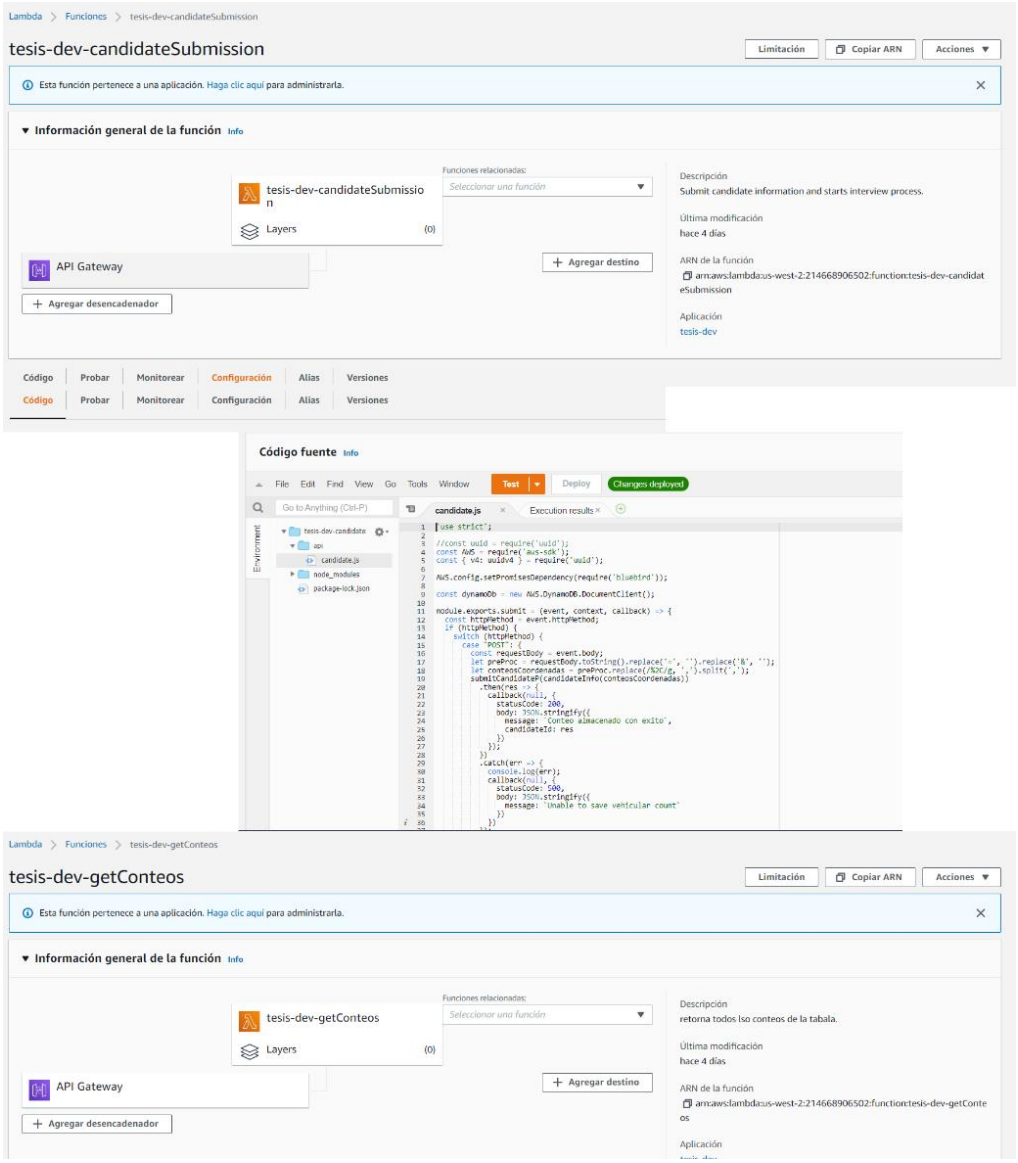



Fuente: Elaboración propia

La interfaz del aplicativo es simple y el usuario final únicamente debe enfocar los vehículos de la carretera mediante la cámara del teléfono y presionar el botón enviar, el proceso de detección, conteo y envío de datos es automático y transparente al usuario.

Una vez los datos de conteo son enviados por el usuario, la petición es procesada por el endpoint encargado de manejar las peticiones de tipo POST y enviar los datos a la base de datos, en la Figura 27 se presenta la estructura de los endpoint que procesan las peticiones POST y GET en el servicio serverless de AWS.

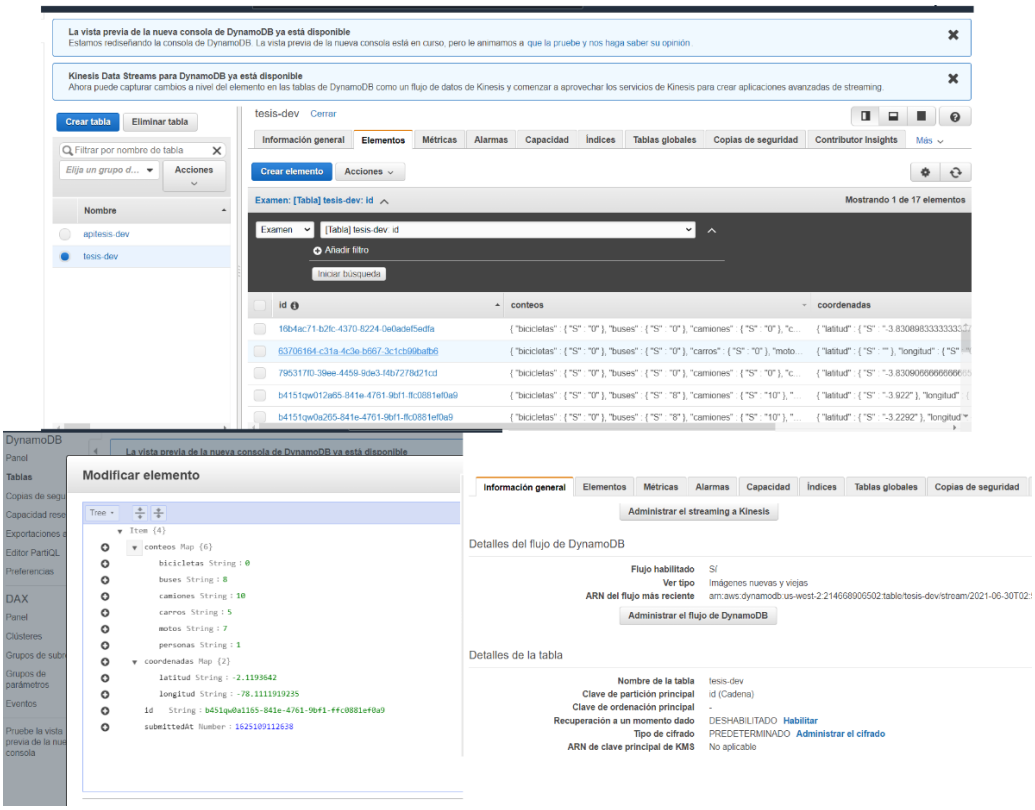
**Figura 27.** *Endpoints serverless POST y GET en AWS.*




Fuente: Elaboración propia

Una vez los datos de los usuarios son procesados por el endpoint, se envían a la base de datos DynamoDB donde se almacenan en una tabla para su posterior consulta, la Figura 28 presenta la estructura de la tabla de almacenamiento de datos y su configuración en AWS.

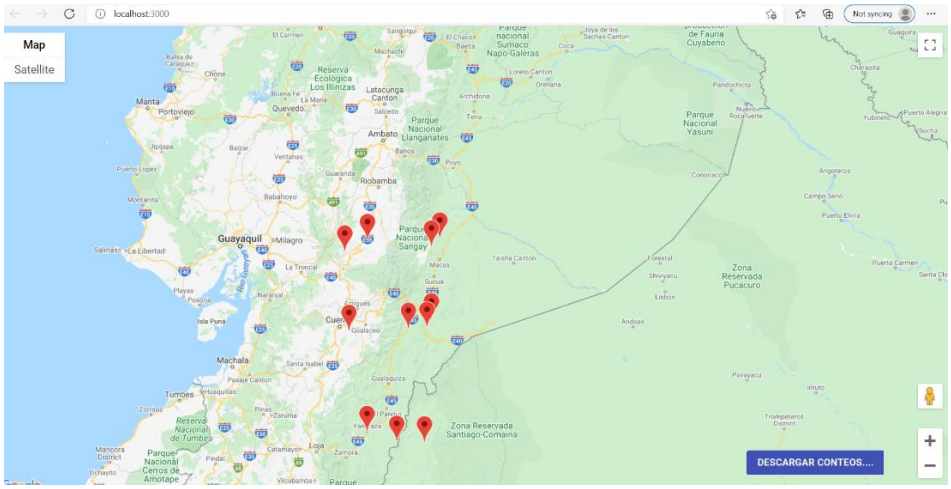
**Figura 28.** *Estructura y configuración de la base de datos DynamoDB.*




Fuente: Elaboración propia

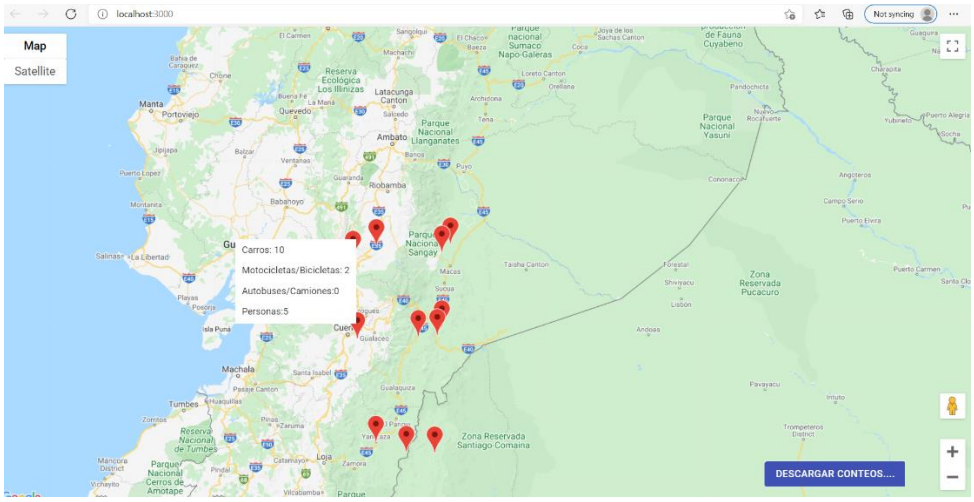
El aplicativo Web implementado para la consulta, visualización y descarga de los datos de conteos se presenta en las Figuras 29 y 30, se observa que los conteos almacenados se representan en un mapa mediante unos marcadores de color rojo, los cuales indican la ubicación y lugar de captura de los conteos.

**Figura 29.** *Aplicación Web de consulta de datos con marcadores de conteo.*




Fuente: Elaboración propia

**Figura 30.** *Modal de consulta de conteos asociado a un marcador.*




Fuente: Elaboración propia

Al hacer clic en el marcador se despliega un pequeño modal (Figura 30), en el cual se indica el número de vehículos presentes en la zona. Como se puede observar en las gráficas en la parte inferior derecha se presenta un botón el cual implementa la funcionalidad de descarga de conteos en formato csv.

# 6. CONCLUSIONES Y TRABAJO FUTURO

## 6.1. Conclusiones

El Sistema de detección y conteo vehicular implementado en el presente TFM satisface los requerimientos mínimos y objetivos planteados en los capítulos III e historia de usuario descritas en el capítulo IV, cabe destacar que la adopción de una metodología como SCRUM fue de gran utilidad para el desarrollo ordenado de cada una de las historias de usuario planteadas. Uno de problemas que se generaron al crear el sistema fue el proceso de implementación de la red neuronal en un dispositivo Android, si bien el modelo SSD provee soporte para este tipo de dispositivos, existieron problemas para configurar las librerías relacionados con las versiones más recientes de Android. En lo referente a la aplicación Web el uso de serverless y DynamoDB permitieron agilizar el desarrollo, al ser soluciones en la nube evitaron el proceso de contratación y configuración de servidores propios lo cual demanda gran cantidad de tiempo y dinero.

Este trabajo representa un aporte en lo referente al diseño de un sistema de conteos vehiculares en todas sus etapas debido a que provee los medios para la captura, generación y consulta de información en tiempo real, los datos generados se presentan de libre acceso a la comunidad siendo de utilidad para personas y grupos interesados en contribuir con soluciones de movilidad, además se fomenta una cultura de colaboración y generación de datos de libre acceso. Uno de los principales aportes que se pudo evidenciar en este trabajo es el uso de técnicas basadas en redes neuronales convolucionales aplicadas a dispositivos móviles para la ejecución de conteos vehiculares. En la actualidad la información y datos relacionados con trabajos que implementen iniciativas orientadas al conteo de vehículos es escasa o nula, existen plataformas que se orientan al conteo de otro tipo objetos implementando aplicaciones basadas en computadoras de alto rendimiento y en pocos casos teléfonos inteligentes. Por último, es necesario aclarar que, si bien una inactiva de crowdsourcing incluye e incentiva a la comunidad a contribuir de forma masiva, para garantizar su éxito esta debe ser divulgada a la sociedad.

Finalmente, con la ejecución del presente trabajo se cumplió de forma satisfactoria el objetivo principal planteado en el capítulo III, el cual consiste en implementar una iniciativa de Crowdsourcing haciendo uso de la capacidad de procesamiento de los dispositivos móviles Android para el despliegue de una red neural convolucional con la capacidad de procesar y generar datos de conteos vehiculares y peatones en carreteras, los resultados de la consecución del objetivo se describen en el capítulo IV y V.

Otro objetivo desarrollado fue la implementación de un aplicativo Web de consulta y visualización de datos de conteos vehiculares en tiempo real, para la consecución del objetivo se implementó una API encargada del manejo de peticiones de escritura y consulta de datos en conjunto con una interfaz gráfica de consulta de datos, para lo cual se utilizaron tecnologías tales como serverless, react JS y almacenamiento de datos en la nube con DynamoDB, el capitulo V describen el proceso de implementación de la solución planteada. La consecución de los objetivos antes descritos permitió alcanzar la finalidad del TFM que consiste en proveer un sistema completo de generación y consulta de datos abiertos a la comunidad interesada en soluciones de movilidad.

## 6.2. Trabajo futuro

El Sistema de software propuesto en el presente TFM consideró el desarrollo e investigación en diversas áreas, una de la más relevantes fue el estudio de las técnicas de detección y reconocimiento de objetos basada en redes neuronales convolucionales, para el presente trabajo se implementó una red neuronal pre-entrenada basada en el modelo SSD Mobilenet, la cual proporciono las métricas de calidad descritas en el capítulo V, una mejora en este ámbito para futuras implementaciones podría involucrar el entrenamiento de una red neuronal propia con un data-set de imágenes capturadas en las áreas de detección bajo la cuales trabajará el sistema, con ello la tasa de detección y la capacidad de inferencia del modelo mejoraran notalmente.

