MAESTRÍA EN SISTEMAS EMBEBIDOS

MEMORIA DEL TRABAJO FINAL

Cámara IoT para detección facial con conectividad Wi-Fi

Autor:
Esp. Ing. Mauricio Barroso Benavides

Director:
Mg. Ing. Gonzalo Sanchez (FF.AA, FIUBA)

Jurados:
Mg. Ing. Edgardo Torrelli (FIUBA)
Mg. Lic. Leopoldo Zimperz (FIUBA)
Mg. Ing. Sebastián Guarino (FIUBA)

Este trabajo fue realizado en la ciudad de Tupiza,
entre junio de 2021 y diciembre de 2022.

Resumen

Esta memoria describe el proceso de desarrollo de un dispositivo electrónico compuesto principalmente por un módulo de procesamiento con conectividad Wi-Fi y una cámara, que puede capturar imágenes para procesarlas mediante algoritmos de Inteligencia Artificial y así detectar rostros humanos. Los datos generados por el dispositivos son transmitidos hacia servidores en la nube encargados de procesar, almacenar y facilitar su visulización para los usuarios finales. La principal aplicación de este trabajo es generar información sobre la presencia de personas para, por ejemplo, activar otros dispositivos como bocinas o mecanismos de cierre/apertura de puertas.

En la realización del presente trabajo se utilizaron conocimientos adquiridos a lo largo de la carrera como desarrollo de firmware, visión artificial, diseño de hardware, sistemas distribuidos, gestíon de proyectos y gestión de tecnología.

Agradecimientos

A Gonzalo Sanchez, director de este trabajo, por sus valiosos consejos y criterios que se ven reflejados a lo largo de este desarrollo.

A los profesores de la Maestria en Sistemas embebidos, por contribuir en mi formacion academica con sus conocimientos y experiencias.

Índice general

Índice de figuras

Índice de tablas

Este trabajo se lo dedico a mi familia, eternas gracias por sus apoyo incondicional en cada etapa de mi vida, Ustedes son la luz que guia mi camino

.

Capítulo 1

Introducción general

En este capítulo se presentan conceptos básicos sobre las tecnologías y técnicas que fueron utilizadas en el desarrollo del trabajo. Se abordan nociones sobre inteligencia artificial, aprendizaje automático, aprendizaje profundo, redes neuronales convolucionales, visión artificial y servicios en la nube. También se citan trabajos anteriores que inspiraron a este, las motivaciones para llevarlo a cabo junto a sus objetivos y alcances.

1.1. Inteligencia Artificial, Aprendizaje Automático y Aprendizaje Profundo

AI (*Artificial Intelligence*, Inteligencia Artificial), ML (*Machine Learning*, Aprendizaje Automático) y DL (*Deep Learning*, Aprendizaje Profundo), son términos muy utilizados hoy en día en el mundo del desarrollo tecnológico [ref]. Aunque estos términos son muy parecidos, entre ellos existen dependencias que pueden ser visualizadas con ayuda de la figura 1.1.

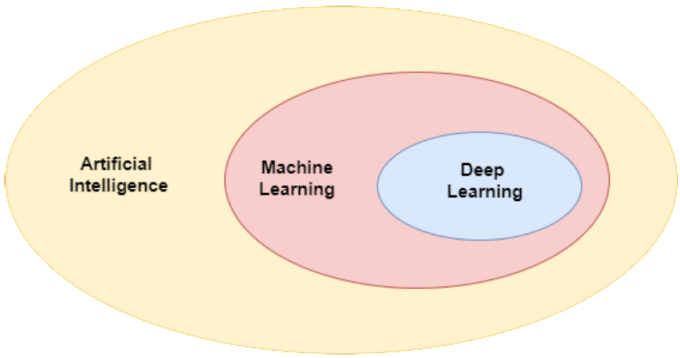

FIGURA 1.1. Diferencias entre AI, ML y DL.

AI es un área de la computación que permite a los sistemas computacionales imitar la inteligencia humana para entender su entorno y tomar acciones que maximicen sus posibilidades de lograr sus objetivos [ref]. Sus aplicaciones más importante se ecuentran en la áreas de comercio, educación, robótica, salud, agricultura, automotriz y finanzas[https://www.simplilearn.com/tutorials/artificial-intelligence-tutorial/artificial-intelligence-applications]. Todos los sistemas de inteligencia artificial reales e hipotéticos pueden ser clasificados en alguno de los siguientes tipos:

- ANI (*Artificial Narrow Intelligence, Inteligencia Artificial Estrecha*): también conocida como inteligencia artificial débil, su objetivo es llevar a cabo un solo tipo de tarea. Estos sistemas no poseen conciencia y no son manejados por sentimientos como lo haría un humano. Algunos ejemplos de ANI son los *chatbots* o los automóviles autónomos.
- AGI (*Artificial General Intelligence*, Inteligencia Artificial General): también conocida como inteligencia artificial fuerte, es un concepto en el que las máquinas exhiben inteligencia humana. Estos sistemas tendrían la capacidad de aprender, entender y actuar de tal manera que sería indistinguible a un humano. AIG actualmente no existe, pero es utilizado en industrias como el cine
- ASI (*Artificial Super Intelligence*, Super Inteligencia Artificial): ASI también forma parte de la inteligencia artificial fuerte. Se le considera muy poderosa por ser capaz de volverse consciente y autónoma. No sólo replica el comportamiento humano, sino que lo supera. Puede pensar mejor y tener más habilidades. Sin embargo, esta tecnología aún está en desarrollo.

ML es un subconjunto de AI que utiliza algoritmos de aprendizaje estadísticos para construir sistemas con la habilidad de aprender automáticamente y mejorar a partir de experiencias previas sin ser explícitamente programados para esto. Muchos de los servicios de recomendación utilizados por empresas como Netflix, YouTube o Spotify, utilizan ML para adaptarse a un usuario en particular y ofrecer una mejor experiencia más personalizada. