SISTEMA DE CONTEO DE PERSONAS POR MEDIO DE INTELIGENCIA ARTIFICIAL

CRISTIAN DE JESÚS BASTIDAS PEDROZO

JORGE ALEJANDRO GARZÓN TORRES

YULIETH VANESA RINCON SAAVEDRA

UNIVERSIDAD INDUSTRIAL DE SANTANDER

FACULTAD DE INGENIERÍAS FISICOMECÁNICAS

ESCUELA DE INGENIERÍAS

ELÉCTRICA, ELECTRÓNICA Y DE TELECOMUNICACIONES

BUCARAMANGA

2023

SISTEMA DE CONTEO DE PERSONAS POR MEDIO DE INTELIGENCIA ARTIFICIAL

CRISTIAN DE JESÚS BASTIDAS PEDROZO

JORGE ALEJANDRO GARZÓN TORRES

YULIETH VANESA RINCON SAAVEDRA

Trabajo de Grado para optar al título de

Ingeniero Electrónico

Director

Jaime Guillermo Barreo Pérez

Magíster en potencia eléctrica

UNIVERSIDAD INDUSTRIAL DE SANTANDER

FACULTAD DE INGENIERÍAS FISICOMECÁNICAS

ESCUELA DE INGENIERÍAS

ELÉCTRICA, ELECTRÓNICA Y DE TELECOMUNICACIONES

BUCARAMANGA

2023

Dedicatoria

Dedico este trabajo principalmente a Dios dando gracias por todas las bendiciones que me brindado para llegar a este momento de mi vida.

A mi madre Cecilia, mis hermanos Lorena y Esteban, a mi hijo Juan David y mi novia quienes con su apoyo me han brindado la motivación para lograr mi desarrollo profesional siendo la base central de mi vida.

A mis familiares más cercanos quienes han sido fundamental en el apoyo constante sin importar la distancia.

A mis amigos y compañeros que me deja el paso por mi Alma mater.

A mi abuela y mi segunda madre quienes desde el cielo son participes de este logro.

Finalmente, a la Universidad Industrial de Santander por brindarme innumerables experiencias y aprendizajes. Mención especial para el personal de Bienestar Universitario, específicamente asociado al servicio de comedores, familia que me abrió las puertas y quienes merecen toda mi admiración, respeto y agradecimiento.

Cristian de Jesús Bastidas Pedrozo

Dedico este proyecto de grado a Dios por darme sabiduría y paciencia en los momentos más complejos.

A mi tía Fabiola Saavedra Garzón, muchas gracias por el apoyo, por sus consejos, enseñanzas, dedicación y amor que me brindo a lo largo de este proceso.

A David mi novio, por estar presente durante los momentos más difíciles, por ser ese apoyo incondicional y creer en mí. Gracias por motivarme a ser mejor persona, por todo el amor y paciencia durante esta etapa.

Finalmente, gracias a mis hermanas, mi familia y mis amigos, por siempre estar presente y apoyarme de una u otra forma, gracias por todos los momentos compartidos que siempre llevare en mi corazón.

Yulieth Vanesa Rincon Saavedra

Tabla de Contenido

Lista de Figuras

Lista de Tablas

Resumen

Título: Sistema de conteo de personas por medio de inteligencia artificial [*]

Autor: Cristian de Jesús Bastidas Pedrozo, Jorge Alejandro Garzón y Yulieth Vanesa Rincon Saavedra[**]

Palabras Clave: Red, Neuronal, CNN, YOLO, Personas, Conteo, MAIX II, Python, Pytorch, Raspberry, Maixhub,

Descripción: Este trabajo se centra en la aplicación de redes neuronales para solucionar o facilitar problemas cotidianos, como es el caso del conteo de personas. Se exploraron algunas técnicas necesarias en el campo de la inteligencia artificial que abordan algoritmos alternativos con los que se pueda construir dicho sistema. En este trabajo se utilizó la tarjeta Raspberry Pi 4 para implementar el sistema de conteo de personas. Se aprovechan los avances en visión por computadora, procesamiento de imágenes y reconocimiento de patrones para lograr esta tarea. El objetivo consistió en desarrollar un sistema capaz de detectar y mantener una información cuantitativa de las personas que entran y salen de un establecimiento.

[*] Trabajo de Grado

[**] Facultad de Ingenierías Fisicomecánicas. Escuela de Ingeniera Eléctrica, Electrónico y Telecomunicaciones (E3T). Mag. Jaime Guillermo Barrero Pérez.

Abstract

Title: People counting system through artificial intelligence [*]

Author(s): Cristian de Jesús Bastidas Pedrozo, Jorge Alejandro Garzón and Yulieth Vanesa Rincon Saavedra [**]

Key Words: Neural Network, CNN, YOLO, People, Counting, MAIX II, Python, Pytorch, Raspberry, Maixhub,

Description: This work focuses on the application of neural networks to solve or facilitate everyday problems, such as people counting. Some techniques needed in the field of artificial intelligence that address alternative algorithms with which such a system can be built were explored. In this work, the Raspberry Pi 4 board was used to implement the people counting system. It takes advantage of advances in computer vision, image processing and pattern recognition to accomplish this task. The objective was to develop a system capable of detecting and maintaining quantitative information about people entering and leaving a facility.

[*] Degree Work

[**]Faculty of Physicmechanics Engineering. School of Electrical, Electronic and Telecommunications Engineering (E3T). Mag. Jaime Guillermo Barrero Pérez.

Introducción

La inteligencia artificial para la resolución de problemas es cada día un tema que está más presente en nuestra sociedad, Entre 2016 y 2022 se esperaba que el mercado de sistemas de conteo de personas creciera gradualmente (*Mercado de sistemas de conteo de personas Insights*, s. f.); teniendo en cuenta la pandemia que afrontamos en el año 2019, dicho crecimiento se elevó significativamente por la necesidad de sistemas de conteo de personas en establecimientos públicos, transporte, video vigilancia entre otros, para tener control de aforo en estos establecimientos. *Mercado de sistemas de conteo de personas por tecnología, aplicación y geografía.* [En linea] Disponible en:

https://www.prnewswire.com/news-releases/people-counting-system-market-by-technology-application-and-geography---global-forecast-to-2022-300396923.html

Según estudios de negocios y mercadeo (Sbt, 2018), un contador de personas es una herramienta que puede optimizar las ventas, mejorar la conversión y el tráfico en establecimientos públicos, también se puede hacer un estudio de los datos obtenidos para la gestión de los horarios de los empleados o si es conveniente expandir operaciones comerciales o reducir costos, en pocas palabras un sistema de conteo de personas es una herramienta que nos puede servir en diferentes ámbitos sociales (Sbt, 2018).

En revisión de la literatura, a lo largo de los años se han propuesto diversas técnicas para el conteo de personas, los primeros contadores de personas utilizaron sensores infrarrojos CapsNet-based(V. H. Roldão et al, 2020), posteriormente se usaron dispositivos con sensores de imagen térmica (V. H. Roldão et al, 2020); por último en la actualidad tenemos el aprendizaje automático junto el procesamiento de imágenes, en (Huazhong et al, 2010) basado en

procesamiento de imágenes por vista aérea donde principalmente distribuyen el conteo de personas en extracción en primer plano, detección, seguimiento y conteo, en (Ahmed et al, 2019) se tiene un sistema basado en la cabeza y el hombro y SVM lineal para entrenar al clasificador, en el seguimiento se usa el filtro de partículas con histograma de color (*Support Vector Machine (SVM)*, MathWorks).

A partir de esta información y realizado el estudio oportuno para obtener un diseño del sistema que permite el conteo en tiempo real de la cantidad de personas que entran y salen de un establecimiento privado, el sistema se basa en tres módulos fundamentales que permiten la detección y seguimiento, clasificación y conteo. Se elaboró una base de datos con imágenes extraídas de una cámara de seguridad de un local comercial, posteriormente se abordaron diferentes métodos de redes de detección, en donde el primero consiste en el entrenamiento con la herramienta de MaixHub y otras dos alternativas con diferentes versiones de YOLO, para finalmente implementar dicho sistema.

1. Objetivos

1.1. Objetivo general

Desarrollar un sistema autónomo usando inteligencia artificial, para determinar cuántas personas hay en un recinto

1.2. Objetivos específicos

- Seleccionar apropiadamente el hardware para realizar la respectiva implementación, teniendo en cuenta aspectos relevantes como precio, especificaciones técnicas, tamaño y consumo de potencia.
- Implementar un algoritmo de inteligencia artificial a partir de una revisión de técnicas en la literatura científica, que permita realizar el conteo de personas, en un ambiente definido y con una alta confiabilidad.
- Implementar el modelo obtenido por el algoritmo, en un sistema embebido, de manera que se pueda valorar su desempeño.

2. Metodología

En esta sección se presenta de manera amplia los conceptos que fueron necesarios comprender para exponer alternativas que brinden respuesta a lo planteado, seguido se expone la investigación realizada para la selección del sistema embebido.

2.1. Aprendizaje automático

La inteligencia artificial (artificial intelligence), el aprendizaje automático (Machinne learning) y el aprendizaje profundo (Deep learning), son 3 terminologías que se usan hoy en día para representar sistemas inteligentes. Aprendizaje automático es la ciencia que permite darle la habilidad a los sistemas de cómputo para que puedan desarrollar una tarea sin tener que darle el paso a paso de esta, un algoritmo de machine learning tiene 3 componentes esenciales: una tarea, una experiencia o entrenamiento y un rendimiento (Sarker, 2022).

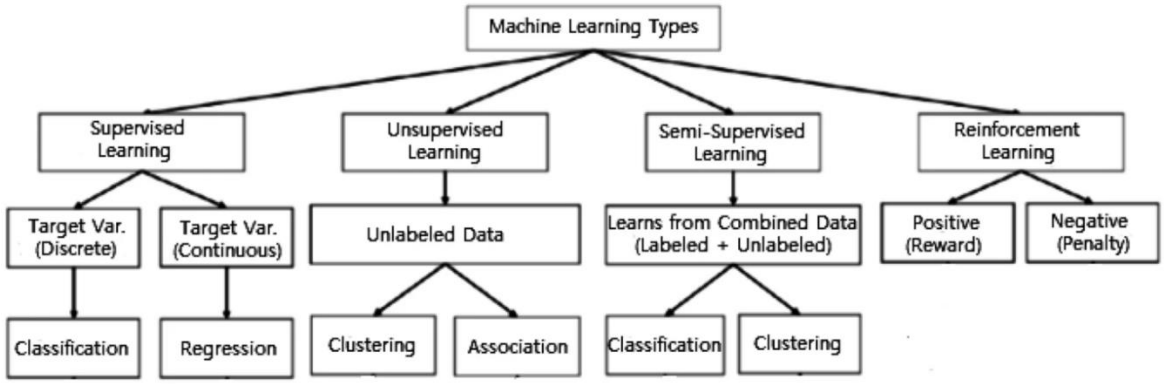
El machinne learning se divide en 4 categorías: aprendizaje supervisado, aprendizaje no supervisado, aprendizaje semisupervisado y aprendizaje reforzado como se muestra en la Figura.1. A continuación se dará una breve explicación de cada una de estas categorías.

- Supervisado: el aprendizaje supervisado se lleva a cabo identificando ciertos objetos, que se logra a partir de un conjunto de insumos; los algoritmos o modelos aprenden de los datos etiquetados (enfoque basado en tareas).
- No supervisado: analiza un conjunto de datos (dataset) sin necesidad de interferencia humana, los algoritmos o modelos aprenden de datos no etiquetados (enfoque basado en datos).

- Semi-supervisado: este tipo de aprendizaje opera con datos supervisados y no supervisados, su objetivo principal es proporcionar un mejor resultado para la predicción.
- Reforzamiento: aprendizaje basado en la recompensa o la penalización, este permite que los agentes de software y máquinas evalúen automáticamente el comportamiento óptimo en un entorno, es decir un enfoque basado en el entorno (Sarker, 2021).

Figura 1

Estructura machinne learning.



Tomado de: Sarker, I.H. Machine Learning: Algorithms, Real-World Applications and Research Directions. SN COMPUT. SCI. 2, 160 (2021). https://doi-org.bibliotecavirtual.uis.edu.co/10.1007/s42979-021-00592-x

2.2. *Deep Learning* y Redes Neuronales.

El aprendizaje profundo o deep learning es básicamente una red neuronal con tres o más capas, estas redes neuronales se inspiran en el funcionamiento del cerebro humano y su sistema

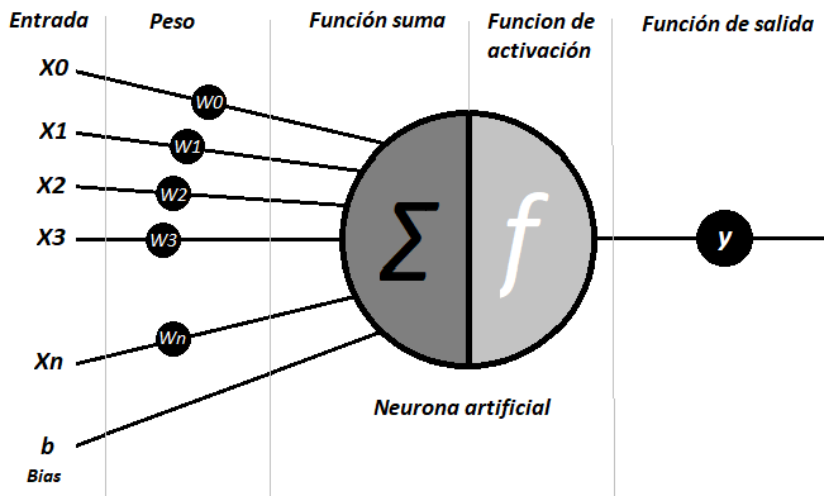
nervioso. Esta red forma parte de nuestra vida cotidiana, por ejemplo, en servicios financieros, controles de TV habilitados por voz, automóviles autónomos, entre otros.

La tecnología de deep learning se originó con base en una red neuronal artificial (ANN), una red neuronal típica se compone de capas de nodos, que contienen una capa de entrada, una o más capas ocultas y una capa de salida. Cada nodo o neurona se conecta a otro y tiene un peso y un umbral asociado; si la salida de cualquier nodo está por encima del valor umbral, dicho nodo se activa (¿Qué son las redes neuronales? | IBM, s. f.). La Figura 2. muestra la representación esquemática de un modelo matemático de una neurona artificial, destacando la entrada Xn, peso (w), sesgo (b), función de suma (∑), función de activación (f) y señal. Las redes neuronales artificiales principalmente están compuestas por este nuevo concepto de neurona, donde se puede hacer el símil con una función matemática de regresión lineal como la mostrada en la ecuación (1).

$$F_a(Y_1) = F_a(W1 \cdot X_1 + W2 \cdot X_2 + b) \quad (1)$$

Figura 2

Representación esquemática del modelo matemático de una neurona artificial.



Nota: **Tomado de:** Elaboración propia

Como se ilustra en la Figura 2, las redes neuronales constan de varias capas de nodos interconectados, las capas de entrada y salida se denominan capas visibles, la capa de entrada recibe los datos para el procesamiento y la capa de salida realiza la predicción y clasificación final; dependiendo del problema se cuenta con una función de activación la cual convierte la operación matemática en una función no lineal que permite resolver problemas con una dispersión particular.

El uso más común de las redes neuronales está relacionado con problemas de reconocimiento, clasificación, predicción y descripción de objetos. De igual modo, existen diferentes tipos de redes neuronales para abordar problemas o dataset, dos ejemplos de estas redes son: redes neuronales convolucionales y redes neuronales recurrentes.

2.2.1. Redes Neuronales Convolucionales

Las redes neuronales convolucionales (CNN) son un tipo de red neuronal artificial, se usan frecuentemente para la clasificación y visión por computadora, CNN están diseñadas específicamente para manejar una variedad de formas en 2D aprovechando los principios del álgebra lineal, por lo cual las CNN tiene amplia aplicación en análisis de imágenes médicas, reconocimiento visual, segmentación de imágenes y muchos más.

Las redes neuronales convolucionales tienen tres tipos principales de capas, esto no significa que solo se componga de 3 capas, por ejemplo, seguido de la capa convolucional puede haber otra capa convolucional o de agrupación haciendo así que con cada capa aumente su complejidad la CNN, realizando así una mayor extracción de las características de la imagen, descritas a continuación:

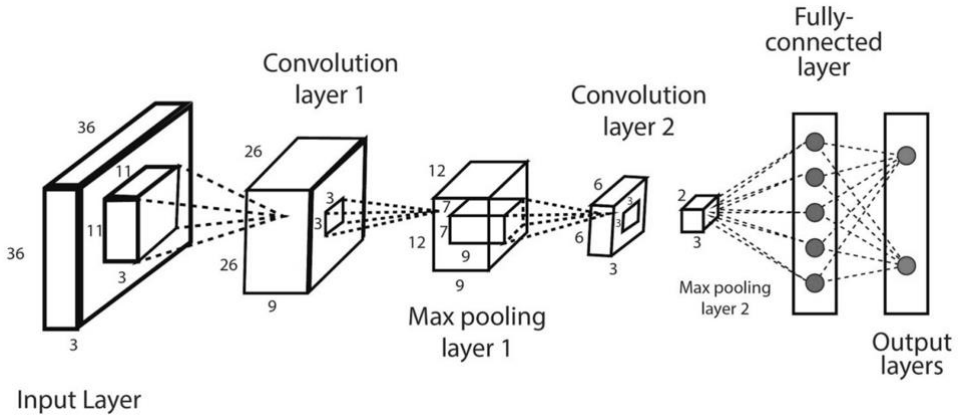
- Capa convolucional: Esta capa es el componente básico de la CNN donde se realiza la mayor parte del cálculo; consta principalmente de datos de entrada, filtros (por ejemplo, kernel) y mapas de características. Un detector de características es una matriz bidimensional (2-D) de pesos que representa una parte de una imagen. Después de cada operación de convolución, CNN aplica una transformación de unidad lineal rectificada (ReLU) al mapa de características, introduciendo así no linealidad en el modelo.
- Capa de agrupación: La agrupación de capas, también conocida como submuestreo, realiza una reducción de dimensionalidad; esto reduce la cantidad de parámetros en la entrada. De manera similar a una capa convolucional, la operación de agrupación barre un filtro sobre toda la imagen, pero la diferencia es que este filtro no tiene pesos.
- Capa completamente conectada (FC): Esta capa realiza tareas de clasificación basadas en características extraídas a través de capas anteriores y sus diferentes filtros. Mientras que las capas convolucionales y de agrupación tienden a usar funciones ReLu, las capas FC generalmente utilizan funciones de activación softmax para clasificar adecuadamente la entrada, lo que genera probabilidades entre 0 y 1.

En la Figura 3 se puede observar un ejemplo de una red neuronal convolucional CNN la cual incluye múltiples convoluciones y capas de agrupación; como resultado CNN mejora el diseño de la red neuronal artificial tradicional. *What are convolutional neural networks*? [En línea] Disponible en:

https://www.ibm.com/topics/convolutional-neural-networks

Figura 3

Ejemplo de una red neuronal convolucional (CNN) con múltiples capas de convolución y agrupación.



Nota: Tomado de: Sarker, I.H. Machine Learning: *Algorithms, Real-World Applications and Research Directions.* SN COMPUT. SCI. 2, 160 (2021). https://doi-org.bibliotecavirtual.uis.edu.co/10.1007/s42979-021-00592-x

2.2.2. Redes Neuronales recurrentes

Una red neuronal recurrente (RNN) es un tipo de red neuronal artificial que utiliza datos secuenciales o datos de series temporales. Así como las redes neuronales convolucionales (CNN), las redes neuronales recurrentes utilizan datos de entrenamiento para aprender. Se distinguen por su "memoria", ya que obtienen información de entradas anteriores para influir en la entrada y salida actuales. Por lo general se usa en problemas ordinarios o temporales tales como: traducción de idiomas, reconocimiento de voz, también se incorporan a aplicaciones populares como Siri, búsqueda por voz, entre otras. Algunas arquitecturas variantes de RNN son:

- Redes neuronales recurrentes bidireccionales (BRNN): Mientras que las redes unidireccionales solo pueden extraerse de entradas anteriores para realizar predicciones sobre el estado actual, las RNN bidireccionales extraen datos futuros para mejorar su exactitud.
- Memoria a corto-largo plazo (LSTM): esta red se diseñó con el fin de resolver una problemática común de las redes recurrentes, conocida como el problema del *Long-Term*. En secuencias largas en donde se necesite hacer una predicción o aproximación al final de la cadena de entrada, es probable que requiera de información del inicio que ya no se tiene almacenada.
- Unidades recurrentes cerradas (GRU): También intenta solucionar el problema de la memoria a corto plazo; en lugar de usar el estado de la celda usa estados ocultos, en lugar de tres puertas, tiene dos: una puerta de restablecimiento y una puerta de actualización. (*¿Qué son las redes neuronales recurrentes? | IBM*, s. f.)

2.2.3. Redes neuronales profundas preentrenadas

En el mundo de la inteligencia artificial se puede emplear una red neuronal preentrenada la cual como su nombre lo indica ya ha sido empleada para cumplir una funcionalidad especifica en otro contexto. Se pueden usar redes neuronales preentrenadas para realizar tareas tales como clasificación, extracción de características, transferencia de características, entre otras. Estas redes tienen diferentes características las cuales son importantes a la hora de elegir una de ellas para aplicar a la resolución de algún problema, las características más relevantes son: precisión, velocidad y el tamaño de la red neuronal, una red neuronal buena tiene una precisión alta y es rápida (*Redes neuronales profundas preentrenadas - MATLAB & Simulink - MathWorks España*, s. f.).

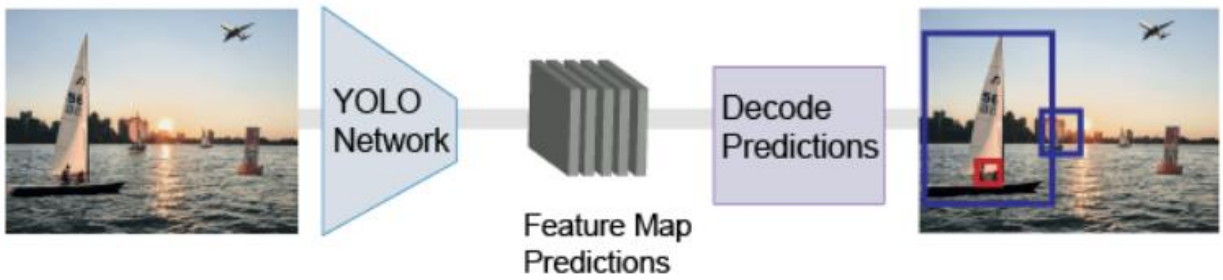
Algunos ejemplos de redes neuronales preentrenadas son:

- alexnet
- Red eficiente
- EfficientNetV2
- Darknet-19
- MNASNet
- MobileNet V2
- MóvilNet V3
- RegNet
- Resnet
- Resiguiente

Los modelos de aprendizaje profundo basados en Yolo utilizan las redes previamente entrenadas mencionadas anteriormente; Yolo se introdujo por primera vez en 2016 el cual marco un hito en la investigación de detección de objetos debido a su capacidad para detectar objetos en tiempo real con mayor precisión (Supeshala, 2021), es uno de los mejores modelos de reconocimiento de objetos el cual utiliza una red de detección de objetos de una sola etapa. El modelo ejecuta una CNN de aprendizaje profundo para producir predicciones de red, el detector de objetos decodifica las predicciones y genera cuadros delimitadores tal como se ilustra en la Figura 4.

Figura 4.

Detección de objetos con Yolo



Nota. Tomado de: *Getting Started with YOLO v2 - MATLAB & Simulink*. (s. f.). https://www.mathworks.com/help/vision/ug/getting-started-with-yolo-v2.html

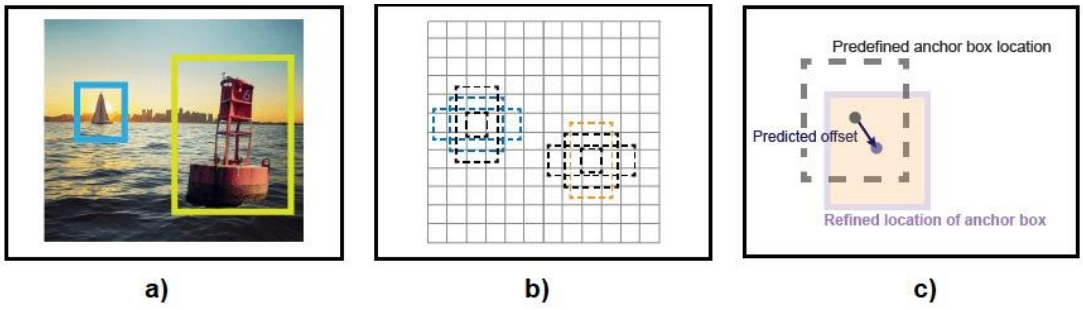
La principal característica del método de predicción de yolo son los cuadros de anclaje, los cuales se obtienen utilizando el algoritmo K-Means (con distancia euclidiana simple) para agrupar el ancho y el alto de todos los cuadros etiquetados en los datos de entrenamiento. El objetivo del agrupamiento es hacer que la relación de aspecto de cada ancla se acerque a la relación de aspecto del cuadro real, en la figura 5.b se puede observar cuadros de anclaje predefinidos (las líneas punteadas) en cada ubicación en un mapa de características y la ubicación refinada; los recuadros coincidentes con una clase están en color y se observan en la Figura 5.a (*Getting Started with YOLO v2 - MATLAB & Simulink*, s. f.).

Yolo predice estos tres atributos para cada cuadro de anclaje:

- Intersección sobre unión (IoU): predice la puntuación de objetividad de cada cuadro de anclaje y mejora la precisión de localización para esto los detectores de objetos de aprendizaje profundo aprenden compensaciones para aplicar a cada cuadro de anclaje en mosaico, refinando la posición y el tamaño del cuadro de anclaje Figura 5.c.
- Desplazamientos del cuadro de anclaje: refina la posición del cuadro de anclaje
- Probabilidad de clase: predice la etiqueta de clase asignada a cada cuadro ancla.

Figura 5

Cuadros de anclaje yolo



Nota. Tomado de: *Getting Started with YOLO v2 - MATLAB & Simulink*. (s. f.). https://www.mathworks.com/help/vision/ug/getting-started-with-yolo-v2.html

Algunos conceptos que ayudaran a comprender factores importantes a la hora de entrenar un modelo se trataran a continuación, seguidos de 3 métodos de entrenamiento de redes neuronales.

Overfitting Esto se aplica cuando una red neuronal aprende mucho con los datos de entrenamiento, pero tiene un rendimiento deficiente con los datos de validación o con los datos que nunca había visto. Siempre queremos evitar este tipo de comportamiento.

Underfitting es lo opuesto al overfitting. Esto ocurre cuando la red neuronal no aprende correctamente los datos de entrenamiento y, en general, tiene un rendimiento deficiente en todas las predicciones.

Dataset o conjunto de datos hace referencia a un conjunto de datos históricos el cual sirve de base para entrenar un algoritmo con el objetivo de que este aprenda a tomar decisiones.

Labels el etiquetado de datos en el aprendizaje automático implica identificar datos sin procesar (imágenes, archivos de texto, videos, etc.) y agregar una o más etiquetas significativas e informativas que proporcionen contexto para que los modelos de aprendizaje automático puedan aprender de ellos.

Training Set cuando tenemos un dataset (también le podemos llamar agrupación de datos) es importante dividirlo en dos o tres sets diferentes, el primero de ellos es el set de entrenamiento, este es el set que se usa para que la red neuronal aprenda patrones sobre los datos.

Validation set es necesario que este set tenga datos diferentes de los otros dos sets, esto es importante ya que queremos ver que tan buena es la red neuronal en datos que nunca ha visto.

Learning rate este hiperparametro indica que tan largo será el camino que tome el algoritmo de optimización.

Lost function la función de perdida, también conocida como función de costo, es la función que nos dice que tan buena es la red neuronal, un resultado alto indica que la red neuronal tiene un desempeño pobre y un resultado bajo indica que la red neuronal está haciendo un buen trabajo.

Época alude al número de épocas o iteraciones completas que se utilizarán durante el entrenamiento. En cada ciclo (epoch) todos los datos de entrenamiento pasan por la red neuronal para que esta aprenda sobre ellos, si existen 10 ciclos y 1000 datos, cada ciclo los 1000 datos pasaran por la red neuronal.

2.3. Sistema Embebido

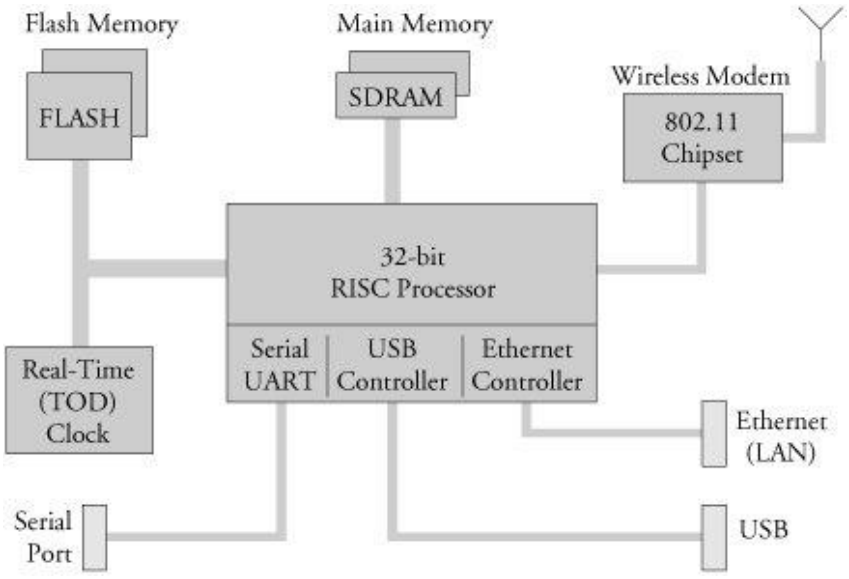
Son herramientas de computación utilizadas para ejecutar tareas de control y cuyos componentes se encuentran integrados en una placa base. En revisión de la literatura e indagando el sistema embebido que más se adaptó a los alcances del proyecto para así poder ejecutar de la manera más precisa el modelo a utilizar.

Un sistema embebido posee un hardware; un software primario el cual está diseñado bajo unas restricciones importantes como memoria, capacidades limitadas de procesamiento, consumo

de energía y un sistema operativo que posea características en tiempo real. En la Figura 6. se puede observar un ejemplo de la arquitectura de un sistema embebido de acceso inalámbrico y sus principales partes, las cuales se pueden resaltar un procesador RISC de 32 bits, una memoria flash, la memoria principal, interfaz de ethernet, puerto USB, puerto serial y un chip que contiene la implementación estándar IEEE 802.11. *Industrial Systems Engineering* [En linea] Disponible en: http://www.ieec.uned.es/investigacion/Dipseil/PAC/archivos/Informacion_de_referencia_ISE5_3_1.pdf

Figura 6

Ejemplo de un sistema embebido.



Nota. Tomado de: *Embedded Linux Primer: A Practical Real-World Approach | Section 2.2. Anatomy of an Embedded System, Hallinan, 2006, Prentice Hall*.

Algunas características que se pueden distinguir de un sistema embebido son su funcionamiento específico, su naturaleza híbridos (poseen partes analógicas y digitales), una

fuente de alimentación limitada y poseen una administración de energía efectiva. Los sistemas embebidos están orientados a minimizar los costos, maximizar la confiabilidad y por último el bajo consumo de potencia (*Maix-II-Dock(M2dock) introduction - Sipeed Wiki*, s. f.-b).

Teniendo presente las características principales de un sistema embebido, a continuación, se exponen los sistemas propuestos y cual se seleccionó para implementación del modelo:

Las alternativas de implementación planteadas son jetson nano 2 Gb (*Kit para desarrolladores Jetson Nano NVIDIA Jetson Nano*, s. f.), Maix-II-Dock (*Maix-II-Dock(M2dock) introduction - Sipeed Wiki*, s. f.) y la Raspberry Pi 4 (*DATASHEET Raspberry Pi 4 Model B*, s. f.). Se realizó un cuadro comparativo el cual corresponde a la tabla 1, en donde se detallan las características principales de los sistemas descritos anteriormente.

Tabla 1

Caracteristicas comparando Jetson nano, Raspberry Pi 4, Maix-II Dock.

	Sistemas embebidos		
Características	**Jetson nano[a]**	**Raspberry Pi 4[b]**	**Maix-II-Dock[a]**
CPU	ARM Quadcore A57 1.43 GHz	Broadcom BCM2711, quad-core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz	ARM Single-core Cortex-A7 800-1000MHz
Codificador de video	4kp30 4x1080p30 9x730p30	H.265 (4kp60 decode), H264 (1080p60 decode, 1080p30 encode)	H264 1080p@30fps H265 1080p@30fps JPGE 1080p@30fps
Memoria	2GB LPDDR4	8GB LPDDR4-3200 SDRAM	SIP64MB DDR2
Cámara	12 lanes (3x4) MIPI CSI-2	5 Mpx 1080P@60fps	2lane MIPI 1080P@60fps
Pantalla	HDMI 2.0	2 micro-HDMI® ports (up to 4kp60 supported)	8bit MCU LCD placa adaptadora LCD RGB 10in
Ethernet	gigabit Ethernet	gigabit Ethernet	10/100 Mbit/s RMII interface
SPI	Si	Si	(SPI0 SPI1)
ADC	No	No	LRADC de 6 bits 1 canal para clave

Audio	No	Si	LINEOUTP+MICIN1P/N
Precio	$ 1’050.000	$ 750.000	$ 260.000

Nota. Tomado de: Maslov (2021d), *DATASHEET Raspberry Pi 4 Model B*, (2019)

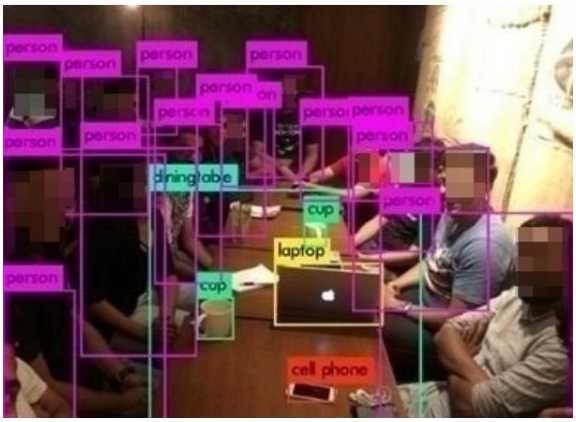
Revisando las especificaciones de las tarjetas y teniendo en cuenta la relación entre su precio y la facilidad de compra en tiendas nacionales, se eligió el sistema embebido correspondiente a la Raspberry Pi 4, esta es una opción favorable ya que cuenta con un procesador de cuatro núcleos, memoria ram de 8GB, adicionalmente incluye dos puertos USB 3.0, dos puertos USB 2.0, puertos Ethernet Gigabit, Wi-Fi 802.11ac y Bluetooth 5.0, lo que facilita la conexión de periféricos y la conectividad en red (Rus, 2019). Asimismo, cuenta con distintas referencias asociadas a la implementación, como la facilidad de ejecutar las librerías necesarias, además de eso, proporciona una mejor resolución y calidad de imagen en nuestro sistema lo cual resulta útil al momento de usar visión por computadora.

3. Implementación de las redes neuronales

Luego de realizar la respectiva investigación de los conceptos más importantes que involucran el contexto del trabajo se planteó la idea de poder desarrollar varios modelos de detección de personas por redes neuronales convolucionales Figura 7, dado que por medio de esta técnica se disminuyen los costos en la adquisición de componentes adicionales y se tiene una alternativa más precisa en la detección de personas a comparación de los sistemas tradicionales. A partir de dichos modelos, se realiza una comparativa entre las alternativas seleccionadas, teniendo en cuenta aspectos importantes como precisión, velocidad de predicción, consumo de recursos, compatibilidad con el sistema embebido y densidad de la red neuronal.

Figura 7

Modelo de detección



Nota. Tomado de: 330ohms. ¿Cómo detectar objetos con YOLO? 330ohms. (2020, noviembre 17). https://blog.330ohms.com/2020/11/17/deteccion-de-objetos-con-yolo/

3.1. Entrenamiento YoloV5

La industria del aprendizaje automático creó un algoritmo de reconocimiento de objetos en tiempo real llamado "solo miras una vez versión 5" (YOLO V5) el cual detecta objetos utilizando cuadriculas. Se crea una cuadrícula de tamaño S x S a partir de la imagen de entrada, con cada celda de la cuadrícula a cargo de la detección de elementos. Además, los cuadros delimitadores, las calificaciones de confianza y las probabilidades de clase de los objetos descubiertos se incluyen en las celdas de la cuadrícula (Madhumathi et al., 2023); El entrenamiento con Yolo V5 se divide en 3 partes fundamentales.

3.1.1. Recopilación de Datos

Se construyó la base de datos con las imágenes que se usarían para llevar a cabo el entrenamiento y la validación del modelo. Los datos fueron extraídos de una cámara de seguridad de un establecimiento comercial; donde se obtuvieron recorte de personas para posteriormente redimensionar por procesamiento de imágenes a un tamaño de 224x224, estas dimensiones fueron seleccionadas basados en la literatura (Martínez Peiró, 2021). El dataset obtenido se puede visualizar en el repositorio del *GifHub* https://github.com/gitale9/contadorDePersonasIA#base-de-datos

Dicho dataset consta de 1068 imágenes las cuales se dividen entre entrenamiento del modelo y validación. Las imágenes se encuentran debidamente etiquetadas en formato yolo, el cual corresponde a un archivo en formato .txt, dicho formato establece la clase y las coordenadas de los cuadros delimitadores con la siguiente estructura (c, xn, yn, wn, hn) (Salinas, 2022) donde:

- c: es el número de la clase, en este proyecto hay 1 clase, por lo que 0 será persona.
- xn: centro del cuadro delimitador normalizado en la dirección x.
- yn: centro del cuadro delimitador normalizado en la dirección y.

- wn: ancho normalizado del cuadro delimitador (x).
- hn: alto normalizado del cuadro delimitador (y).

3.1.2. Entrenamiento del Modelo

Respecto al entrenamiento, se llevó a cabo con la red de YoloV5 la cual es un sistema de código abierto para la detección de objetos en tiempo real pre-entrenado con el dataset COCO, el cual hace uso de una única red neuronal convolucional CNN para detectar objetos en imágenes. Los parámetros de entrenamiento se pueden detallar en la tabla 2. Esto con el fin de que dicha red de detección lleve a cabo la identificación de personas mediante la extracción de características y por medio de la observación en la entrada del establecimiento poder mantener un conteo en tiempo real del aforo mediante el procesamiento de imágenes.

Para llevar a cabo el entrenamiento del modelo se decidió detener la transferencia de aprendizaje en la época 100 dado que a partir de la época 80 se evidenció que el modelo no presentaba cambios significativos en el rendimiento de la red de detección. Esto es importante ya que al momento del entrenamiento se puede experimentar el fenómeno de sobre ajuste.

Tabla 2

Parámetros de entrenamiento

Parámetro	Valor
Dimensión de Imagen	224 x 224
Época	100
Tamaño de secuencia	5
Learning rate	0.01
Momentum	0.937

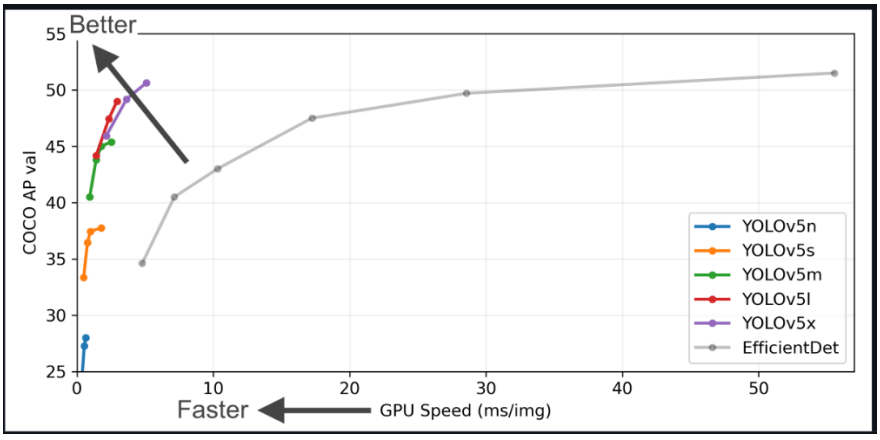
Nota. Tomado de: Ultralytics. (s. f.). *Hyperparameter evolution*. Ultralytics YOLOv8 Docs. https://docs.ultralytics.com/yolov5/tutorials/hyperparameter_evolution/#1-initialize-hyperparameters

Para llevar a cabo el proceso de entrenamiento de la red de yolov5 inicialmente se abordó el dataset previamente etiquetado, ya que esto constituye una parte fundamental en el rendimiento, dado que la calidad de la inferencia del modelo estará directamente relacionada con la variedad del dataset. Por otra parte, es fundamental el comprender los distintos arquetipos Figura 8, que se pueden usar dado que de esta selección dependerá aspectos importantes como lo son el tiempo de entrenamiento y la inferencia misma, el entrenamiento de esta red se realizó en colab y se encuentra en el repositorio de *GitHub.*

https://github.com/gitale9/contadorDePersonasIA/blob/01273ccb5c38e3c4221d5998a90375e9a1542ce6/Entrenamiento/YOLOv5/EntrenamientoConYOLOv5.ipynb

Figura 8

Figura de arquetipos



Nota. Tomado de: *Paperspace*. (s. f.). https://console.paperspace.com/github/gradient-ai/yolov5/blob/master/tutorial-paperspace.ipynb?machine=Free-GPU

Tabla3

Prestaciones de los arquetipos de YOLOv5

Model	size (pixels)	mAP^{val}_{50-95}	mAP^{val}_{50}	Speed CPU b1 (ms)	Speed V100 b1 (ms)	Speed V100 b32 (ms)	params (M)	FLOPs @640 (B)
YOLOv5n	640	28.0	45.7	**45**	**6.3**	**0.6**	**1.9**	**4.5**
YOLOv5s	640	37.4	56.8	98	6.4	0.9	7.2	16.5
YOLOv5m	640	45.4	64.1	224	8.2	1.7	21.2	49.0
YOLOv5l	640	49.0	67.3	430	10.1	2.7	46.5	109.1
YOLOv5x	640	50.7	68.9	766	12.1	4.8	86.7	205.7
YOLOv5n6	1280	36.0	54.4	153	8.1	2.1	3.2	4.6
YOLOv5s6	1280	44.8	63.7	385	8.2	3.6	12.6	16.8
YOLOv5m6	1280	51.3	69.3	887	11.1	6.8	35.7	50.0
YOLOv5l6	1280	53.7	71.3	1784	15.8	10.5	76.8	111.4
YOLOv5x6	1280	55.0	72.7	3136	26.2	19.4	140.7	209.8
+ TTA	1536	**55.8**	**72.7**	-	-	-	-	-

Nota. Tomado de: *Paperspace*. (s. f.-c). https://console.paperspace.com/github/gradient-ai/yolov5/blob/master/tutorial-paperspace.ipynb?machine=Free-GPU

Dada la información resaltada en la Tabla 3 se menciona la métrica mAP (mean Average Precision), la cual es una herramienta que permite cuantificar la precisión de cualquier detector desarrollado por medio de redes neuronales. Sin embargo, dicha herramienta parte de conceptos fundamentales de una matriz de confusión los cuales son:

- Un Verdadero Positivo es el resultado de una predicción acertada ya que coincide con la referencia.
- Un Falso Positivo corresponde a la determinación de que la predicción es válida cuando realmente no lo es.
- Un Verdadero Negativo es el resultado de todas las detecciones correctamente no detectadas.
- Un Falso Negativo corresponde a la no detección cuando no se encuentra en la imagen.

Para efectos del proyecto se seleccionó las versiones de YoloV5x y YoloV5l dado que corresponden a una arquitectura

3.1.3. Detección de Objetos

El modelo que solo miras una vez versión 5 (YOLOV5) se utilizó para detectar personas en tiempo real. Cuando se detectan personas, el modelo producirá las coordenadas del cuadro delimitador alrededor de ellos y el nombre de su clase en este caso personas.

De los conceptos anteriores que componen la denominada matriz de confusión Figura 12, se desprenden aspectos muy importantes que se definen como Precisión y exhaustividad.

La Precisión se define matemáticamente como el cociente entre los Verdaderos positivos y la suma de los verdaderos positivos y los falsos positivos. Es decir, es una medida de "cuándo su modelo adivina, ¿con qué frecuencia lo hace correctamente?" Recordar es una medida de "¿su modelo ha adivinado cada vez que debería haber adivinado? Solawetz (2022), por ende, dicha métrica cuantifica la calidad del modelo.

La exhaustividad es un parámetro el cual cuantifica la cantidad que el modelo es capaz de identificar y se define como el cociente entre los verdaderos positivos y la suma de los verdaderos positivos y los falsos negativos.

3.2 Entrenamiento MaixHub con dataset propio

Actualmente, la empresa desarrolladora Sipeed cuenta con una herramienta para llevar a cabo el entrenamiento de redes neuronales sencillas, dicha plataforma se llama MaixHub y resulta ser una gran alternativa al ser una herramienta proporcionada por la misma empresa desarrolladora la cual integra funciones como el servicio de modelo de IA y la comunicación comunitaria (*Introduction to MaixHub - Sipeed Wiki*, s. f.), por lo que proporciona modelos que permiten la descarga y la implementación en los dispositivos así como el entrenamiento de modelos de detección, entre otros.

Al explorar dicha herramienta, primeramente, se cargó el dataset correspondiente a las imágenes las cuales están divididas entre 918 training Set y 15 validation set como se muestra en la Figura.10, para posteriormente iniciar un nuevo proyecto de detección de personas. Las imágenes están debidamente relacionadas con sus respectivas etiquetas, dado que la misma plataforma lo permite de manera online tal como se muestra en la figura 9. Luego de tener las imágenes debidamente etiquetadas y organizadas se creó la tarea de entrenamiento en el que se llevó a cabo el aumento de datos del dataset (data augmentation) mediante las técnicas de espejo, rotación y desenfoque.

Figura 9

Etiquetado personas en la plataforma MaixHub

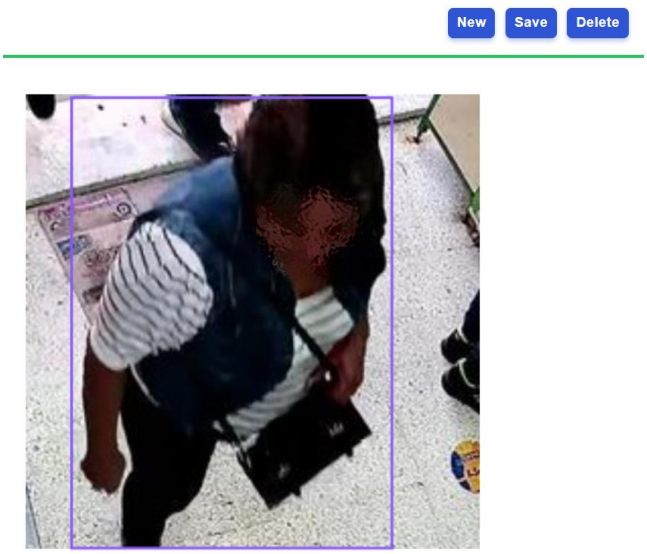
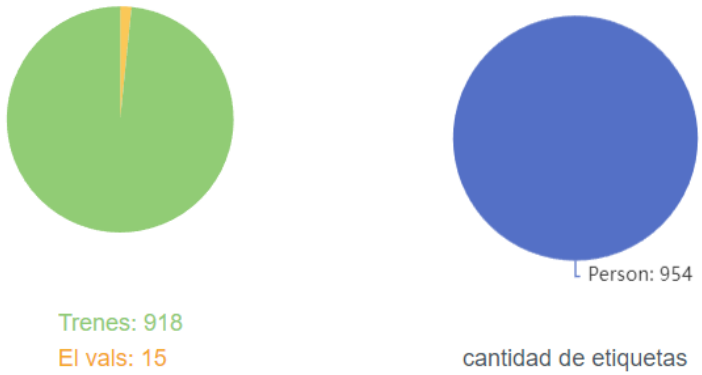


Figura 10

Validación dataset en la plataforma MaixHub



El modelo de transfer learning se desarrolló sobre la red yolo v2 está emplea la red neuronal preentrenada convolucional resnet la cual cuenta con 18 capas de profundidad, y posee un tamaño de 11.18 millones, El tamaño de la entrada de imagen de la red es de 224 por 224, esta red se entrenó bajo los parámetros de la tabla 4, adicionalmente se puede encontrar el entrenamiento de este en el siguiente *GitHub* https://github.com/gitale9/contadorDePersonasIA/tree/0fb6eb138b85d8fa9fcf8653b56e6502c37104bf/Entrenamiento/MaixHub

Tabla 4

Parámetros de entrenamiento Maixhub

Parámetro	Valor
Dimensión de Imagen	224 x 224
Épocas	100
Tamaño de secuencia	4
Learning rate	0.001

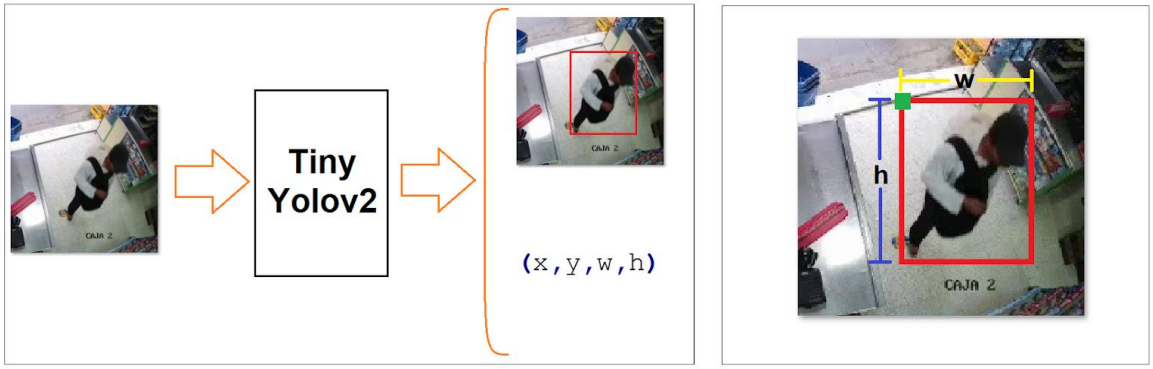
3.3 Entrenamiento Yolov2 con dataset COCO

El entrenamiento con Yolov2 se realizó con el dataset de COCO, se suministran las imágenes al modelo y basado en su funcionamiento éste entrega las coordenadas de la ubicación

de las personas encontradas, dichas coordenadas consisten en cajas delimitadoras del objeto encontrado como se evidencia en la Figura 11.a, en donde para efectos del proyecto se basa en la detección de una única clase. La Figura 11.b. ilustra los valores “x” y “y” obtenidos, los cuales indican las coordenadas del punto verde que corresponden al vértice superior izquierdo del cuadro delimitador; mientras que, los valores “w” y “h” son, respectivamente, el ancho y el largo de la caja delimitadora tal como se aprecia en los recuadros rojos de la Figura 11.

Figura 11

Estructura cuadro delimitador



4. Resultados

En esta sección se presentan todos los resultados obtenidos con cada modelo, con sus respectivas pruebas en tiempo real y gráficas de los valores alcanzados en el entrenamiento.

4.1. Técnica de conteo

A partir de las coordenadas de detección del cuadro delimitador del modelo de detección, se realizó el cálculo del centroide del recuadro el cual nos brinda una posición central de la persona, dicha posición se almacena en un vector a través de los frames que suministra el video, es importante tener en cuenta que el almacenamiento de estos valores solo se tiene en cuenta en una zona de operación definida previamente. Luego de esta detección se realizó una lectura de este vector para determinar el conteo ascendente o descendente del aforo inicial.

4.2. Modelo YoloV5

Para llevar a cabo el entrenamiento del modelo, se hizo uso de la nube de Google Colaboratory dado que es una alternativa muy eficiente al momento de ejecutar redes neuronales por las ventajas a nivel computacional que brinda. Google Colaboratory corresponde a un servicio en la nube basado en Notebook, el cual cuenta con unidades de procesamiento gráfico que permiten el desarrollo de software.

Inicialmente, se clonaron las dependencias necesarias para llevar a cabo la versión de yolov5, las cuales son importantes al momento de ejecutarse. Posteriormente se planteó el entrenamiento con los parámetros definidos en la Tabla 2, con esto se inició el entrenamiento del modelo con la versión de yolov5x el cual fue seleccionado según sus características detalladas en la Tabla 3.

Los resultados del entrenamiento de la red de Yolo v5x se encuentran en la Figura 12 el cual representa la matriz de confusión del entrenamiento, esta información es fundamental para el

desarrollo de las siguientes gráficas, una de ellas corresponde a la Figura 13 la cual relaciona la precisión de esté basándose en el entrenamiento y la validación durante las épocas. Por otra parte, la Figura 14 relaciona la variación de la pérdida en la predicción en cada época del entrenamiento. Finalmente, la Figura 15 ilustra el parámetro más importante al evaluar el rendimiento del modelo de detección el cual corresponde al mAP.

Figura 12

Matriz de confusión

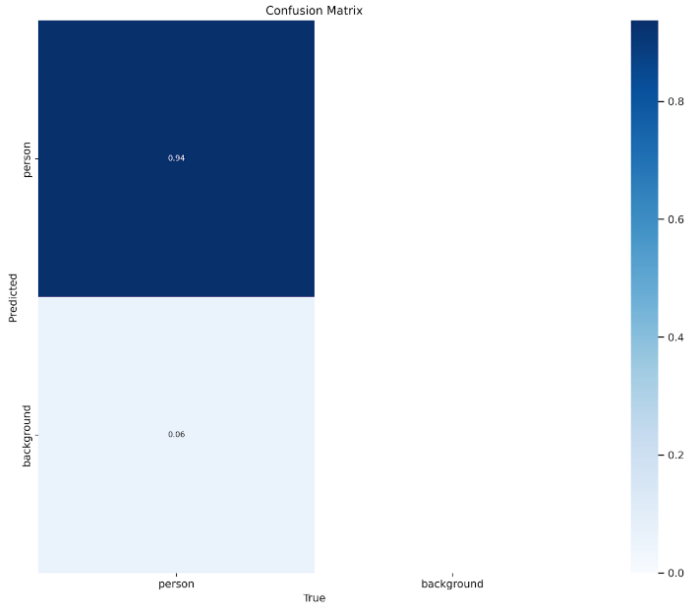


Figura 13

Grafica de precisión.

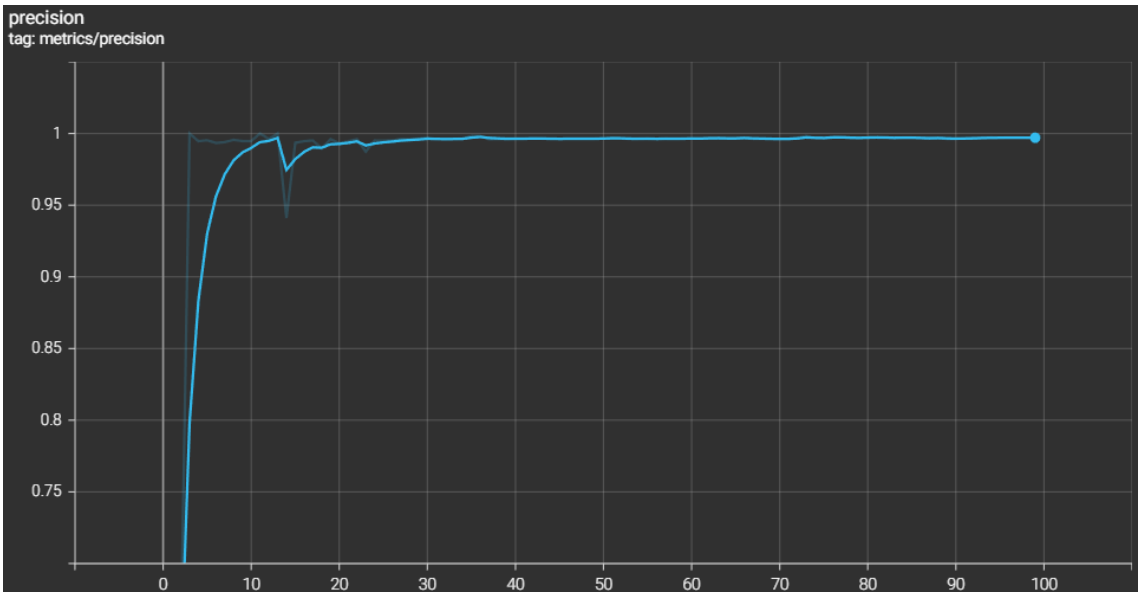


Figura 14

Grafica de perdida.