En lo referente al aplicativo Web en futuras implementaciones se podría incorporar una funcionalidad que permita al usuario segregar los datos de los conteos por zonas de interés y

descargar la información de dicha área, actualmente se permite la descarga de todos los datos de conteos disponibles en la plataforma.

Si bien no se encuentra en el alcance del presente TFM, resulta de gran interés el comparar el rendimiento y las métricas de detección del modelo SSD-Mobilenet utilizado con otro tipo de modelos neuronales, existen alternativas a SSD-Mobilent como R-CNN, GoogleLenet o R-FCN que podrían generar resultados superiores al modelo utilizado en este trabajo.

# Referencias Bibliográficas.

AWS. (2019). Powering Gaming Applications with Amazon DynamoDB. Retrieved April 7, 2021, from https://aws.amazon.com/es/blogs/big-data/powering-gaming-applications-with-amazon-dynamodb/

Axiomtek. (2021). Vehicle Detection System. Retrieved from https://www.axiomtek.com/Default.aspx?MenuId=Solutions&FunctionId=SolutionView&ItemId=36&Title=Vehicle+Detection+System

Board, T. R., of Sciences Engineering, & Medicine. (2014). *Guidebook on Pedestrian and Bicycle Volume Data Collection*. (P. Ryus, E. Ferguson, K. M. Laustsen, R. J. Schneider, F. R. Proulx, T. Hull, & L. Miranda-Moreno, Eds.). Washington, DC: The National Academies Press. https://doi.org/10.17226/22223

Cai, Y., Li, H., Yuan, G., Niu, W., Li, Y., Tang, X., … Wang, Y. (2020). YOLObile: Real-Time Object Detection on Mobile Devices via Compression-Compilation Co-Design. *CoRR*, *abs/2009.0*. Retrieved from https://arxiv.org/abs/2009.05697

Cen, Q., Pan, Z., Li, Y., & Ding, H. (2019). Laryngeal Tumor Detection in Endoscopic Images Based on Convolutional Neural Network. In *2019 IEEE 2nd International Conference on Electronic Information and Communication Technology (ICEICT)* (pp. 604–608). https://doi.org/10.1109/ICEICT.2019.8846399

Chan, J., Lee, J. A., & Kemao, Q. (2018). Real-Time Texture-less Object Recognition on Mobile Devices. In *2018 24th International Conference on Pattern Recognition (ICPR)* (pp. 3273–3278). https://doi.org/10.1109/ICPR.2018.8545158

Chen, Z., Ellis, T., & Velastin, S. A. (2012). Vehicle detection, tracking and classification in urban traffic. In *2012 15th International IEEE Conference on Intelligent Transportation Systems* (pp. 951–956). https://doi.org/10.1109/ITSC.2012.6338852

Choi, I.-H., Hong, S. K., & Kim, Y.-G. (2016). Real-time categorization of driver's gaze zone using the deep learning techniques. In *2016 International Conference on Big Data and Smart Computing (BigComp)* (pp. 143–148). https://doi.org/10.1109/BIGCOMP.2016.7425813

Cortes, C., & Vapnik, V. (1995). Support-Vector Networks. In *Machine Learning* (pp. 273–297).

Díaz, A. R. (2014). *CROWDPROJECTS: Caracterización y Clasificación de Proyectos Colaborativos*. UNIVERSIDAD DE OVIEDO.

El Alaoui El Abdallaoui, H., El Fazziki, A., Sadiq, A., Zohra, E. F., & Sadgal, M. (2016). A web based crowdsourcing framework: Lost child case. In *2016 4th IEEE International Colloquium on Information Science and Technology (CiSt)* (pp. 263–268). https://doi.org/10.1109/CIST.2016.7805053

Espinoza, F. T., Gabriel, B. G., & Barros, M. J. (2017). Computer vision classifier and platform for automatic counting: More than cars. In *2017 IEEE Second Ecuador Technical Chapters Meeting (ETCM)* (pp. 1–6). https://doi.org/10.1109/ETCM.2017.8247454

Fernández, E. G. (n.d.). AFOROS DE TRÁFICO MEDIANTE EL RECONOCIMIENTO AUTOMÁTICO DE MATRÍCULAS. Retrieved from https://www.altergeosistemas.com/blog/2020/12/17/aforos-de-trafico-mediante-el-reconocimiento-automatico-de-matriculas/

Heredia, A., & Barros-Gavilanes, G. (2019). Video processing inside embedded devices using

SSD-Mobilenet to count mobility actors. In *2019 IEEE Colombian Conference on Applications in Computational Intelligence (ColCACI)* (pp. 1–6). https://doi.org/10.1109/ColCACI.2019.8781798

HMI. (2021). Road & Traffic Systems. Retrieved from https://www.hmi.co.nz/en-nz/products/road-traffic-systems/vehicle-detection-system

Howe, & Jeff. (2006). The Rise of Crowdsourcing. *Wired*, *14*.

ICOMS. (2021). Motorised vehicles. Retrieved June 6, 2021, from https://icomsdetections.com/targets/motorised-vehicles/

IntuVision. (2021). intuVision® VA - Solution Modules. Retrieved from https://www.intuvisiontech.com/

Jilani, M. T., Ur Rehman, M. Z., & Abbas, M. A. (2019). An Application Framework of Crowdsourcing based Emergency Events Reporting in Smart Cities. In *2019 International Symposium on Networks, Computers and Communications (ISNCC)* (pp. 1–5). https://doi.org/10.1109/ISNCC.2019.8909146

Levannoza, R., Latif, R. F., Nurkafianti, S. I., Oktriono, K., Devina, Wiharja, C. K., & Cenggoro, T. W. (2020). Enabling a Massive Data Collection for Hotel Receptionist Chatbot Using a Crowdsourcing Information System. In *2020 International Conference on Information Management and Technology (ICIMTech)* (pp. 213–217). https://doi.org/10.1109/ICIMTech50083.2020.9211162

Liang, C., Li, Y., & Luo, J. (2016). A Novel Method to Detect Functional microRNA Regulatory Modules by Bicliques Merging. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, *13*(3), 549–556. https://doi.org/10.1109/TCBB.2015.2462370

Ma, N., & Chia, C. S. (2007). False Alarm Reduction by LS-SVM for Manmade Object Detection from Sidescan Sonar Images. In *OCEANS 2007 - Europe* (pp. 1–6). https://doi.org/10.1109/OCEANSE.2007.4302234

Mairittha, N., & Inoue, S. (2019). Crowdsourcing System Management for Activity Data with Mobile Sensors. In *2019 Joint 8th International Conference on Informatics, Electronics Vision (ICIEV) and 2019 3rd International Conference on Imaging, Vision Pattern Recognition (icIVPR)* (pp. 85–90). https://doi.org/10.1109/ICIEV.2019.8858566

Maitre, E. (2017). Safety problems and difficulties in using public spaces with modern tramways: the example of Marseilles. *8th Young Researchers' Seminar 2017 - ECTRI-FEHRL-FERSI,Berlin, Germany. 16 P*, *1*, 17. Retrieved from https://hal.archives-ouvertes.fr/hal-01658471/document

Moreno, P. L. R. (n.d.). El arte del crowdsourcing. Retrieved November 4, 2021, from https://universoabierto.org/2016/07/25/el-arte-del-crowdsourcing/#:~:text=El término crowdsourcing fue acuñado,es —abastecimiento de la multitud.

NPM. (2021). google-maps-loader. Retrieved from https://www.npmjs.com/package/google-maps

OTHMAN, N. A., & AYDIN, I. (2018). A New Deep Learning Application Based on Movidius NCS for Embedded Object Detection and Recognition. In *2018 2nd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)* (pp. 1–5). https://doi.org/10.1109/ISMSIT.2018.8567306

Peña-González, R. H., & Nuño-Maganda, M. A. (2014). Computer vision based real-time vehicle tracking and classification system. In *2014 IEEE 57th International Midwest Symposium*

*on Circuits and Systems (MWSCAS)* (pp. 679–682). https://doi.org/10.1109/MWSCAS.2014.6908506

Raju, R., Harinishree, M., & Lavanya, S. (2017). The crowdsourcing systems survey work. In *2017 International Conference on Computation of Power, Energy Information and Commuincation (ICCPEIC)* (pp. 357–360). https://doi.org/10.1109/ICCPEIC.2017.8290392

Raquel Flórez López, Jose Miguel Fernandez, J. M. F. F. (2008). *Las Redes Neuronales Artificiales* (1st ed.). Retrieved from https://books.google.es/books?hl=es&lr=&id=X0uLwi1Ap4QC&oi=fnd&pg=PA9&dq=redes+neuronales&ots=gOHCgsjvYm&sig=W-1uMZNhtKP1knIq6pRdwQdM3xA#v=onepage&q=redes&f=false

Real, E., Moore, S., Selle, A., Saxena, S., Suematsu, Y. L., Tan, J., … Kurakin, A. (2017). Large-Scale Evolution of Image Classifiers. In D. Precup & Y. W. Teh (Eds.), *Proceedings of the 34th International Conference on Machine Learning* (Vol. 70, pp. 2902–2911). PMLR. Retrieved from http://proceedings.mlr.press/v70/real17a.html

Roh, M.-C., & Lee, J. (2017). Refining faster-RCNN for accurate object detection. In *2017 Fifteenth IAPR International Conference on Machine Vision Applications (MVA)* (pp. 514–517). https://doi.org/10.23919/MVA.2017.7986913

Saha, S. (2018). A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way. Retrieved from https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

Sanjay Kumar, K. K. R., Subramani, G., Thangavel, S. K., & Parameswaran, L. (2021). A Mobile-Based Framework for Detecting Objects Using SSD-MobileNet in Indoor Environment BT - Intelligence in Big Data Technologies—Beyond the Hype. In J. D. Peter, S. L. Fernandes, & A. H. Alavi (Eds.) (pp. 65–76). Singapore: Springer Singapore.

Shan, R., & Li, T. (2018). Reliability Simulation of Mobile Wireless Sensor Networks Based on Support Vector Machines. In *2018 14th International Conference on Computational Intelligence and Security (CIS)* (pp. 269–272). https://doi.org/10.1109/CIS2018.2018.00066

Shuai, Q., & Wu, X. (2020). Object detection system based on SSD algorithm. In *2020 International Conference on Culture-oriented Science Technology (ICCST)* (pp. 141–144). https://doi.org/10.1109/ICCST50977.2020.00033

Solutions, A. T. (2021). VEHICLE COUNT AND CLASSIFY SOLUTIONS. Retrieved from https://www.alltrafficsolutions.com/solutions/vehicle-count-classify-solutions/

Stevens, Q. (2009). “Broken” public spaces in theory and in practice. *Town Planning Review*, *80*, 371–392. https://doi.org/10.3828/tpr.2009.3

Tecnologica, R. (n.d.). ¿Cómo Google usa la inteligencia artificial sin que usted se dé cuenta? Retrieved from https://www.semana.com/tecnologia/articulo/como-google-usa-la-inteligencia-artificial-sin-que-usted-se-de-cuenta/584810/

TensorFlow Hub. (2021). ssd_mobilenet_v2. Retrieved May 6, 2021, from https://tfhub.dev/tensorflow/ssd_mobilenet_v2/2

TRIGG. (2021). Piezoelectric Sensor: Roadtrax “Brass Linguini” (BL) Axle Sensor. Retrieved from https://www.triggindustries.com/traffic-engineering-products/axle-sensors.html

Tutorials, R. (2018). No Title. Retrieved from https://techvidvan.com/tutorials/svm-in-r/

Wang, G., Rister, B., & Cavallaro, J. (2013). Workload Analysis and Efficient OpenCL-based

Implementation of SIFT Algorithm on a Smartphone. In *2013 IEEE Global Conference on Signal and Information Processing, GlobalSIP 2013 - Proceedings*. https://doi.org/10.1109/GlobalSIP.2013.6737002

Wang, J., Jiang, S., Song, W., & Yang, Y. (2019). A Comparative Study of Small Object Detection Algorithms. In *2019 Chinese Control Conference (CCC)* (pp. 8507–8512). https://doi.org/10.23919/ChiCC.2019.8865157

Wei, X., Wang, S., Zhou, A., Xu, J., Su, S., Kumar, S., & Yang, F. (2017). MVR: An Architecture for Computation Offloading in Mobile Edge Computing. In *2017 IEEE International Conference on Edge Computing (EDGE)* (pp. 232–235). https://doi.org/10.1109/IEEE.EDGE.2017.42

Wikipedia. (2020). Prueba de usabilidad. Retrieved June 7, 2021, from https://es.wikipedia.org/wiki/Prueba_de_usabilidad

Wu, S., & Zhang, L. (2018). Using Popular Object Detection Methods for Real Time Forest Fire Detection. In *2018 11th International Symposium on Computational Intelligence and Design (ISCID)* (Vol. 01, pp. 280–284). https://doi.org/10.1109/ISCID.2018.00070

Yang, Y., Wang, J., & Yang, Y. (2012). Improving SVM classifier with prior knowledge in microcalcification detection1. In *2012 19th IEEE International Conference on Image Processing* (pp. 2837–2840). https://doi.org/10.1109/ICIP.2012.6467490

# Crowdsourcing y datos libres mediante la detección de vehículos y pedestres utilizando redes neuronales convolucionales en smartphones.

Andrés Heredia Álvarez

Universidad Internacional de la Rioja, Logroño (España)

14/07/2021

## Resumen

El presente trabajo presenta la implementación de un Sistema de generación y consulta de datos de movilidad (conteos vehiculares) que comprende las etapas de generación, reporte y consulta de conteos geo-referenciados en tiempo real mediante la creación de un aplicativo móvil que hace uso de un algoritmo de detección de objetos basado en redes neuronales convolucionales para la detección de vehículos (coches, motocicletas, peatones y trasporte pesado) en la carretera mediante la cámara de un teléfono movil y la implementación de un aplicativos Web basado en mapas para la consulta de la información reportada. Con los datos generados se efectúa la implementación de un sistema "crodwsourcing" de datos y colaboración abierta distribuida como elemento esencial en la participación y cultura ciudadana

## Palabras Clave

Convolucional, Crowdsourcing, Mobilenet SSD, movilidad sostenible, red neuronal.

## I. Introducción

Actualmente la información y datos de movilidad permiten solucionar problemas relacionados con la congestión de tráfico, el mal uso de espacios destinados a circulación vehicular o peatonal, uso incorrecto de las carreteras, gestión de espacios destinados a ciclistas, carencia de transporte público urbano, entre otros. En la mayoría de las ciudades la información sobre conteos de flujos vehiculares en diferentes puntos es escasa o limitada y su acceso está restringido. El no disponer de repositorios de acceso libre a información de movilidad limita en gran medida las tareas de diferentes grupos de investigación y personas interesadas en la generación de soluciones y propuestas que permitan alcanzar una movilidad sostenible dentro de las ciudades.