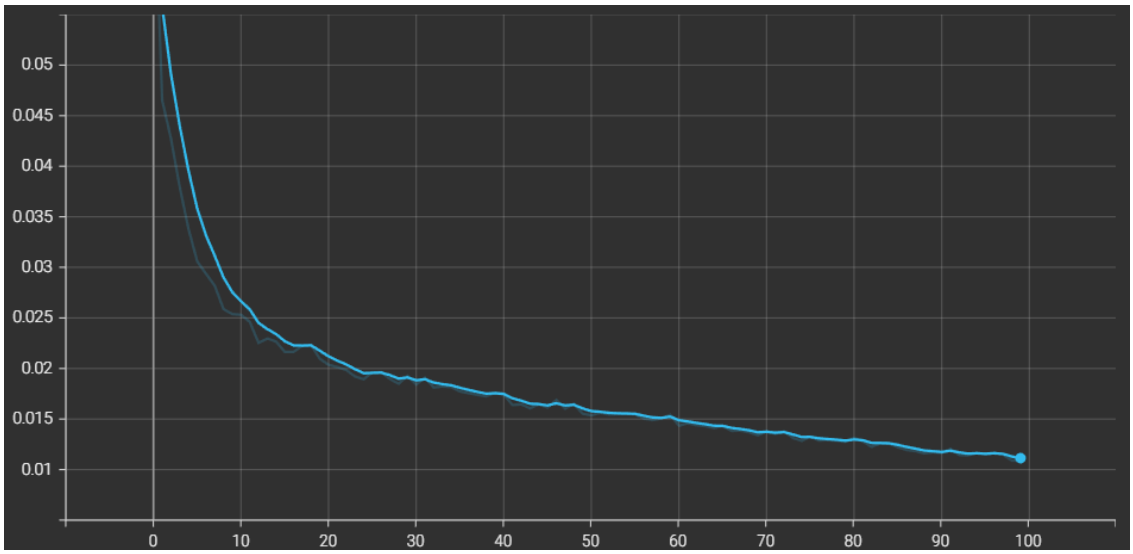
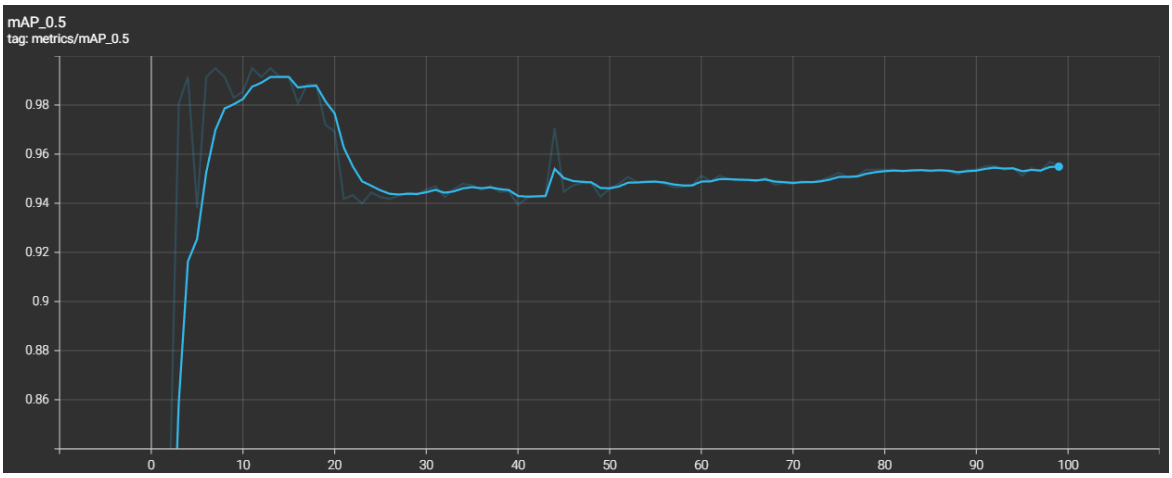


Figura 15

Grafica de mAP



La Tabla 5 muestra los datos finales obtenidos en el modelo de yolo v5x. Los resultados de la validación del modelo se aprecian en la Figura 16, en donde se resalta el cuadro delimitador con la respectiva confianza entregada por el modelo de detección.

Tabla 5

Entrenamiento red de yolov5x

Información	Valor
Precisión entrenamiento	0.9969
Pérdida entrenamiento	0.0109
Pérdida Validación	0.01341
Tiempo entrenamiento(min)	44.52

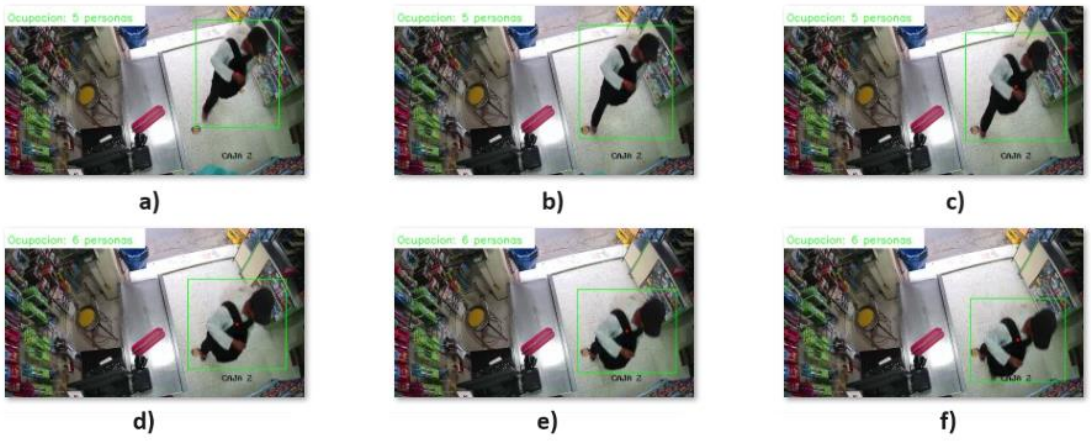
Figura 16

Resultados de la validación.

Con relación a esto se infiere que el modelo desarrollado mejora continuamente en la precisión al transcurrir las épocas del entrenamiento, reduciendo así el margen de error en la predicción y aumentando la exhaustividad de esta.

Figura 17

Resultados de implementación en tiempo real, prueba de precisión.



La Figura 17 refleja la precisión del modelo en las pruebas de tiempo real, se nota detección constante en cada frame, estas detecciones son óptimas para implementar el algoritmo de seguimiento planteado, del cual se presentan los resultados en la Figura 18, estos frames muestran el aforo del establecimiento en la esquina inferior izquierda, en la Figura 18.a se observa aforo de 4 persona, mientras que, debido a la detección del ingreso de una nueva persona, este aforo cambia en la Figura 18.b

Figura 18

Resultados de implementación en tiempo real, prueba de conteo.

4.3. Modelo YoloV2

La Figura 19 presenta la función de pérdida del modelo con respecto a cada época en el entrenamiento, esta gráfica plantea que el modelo sería preciso en el momento de la implementación, lo cual no corresponde dado que los resultados obtenidos de la implementación están reflejados en la Figura 20, dicha figura representa un lote de 18 frames consecutivos, los cuales fueron procesados por el modelo desde el sistema embebido (tarjeta Maix-ii Dock). Esta prueba revela la baja precisión del modelo al ser ejecutado en tiempo real, los resultados de la detección del modelo impiden la implementación del algoritmo de seguimiento planteado para el conteo de las personas encontradas en el establecimiento.

Figura 19

Función de pérdida modelo Yolov2.

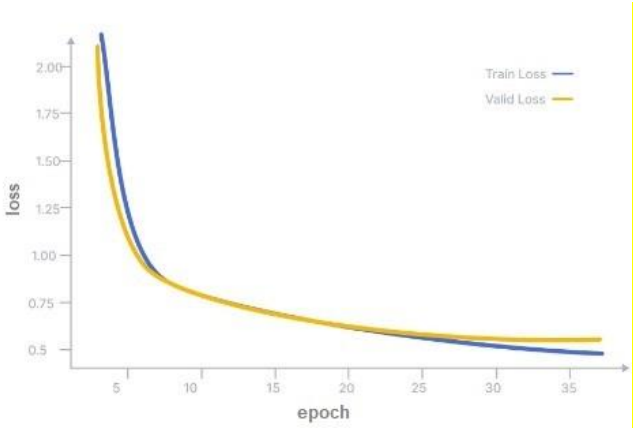


Figura 20

Resultados de implementación.

El modelo no predice las detecciones como se esperaba, tal como se visualiza en la Figura 20, ya que al recorrer los diferentes frames correspondientes al video de prueba no se evidencia una detección continua y estable de la clase asociada, por ende, la implementación del modelo asociado no resulta viable dadas las inconsistencias en la predicción.

4.4. Modelo Maixhub

La Figura 21 muestra cómo varía tanto la función de pérdida como el porcentaje de aciertos a medida que avanza el entrenamiento del modelo; “loss” hace referencia a la función de pérdida total, “loss_pos” es la función de pérdida en la posición del recuadro, “loss_conf” equivale a la función de pérdida para los tamaños de los recuadros y “val_acc” es el porcentaje de acierto del modelo. De acuerdo con estos datos se revela que, en el conjunto de validación, la época de 90 tiene la mejor precisión: 0.909.

Figura 21

Grafica entrenamiento Maixhub.

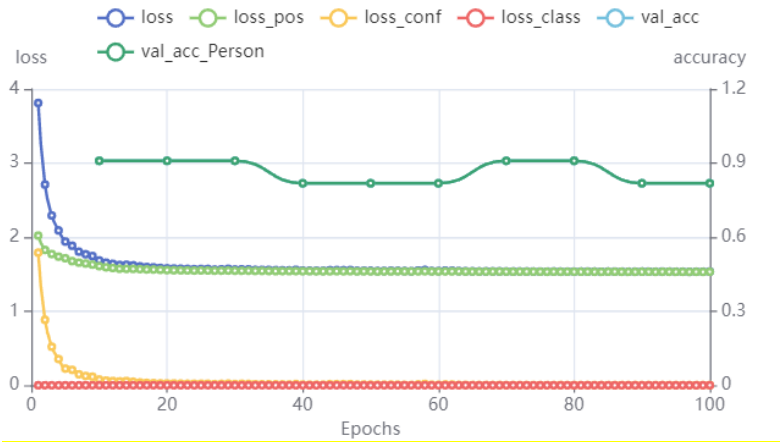


Tabla 6

Entrenamiento red de MaixHub

Resultado para la época 80	Valor
Precisión en validación	0.81818
Pérdida total	1.534
Pérdida posición del recuadro	1.5326
Pérdida tamaño del recuadro	0.0019
Perdida predicción clase	0

Los datos resultados obtenidos en el entrenamiento del modelo están condensados en la Tabla 6. Los valores de la Figura 21 muestran que corresponde a un modelo asertivo para las respectivas detecciones. La Figura 22 muestra los resultados de validación por la plataforma MaixHub, si bien hay algunos verdaderos negativos la forma de los verdaderos positivos es satisfactoria.

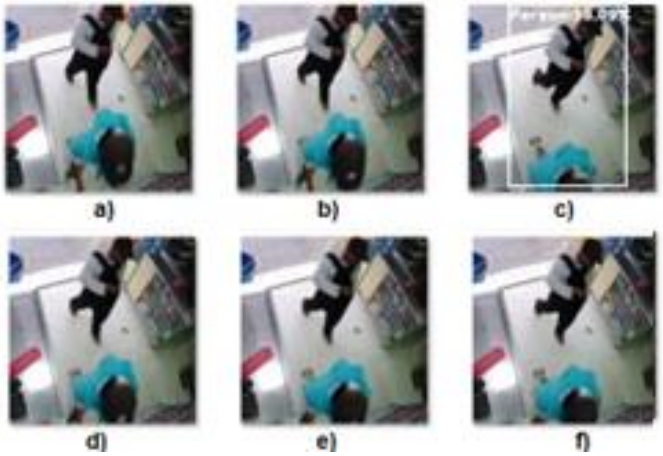
Figura 22

Resultados de validacion Maixhub.