Con el paso de los años ha surgido una corriente denominada Crowdsourcing, que promueve la creación de repositorios con datos de libre acceso y cuya generación está a cargo de personas voluntarias e interesadas en una temática específica. En este contexto las ciudades inteligentes y los sistemas de transporte integrados (ITS) requieren recopilar, procesar e integrar información sobre el estado de la infraestructura de la vialidad pública. Sin embargo, recopilar información que involucre conteos volumétricos de vehículos y peatones en carreteras, intersecciones o espacios públicos de forma manual resulta un trabajo tedioso, costoso y que requiere demasiado tiempo por parte de los interesados. Además, la automatización de estos procesos de conteo requiere hardware y software específicos a un precio muy elevado.

En la actualidad los smartphones cuentan con una gran capacidad para almacenar y procesar información, además disponen de conectividad a la red lo cual los convierte en los dispositivos idóneos para la tarea de reporte de datos de tráfico como conteos vehiculares, los cuales pueden ser detectados utilizando la cámara del dispositivo para su posterior envío a una API externa o repositorio de datos. Otra ventaja del uso de smartphones es la posibilidad de efectuar el procesamiento de video en el mismo dispositivo en lugar de utilizar la nube o servidores intermedios, este procesamiento se denomina de borde o "edge computing".

En este contexto la finalidad del presente estudio consiste en implementar un aplicativo móvil que haga uso de algoritmos de detección de objetos basados en redes neuronales convolucionales para detectar y contar vehículos (coches, motocicletas, trasporte pesado) y pedestres en la carretera, el resultado de estos conteos será enviado a una plataforma Web (API) que almacenará los datos junto con su ubicación de captura en forma de series de tiempo, los cuales podrán ser accedidas para su consulta de forma libre dando lugar a la creación de un sistema "crodwsourcing" de datos abiertos y colaboración distribuida.

## II. Estado del Arte

En el contexto de las redes neuronales convulsiónales existen modelos que permiten efectuar tareas de detección de objetos en imágenes estáticas o secuencia de videos o imágenes, las aplicaciones de este tipo de tecnología son

diversas, por ejemplo en [1], los autores proponen un algoritmo para la identificación de objetos en tiempo real basado en el modelo de red neuronal convolucional denominado Single Shot MultiBox Detector (SSD), la detección se efectúa sobre objetos de uso cotidiano y tiene como finalidad medir la precisión en la detección por modelo neuronal. Pero no solo las redes neuronales permiten efectuar tareas de detección de objetos, antes de las Neural Networks (NN) otros tipos de métodos de clasificación eran los más utilizados en el contexto de Machine Learning. Tomando como referencia, por ejemplo, Support Vector Machine (SVM) como se describe en [2], este método utiliza vectores de tamaño estándar para realizar la clasificación. Un ejemplo práctico de implementación de esta técnica se efectúa en [3], donde los autores utilizan SVM en lugar de redes neuronales convolucionales para tareas de detección y clasificación de patrones a partir de un conjunto de imágenes de sonda en un proceso de barrido lateral utilizado en tareas de contramedida automática de minas.

El número de trabajos que presentan aplicaciones en las cuales los resultados de conteos vehiculares formen parte de una iniciativa de Crowdsourcing para la colaboración libre de datos es reducida, la mayoría de los estudios abarcan otro de tipo de aplicaciones las cuales van desde la recolección de datos de sensores en teléfonos móviles para generar información útil en el análisis de la actividad humana [4]. Otros autores [5], proponen una novedosa aplicación de crowdsourcing mediante la cual la comunidad puede colaborar con información para encontrar a niños reportados como desaparecidos, se emplea una plataforma de datos abiertos en línea. Otro trabajo de interés es propuesto por Muhammad y Zaka [6], los cuales describen un marco de aplicación para un sistema de notificación de eventos de emergencia, usando el crowdsourcing en la detección participativa para aprovechar múltiples usuarios de teléfonos inteligentes e informar colectivamente eventos de emergencia.

No existen alternativas comerciales que utilicen dispositivos móviles para la identificación y conteo de vehículos en carreteras, lo cual tiene sentido desde la perspectiva comercial y financiera de las compañías, puesto la mayoría de ellas busca vender una solución especializada que involucra un hardware y software propietario, el cual debe ser configurado en un sitio determinado para la captura de datos. En cuanto a soluciones de código libre la mayoría de trabajos son estudios que ofrecen alternativas para la ejecución de conteos vehiculares utilizando técnicas de procesamiento de imágenes tradicionales o redes neuronales, las alternativas disponibles en su mayoría se encuentran disponibles mediante repositorios abiertos y de libre acceso, entre los trabajos más destacados tenemos: la implementación efectuado por autores como Chen [7] los cuales utilizan SVM para la detección de vehículos en la carretera en el contexto de un Sistema de transporte integrado (ITS), en este trabajo los autores presentan los diferentes pasos de un sistema de conteo y clasificación de vehículos a través de SVM incluyendo el procesamiento, extracción de background en el fondo de las imágenes a través de modelos Gaussianos mixtos, la técnica reporta más del 99% de precisión. Sin embargo, los videos sobre los que se aplica el algoritmo corresponden al mismo escenario en todos los casos, esto implica que se debe realizar un entrenamiento específico para cada ubicación con la finalidad de obtener la máxima precisión.

El estudio realizado por Peña y Nuño [8] propone un sistema para el conteo, detección y seguimiento de vehículos, en este trabajo los autores implementan un algoritmo para detectar la Región de interés (ROI) que involucra el proceso de agrupamiento, seguimiento, clasificación por frame y el conteo vehicular. El hardware de prueba fue una PC con procesador Intel Centrino Duo (T7250@2.00GHz), 4 GB de RAM DDR2 y sistema operativo Windows Vista 32-Bits. Esta configuración corresponde a una computadora de baja capacidad. Otro trabajo relacionado es propuesto por Myung-Cheol Roh en [9], el cual efectúa la implementación del algoritmo Fast R-CNN con una mejora en la etapa de detección, la cual se argumenta tiene un desempeño pobre en la detección de objetos pequeños. El estudio demuestra que los resultados obtenidos con la modificación aportan mejoras en la detección de personas y vehículos al reducir la detección de falsos positivos (FP).

Los autores citados en la mayoría de los casos utilizan los modelos de detección en computadoras avanzadas que cuentan con GPUs, la razón principal es que los modelos como Faster R-CNN, R-FCN o GoogleLenet demandan un alto procesamiento que no puede ser satisfecho por un dispositivo embebido o teléfono móvil. En este contexto, el modelo más adecuado para el despliegue con dispositivos integrados y teléfonos móviles es SSD-Mobilenet, que está estrictamente diseñado para dispositivos con bajas capacidades de procesamiento [10].

## III. Objetivos y Metodología

### Objetivo general

Implementar un conjunto de herramientas que combinen una iniciativa de Crowdsourcing con las ventajas y capacidad de procesamiento de los dispositivos móviles Android actuales para desplegar una red neural convolucional que permita procesar y generar datos de conteos de vehículos y peatones en la carreta

### Objetivos Específicos

1. Desarrollar un aplicativo móvil para dispositivos Android que permita la captura, procesamiento y detección de vehículos en secuencias de video en tiempo real.

2. Implementar un aplicativo Web que permita almacenar y visualizar datos de conteos vehiculares.

3. Generar la información necesaria para dar solución a la problemática de la congestión de tráfico, uso inadecuado de espacios públicos para bicicletas, peatones y la falta de opciones de transporte público urbano en una ciudad intermedia.

### Metodología

La metodología considerada toma como referencia a SCRUM, la cual está orientada al trabajo en equipos, se ha

considerado modificar algunos procesos definidos por este Framework para adaptarlo a una metodología unipersonal. En los referentes a los roles SCRUM implementa tres tipos: el Product owner, Development team y Scrum master, los cuales asumen las responsabilidades de la creación, desarrollo e implementación de requisitos e historias de usuario. Para la construcción del sistema se considera la creación y manejo de los incrementos mediante el uso del producto backlog, la creación de los objetivos de Sprint y la generación del sprint backlog.

## IV. Contribución

### Topología general del sistema (Sprint 1)

La arquitectura general propuesta para el sistema de conteo vehicular se describe en la Figura 1.

El sistema consta de tres etapas: en la primera fase el usuario mediante un teléfono móvil efectúa la captura de imágenes, la cuales son pre-procesadas por el sub-sistema de detección y conteo. Es en esta parte del sistema donde se implementa el modelo de red neuronal encargado de llevar a cabo la detección y conteo de vehículos, posteriormente los conteos se envían al servicio Web para su almacenamiento y consulta.

En la fase 2 del sistema se implementa la lógica de 2backend, en esta fase se optó por implementar una lógica de micro servicios basada en serverless. Se consideró este tipo de tecnología puesto que hoy en día la informática sin servidor ofrece una serie de ventajas sobre la infraestructura tradicional basada en la nube o centrada en el servidor, entre las que destacan una mayor escalabilidad, más flexibilidad y un tiempo de lanzamiento más rápido, todo a un costo reducido.

Para el desarrollo de la aplicación móvil se consideró el uso de Android Studio en conjunto con la librería Tensorflow Lite, la cual posee soporte para la implementación de redes neuronales convoluciones en dispositivos móviles, para el manejo de la logica de backend en la API se consideró utilizar las funciones lambda de AWS en conjunto con DynamoDB como base de datos y finalmente para la creación del Frontend de la aplicación web de consulta de datos se empleó React JS.

### Topología y descripción de la etapa de detección y conteo de objetos (Sprint1)

Esta sección describe la topología del sub-sistema de detección, clasificación y conteo de vehículos (ver Figura 2).

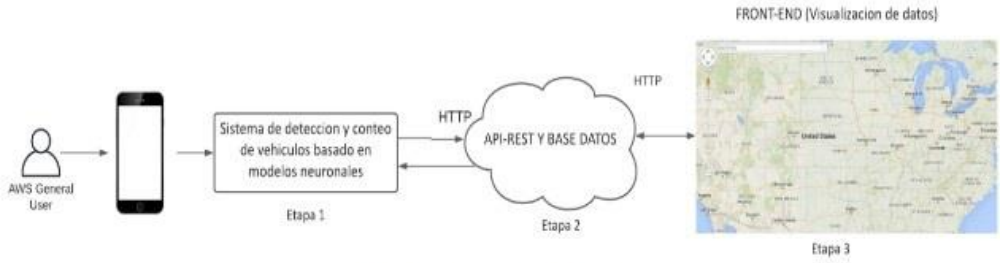



Figura 1. Topología general del Sistema

Esta etapa consta de 4 fases, en una primera instancia el usuario efectúa la captura imágenes a las cuales se aplica un preprocesamiento que consiste en la redimensión y conversión de datos de la imagen a un formato adecuado al modelo neuronal utilizado, en nuestro caso como se optó por utilizar el modelo de red neuronal denominado SSD Mobilenet.

Una vez los datos ingresan al modelo neuronal, el mismo se encarga de efectuar la detección y clasificación de los objetos de interés devolviendo los datos de detección en un array con etiquetas las cuales identifican mediante un número el objeto de interés. Esta información es procesada y se efectúa el conteo del número de incidencias por frame detectado en la escena lo cual representa el número de vehículos, personas, camiones o motos/bicicletas en el instante de tiempo de la captura de la imagen, dichos conteos se envían al servicio Web para su almacenamiento y posterior consulta.

### Topología y descripción del sub-sistema API-RESTful para el manejo de peticiones (Sprint 2).

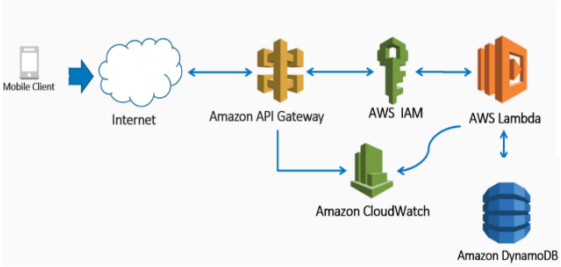



Figura 3. Topología y configuración de la API REST para el manejo de peticiones.

Para el desarrollo del backend se optó por utilizar los servicios de Amazon Web services, AWS Serverless Computing Architecture es una forma de crear y ejecutar servicios y aplicaciones sin tener que administrar la infraestructura. La aplicación de los usuarios aún se ejecuta en servidores, pero los servidores son administrados por AWS. Es un método que proporciona soporte para servicios de back-end. A los usuarios se les cobra en función de un cálculo establecido, y no se cobra nada más solo por los servicios utilizados. Aunque se

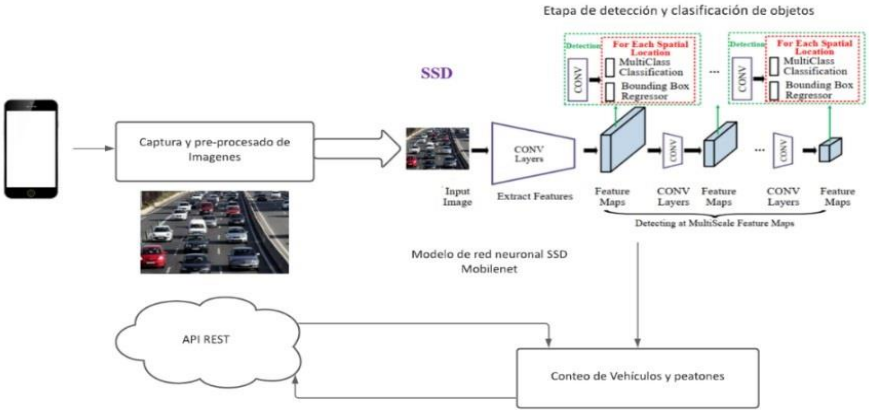



Figura 2. Topología del subsistema de captura, detección y conteo de vehículos en Smartphones

denominan servidores sin "servidor", se siguen utilizando servidores físicos, pero los desarrolladores no tienen que conocerlos. En la Figura 3 se describe la topología del sistema implementado para la creación de la API encargada del manejo de la lógica de peticiones y almacenamiento de datos.

## TOPOLOGÍA Y DESCRIPCIÓN DE LA APLICACIÓN WEB PARA LA VISUALIZACIÓN Y DESCARGA DE RESULTADOS (SPRINT 3).

Para la consulta y visualización de los datos de conteos se hizo uso del Framework React JS en conjunto con la librería "google-maps-loader" [11], la cual permite renderizar componentes de React en el mapa provisto por Google, la librería es completamente isomórfica y puede renderizarse del lado del cliente o servidor. Además, puede representar componentes de mapas en el navegador incluso si la API de Google Maps no está cargada.