Los resultados de las pruebas en tiempo real mediante la implementación del algoritmo en la tarjeta están reflejados en la Figura 23, en ella se encuentran reflejados los frames consecutivos de un tramo de imágenes tomadas por la tarjeta Maix-ii Dock; también se puede observar que al momento de la implementación la frecuencia de las detecciones es baja, la precisión del modelo cae, de manera que impide la implementación del algoritmo planteado para el conteo de las personas en el establecimiento a partir de estas detecciones.

Figura 23

Resultados implementación Maixhub.



Para finalizar la presentación de resultados, en la Tabla 7 se muestra la tasa de fotogramas por segundo (FPS) procesados por cada sistema embebido al momento de la implementación, esto es de alta importancia dado que como se mencionó en capítulos anteriores, el método de conteo depende fundamentalmente de la detección misma.

Tabla 7

Fps sistemas embebidos

Sistema embebido	FPS
Raspberry Pi 4	7
Maix-II-Dock	14

5. Conclusiones

En este trabajo se propusieron 3 alternativas en modelos de detección basados en redes neuronales convolucionales como principal solución para la detección ante la problemática inicial y en pro del objetivo del proyecto.

Para el primer modelo basado en la alternativa de Yolo v5, los resultados obtenidos indican una alta confiabilidad en las predicciones del modelo dado que los resultados obtenidos tanto en la validación como la puesta en escena son asertivos y constantes. La recomendación para esta solución estaría relacionada hacía la expansión de la base de datos en diferentes ambientes de tal manera que se pueda expandir la estimación hacia las distintas perspectivas de visualización según la referencia, así como la implementación de modelos de seguimiento densos que generen un control más detallado sobre las características de la detección.

En el segundo modelo de la plataforma Maixhub se afirma que es una alternativa sencilla e intuitiva en el entrenamiento de modelos de IA, sin embargo, es una solución que actualmente se encuentra muy limitada dado que restringe aspectos importantes como la cantidad de elementos pertenecientes al dataset. Así mismo, los resultados del entrenamiento arrojan una precisión en la validación por encima del 80% lo cual indicaría una alta asertividad, en cambio, en las pruebas llevadas a cabo se evidencia que las detecciones no son precisas ni confiables a lo largo de los cuadros procesados por lo que representa una alta dificultad en su implementación teniendo en cuenta el método de conteo usado.

La implementación del modelo Yolov2 pre-entrenado resulto no ser un modelo preciso para las detecciones en tiempo real, debido a la diferencia entre las imágenes del entrenamiento con las imágenes de prueba; para la implementación del modelo Yolov2 se sugiere entrenar el

modelo con un dataset específico de la aplicación que se requiera, este entrenamiento se realizó con el fin de experimentar como serian los resultados con un dataset ya existente.

Comparando los dos modelos de Yolo se puede concluir que YOLOv5 es el primer método en usar capa de enfoque y en usar Pytorch, es extremadamente rápido y casi un 95% más pequeño (más liviano) que YOLOv2 el cual tiene parámetros de red reducidos, por esta y muchas mas razones Yolov5 resulta ser una mejor opción a la hora de entrenar nuestro modelo.

6. Trabajo a Futuro

Como continuación a este trabajo y teniendo en cuenta que corresponde a un proyecto de investigación donde convergen diferentes áreas del conocimiento en el cual habrá posibilidades de prolongar el estudio realizado como trabajo futuro a otros investigadores.

La optimización de la red de detección junto con las alternativas para la trasmisión del conteo son aspectos fundamentales en la proyección de este trabajo a futuro. Para la primera se sugiere enriquecer la base de datos con mayor cantidad de imágenes, ya que, de esta manera se ampliaría la variedad de perspectivas a las cuales se expone el modelo, permitiéndole así ampliar su rango de selección dada la diversidad que se podría tener. En cuanto a la presentación del conteo se recomienda explorar las distintas tecnologías en la trasmisión de datos, puesto que, la mayoría de los sistemas embebidos actuales poseen herramientas que permiten acceso a Internet, de esta manera se podría aprovechar la alternativa teniendo una transmisión de datos más eficiente sin saturar el sistema y permitiendo una mejor interacción para el usuario. Adicionalmente, los datos obtenidos del conteo se podrían almacenar en una base de datos para realizar un estudio de mercado de los días más concurridos en el establecimiento.

Referencias Bibliográficas

Ahmed, I., Ahmad, M., Adnan, A. et al. Person detector for different overhead views using machine learning. Int. J. Mach. Learn. & Cyber. 10, 2657–2668 (2019). https://doi-org.bibliotecavirtual.uis.edu.co/10.1007/s13042-019-00950-5

DATASHEET Raspberry Pi 4 Model B. (2019, junio). Recuperado 6 de julio de 2023, de https://datasheets.raspberrypi.com/rpi4/raspberry-pi-4-datasheet.pdf

Getting Started with YOLO v2 - MATLAB & Simulink. (s. f.). https://www.mathworks.com/help/vision/ug/getting-started-with-yolo-v2.html

Huazhong Xu, Pei Lv and Lei Meng, "A people counting system based on head-shoulder detection and tracking in surveillance video," 2010 International Conference On Computer Design and Applications, Qinhuangdao, 2010, pp. V1-394-V1-398, doi: 10.1109/ICCDA.2010.5540833.

Madhumathi, C. S., Naveen, V., Akshay, N., Kumar, M. S., & Aslam, M. M. (2023). Advanced wild animal detection and alert system using YOLO V5 model. 2023 7th International Conference on Trends in Electronics and Informatics (ICOEI), 365–371.

Maix Go - Sipeed Wiki. (s. f.). https://wiki.sipeed.com/soft/maixpy/en/develop_kit_board/maix_go.html

MaixHub. (s. f.). https://maixhub.com/model/zoo/74

Maix-II-Dock(M2dock) introduction - Sipeed Wiki. (s. f.). https://wiki.sipeed.com/hardware/en/maixII/M2/resources.html

Maslov, D. (2021b, enero 28). *Sipeed Maix-II SoM and Devboard for AI/IoT/Machine Learning - Latest Open Tech From Seeed*. Latest Open Tech From Seeed.

https://www.seeedstudio.com/blog/2021/01/28/sipeed-maix-ii-som-and-devboard-for-ai-iot-machine-learning/

Mercado de sistemas de conteo de personas Insights. (s. f.). https://www.mordorintelligence.com/es/industry-reports/people-counting-system-market

Salinas, M. (2022, 9 marzo). YoloV5 archivos • Saturdays.AI. Saturdays.AI. https://saturdays.ai/tag/yolov5/#:~:text=El%20modelo%20seleccionado%20fue%20el,para%20detectar%20objetos%20en%20im%C3%A1genes.

Sarker, I. H. (2021). Machine learning: Algorithms, real-world applications and research directions. SN Computer Science, 2(3), 160. https://doi.org/10.1007/s42979-021-00592-x

Sarker, I.H. AI-Based Modeling: Techniques, Applications and Research Issues Towards Automation, Intelligent and Smart Systems. SN COMPUT. SCI. 3, 158 (2022). https://doi-org.bibliotecavirtual.uis.edu.co/10.1007/s42979-022-01043-x

Sbt. (2018, 29 enero). *Conteo de personas: qué es, para qué sirve y por qué debes tenerlo - SBT*. SBT. https://www.sb-tec.com/conteo-de-personas-que-es/

Section 2.2. Anatomy of an Embedded System | Embedded Linux Primer: A Practical Real-World Approach. (s. f.-b). https://flylib.com/books/en/1.98.1.18/1/

Sipeed Wiki. (s. f.). https://wiki.sipeed.com/soft/maixpy3/zh/usage/AI_net/yolo.html

Solawetz, J. (2022). Mean Average Precision (mAP) in Object Detection. *Roboflow Blog*. https://blog.roboflow.com/mean-average-precision/#what-is-the-confusion-matrix

Supeshala, C. (2021, 15 diciembre). YOLO V4 or YOLO V5 or PP-YOLO? Which should I use? | Towards data science. Medium. https://towardsdatascience.com/yolo-v4-or-yolo-v5-or-pp-yolo-dad8e40f7109

Support Vector Machine (SVM). (MathWorks). MATLAB & Simulink. https://la.mathworks.com/discovery/support-vector-machine.html#:~:text=Support%20vector%20machine%20(SVM)%20es,reconocimiento%20de%20im%C3%A1genes%20y%20voz

Redes neuronales profundas preentrenadas - MATLAB & Simulink - MathWorks España. (s. f.). https://es.mathworks.com/help/deeplearning/ug/pretrained-convolutional-neural-networks.html

(Rus, 2019)

Rus, C. (2019, junio 24). *Raspberry Pi 4 es oficial: una completa actualización con procesador Cortex-A72, hasta 4 GB de RAM y desde 35 dólares*. Xataka.com; Xataka. https://www.xataka.com/ordenadores/raspberry-pi-4-caracteristicas-precio-ficha-tecnica

Ultralytics. (s. f.). *Hyperparameter evolution*. Ultralytics YOLOv8 Docs. https://docs.ultralytics.com/yolov5/tutorials/hyperparameter_evolution/#1-initialize-hyperparameters

V. H. Roldão Reis, S. J. F. Guimarães and Z. K. Gonçalves do Patrocínio, "Dense Crowd Counting with Capsule Networks," 2020 International Conference on Systems, Signals and Image Processing (IWSSIP), Niteroi, Brazil, 2020, pp. 267-272, doi: 10.1109/IWSSIP48289.2020.9145163.