El flujo de consulta de datos y visualización de conteos se describe en la Figura 4, en donde se aprecia que una petición debe ser procesada por la API de AWS y enviada a la función Lambda encargada de manejar las peticiones GET, en esta función se efectúa el proceso de conexión y consulta de datos con la base de datos DynamoDB.

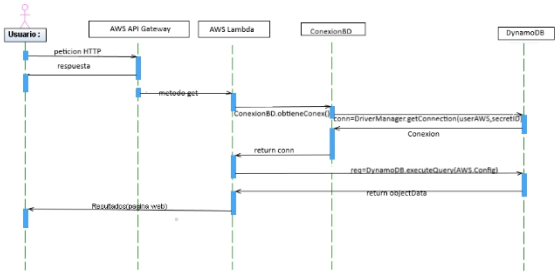



Figura 4. Diagrama de secuencia para la visualización de datos

## V. RESULTADOS

### Resultado 1.

El presente trabajo tiene como finalidad desarrollar un sistema de generación de datos abiertos que involucre la participación de la comunidad e interesados en el ámbito de la movilidad, lo cual implica que el sistema se adecue a las necesidades de los usuarios resultando intuitivo y fácil de utilizar al momento de generar y reportar datos. Considerando esta premisa se ha efectuado un conjunto de pruebas de usabilidad que consideran las siguientes métricas para el análisis [12].

**Exactitud:** Considera la cantidad errores generados por los usuarios o la aplicación durante la etapa de prueba y considera si la ampliación se recuperó de los mismos.

**Tiempo** necesario para completar las pruebas.

**Recuerdo:** establece en porcentaje que el usuario recuerda acerca del uso de la aplicación después de un periodo de haberla utilizado.

**Respuesta emocional**: El estado de ánimo de los usuarios al culminar las pruebas.

Para extraer el valor de la métrica antes descrita se efectuó un proceso de pruebas con 20 voluntarios, los cuales utilizaron la aplicación y reportaron resultados, con esta información se determinó las métricas descritas en al Tabla 1.

Tabla 1. Métricas de usabilidad del sistema de detección y conteo vehicular.

| Exactitud app móvil (% Errores) | Exactitud app web (% Errores) | Tiempo Promedio | Recuerdo | % satisfacción global | Tipo de dispositivo | Apoyo a la iniciativa |
|---|---|---|---|---|---|---|
| 10% | 5% | 40 seg (app movil) 3.5min (app web) | 90% | 95% | 60% gama alta<br>20% gama baja<br>20% gama media | 95% |

Se obtuvieron resultados favorables en el conjunto de pruebas realizadas, tan solo dos de cada 20 usuarios presentaron problemas al iniciar a la aplicación en su teléfono, los errores estuvieron relacionados al tipo de dispositivo utilizado en estos casos los dispositivos empleados fueron teléfonos de gama baja lo cual afecto la

velocidad del procesamiento, detección y conteo de vehículos en los dispositivos. Otro aspecto importante que se analizó fue el tiempo promedio que le toma a un usuario ejecutar el ciclo de conteo y envio de datos desde la aplicación a la API, para las pruebas realizadas no se superó los 40 seg en promedio, lo cual es un claro indicador de que el aplicativo resulta intuitivo y fácil de utilizar.

### Resultado 2: Análisis de precisión en la detección y conteo de vehículos

Para evaluar la precisión del proceso de conteo de vehículos se emplearon las métricas precisión (PRE), recall (REC) y F1 score (F1) las cuales se definen con detalle en [7] y consideran la cantidad falsos positivos, verdaderos positivos y falsos negativos involucrados en pruebas donde se tiene un conjunto de datos de referencia y un conjunto de datos de prueba. Estas métricas se definen en las ecuaciones 1-3, donde TP representa la cantidad de verdaderos positivos, FP los falsos positivos y FN representa a los falsos negativos.

$$PRE = \frac{TP}{TP + FP} \quad (1)$$

$$REC = \frac{TP}{TP + FN} \quad (2)$$

$$F1 = \frac{2 * REC * PRE}{REC + PRE} \quad (3)$$

Para efectuar las pruebas se generó un conjunto de datos de referencia (ground truth) para lo cual se efectuó una grabación de video de un escenario de prueba con vehículos y peatones de 15min de duración, luego se procedió a efectuar el conteo manual de los vehículos y peatones en la secuencia de video, con ello se obtuvo la cantidad real de vehículos y peatones en la escena, los resultados de este proceso se resumen en la Figura 5.

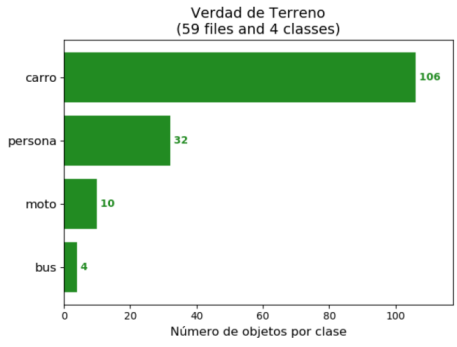



Figura 5. Conteos reales en la escena de prueba.

Posterior a este paso se procesó la secuencia de video en un teléfono móvil de gama media (xiaomi note 9) utilizando la aplicación con el algoritmo de detección y conteo vehicular, lo cual generó un conjunto de datos con las detecciones del algoritmo. Con esta información se hallaron los TP, FP y FN generados por el algoritmo de detección y se los empleo en el cálculo de las metricas PRE, REC y F1 score. Los resultados se presentan en la Figura 6.

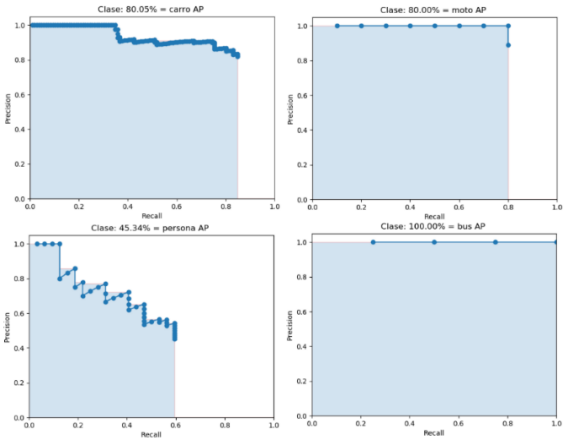



Figura 6. Métricas PRE, REC y AP del sistema.

### Resultado 3: Presentación de resultados

En esta sección se presentan los resultados del Sistema de conteo vehicular bajo condiciones de prueba en un escenario real, la Figura 7 presenta la interfaz de la aplicación móvil en sus distintas etapas de trabajo: proceso de detección, conteo y envió de datos a la API.

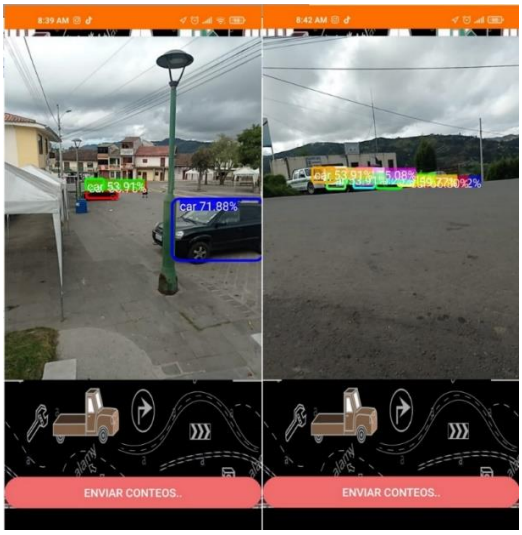



Figura 7. Interfaz de detección y envío de datos de la aplicación.

La interfaz del aplicativo es simple y el usuario final únicamente debe enfocar los vehículos de la carretera mediante la cámara del teléfono y presionar el botón enviar, el proceso de detección, conteo y envío de datos es automático y transparente al usuario.

Una vez los datos de conteo son enviados por el usuario, la petición es procesada por el endpoint encargado de manejar las peticiones de tipo POST y enviar los datos a la

base de datos, en la Figura 8 se presenta como ejemplo la estructura de los endpoint que procesan las peticiones POST en el servicio serverless de AWS.

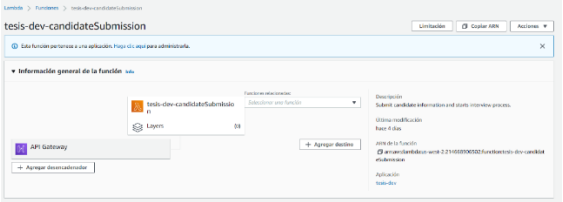



Figura 8. Endpoints serverless POST en AWS

### Resultado 4:

El aplicativo Web implementado para la consulta, visualización y descarga de los datos de conteos se presenta en las Figuras 9 y 10, se observa que los conteos almacenados se representan en un mapa mediante unos marcadores de color rojo, los cuales indican la ubicación y lugar de captura de los conteos.

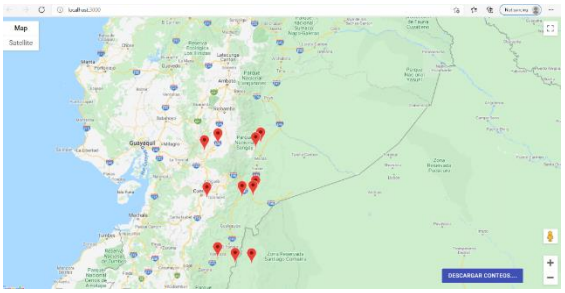



Figura 9. Aplicación Web de consulta de datos con marcadores de conteo.

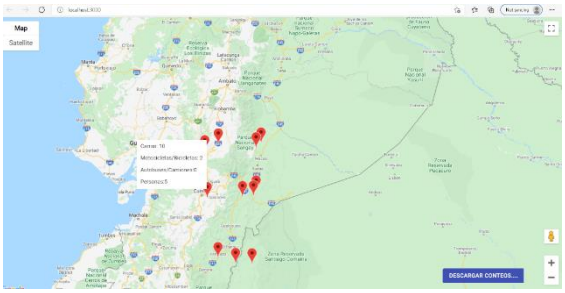



Figura 10. Modal de consulta de conteos asociado a un marcador.

Al hacer clic en el marcador se despliega un pequeño modal, en el cual se indica el número de vehículos presentes en la zona. Como se puede observar en las gráficas en la parte inferior derecha se presenta un botón el cual implementa la funcionalidad de descarga de conteos en formato csv.

## VII. Conclusiones

Este trabajo representa un aporte en lo referente al diseño de un sistema de conteos vehiculares en todas sus etapas debido a que provee los medios para la captura, generación y consulta de información en tiempo real y de libre acceso a la comunidad siendo de gran utilidad para personas y grupos interesados en contribuir con soluciones de movilidad, además se fomenta una cultura de colaboración y generación de datos de libre acceso. Uno de los principales aportes que se pudo evidenciar en este trabajo es el uso de técnicas basadas en redes neuronales aplicadas en dispositivos móviles para la ejecución de conteos vehiculares. En la actualidad la información y datos relacionados con trabajos que implementen iniciativas orientadas al conteo de vehículos es escasa o nula, existen plataformas que se orientan al conteo de otro tipo objetos implementando aplicaciones basadas en computadoras de alto rendimiento y en pocos casos teléfonos inteligentes. Por último, es necesario aclarar que, si bien una inactiva de crowdsourcing incluye e incentiva a la comunidad a contribuir de forma masiva, para garantizar su éxito esta debe ser divulgada a la sociedad.

En lo referente al aplicativo Web en futuras implementaciones se podría incorporar una funcionalidad que permita al usuario segregar los datos de los conteos por zonas de interés y descargar la información de dicha área, actualmente se permite la descarga de todos los datos de conteos disponibles en la plataforma.

Si bien no se encuentra en el alcance del presente TFM, resulta de gran interés el comparar el rendimiento y las métricas de detección del modelo SSD-Mobilenet utilizado con otro tipo de modelos neuronales, existen alternativas a SSD-Mobilent como R-CNN, GoogleLenet o R-FCN que podrían generar resultados superiores al modelo utilizado en este trabajo.

## Referencias

[1] Q. Shuai and X. Wu, “Object detection system based on SSD algorithm,” in *2020 International Conference on Culture-oriented Science Technology (ICCST)*, 2020, pp. 141–144.

[2] C. Cortes and V. Vapnik, “Support-Vector Networks,” in *Machine Learning*, 1995, pp. 273–297.

[3] N. Ma and C. S. Chia, “False Alarm Reduction by LS-SVM for Manmade Object Detection from Sidescan Sonar Images,” in *OCEANS 2007 - Europe*, 2007, pp. 1–6.

[4] N. Mairittha and S. Inoue, “Crowdsourcing System Management for Activity Data with Mobile Sensors,” in *2019 Joint 8th International Conference on Informatics, Electronics Vision (ICIEV) and 2019 3rd International Conference on Imaging, Vision Pattern Recognition (icIVPR)*, 2019, pp. 85–90.

[5] H. El Alaoui El Abdallaoui, A. El Fazziki, A. Sadiq, E. F. Zohra, and M. Sadgal, “A web based crowdsourcing framework: Lost child case,” in *2016 4th IEEE International Colloquium on Information Science and Technology (CiSt)*,

2016, pp. 263–268.

[6] M. T. Jilani, M. Z. Ur Rehman, and M. A. Abbas, “An Application Framework of Crowdsourcing based Emergency Events Reporting in Smart Cities,” in *2019 International Symposium on Networks, Computers and Communications (ISNCC)*, 2019, pp. 1–5.

[7] Z. Chen, T. Ellis, and S. A. Velastin, “Vehicle detection, tracking and classification in urban traffic,” in *2012 15th International IEEE Conference on Intelligent Transportation Systems*, 2012, pp. 951–956.

[8] R. H. Peña-González and M. A. Nuño-Maganda, “Computer vision based real-time vehicle tracking and classification system,” in *2014 IEEE 57th International Midwest Symposium on Circuits and Systems (MWSCAS)*, 2014, pp. 679–682.

[9] M.-C. Roh and J. Lee, “Refining faster-RCNN for accurate object detection,” in *2017 Fifteenth IAPR International Conference on Machine Vision Applications (MVA)*, 2017, pp. 514–517.

[10] N. A. OTHMAN and I. AYDIN, “A New Deep Learning Application Based on Movidius NCS for Embedded Object Detection and Recognition,” in *2018 2nd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)*, 2018, pp. 1–5.

[11] NPM, “google-maps-loader,” *1*, 2021. [Online]. Available: https://www.npmjs.com/package/google-maps.

[12] Wikipedia, “Prueba de usabilidad,” *1*, 2020. [Online]. Available: https://es.wikipedia.org/wiki/Prueba_de_usabilidad. [Accessed: 07-Jun-2021].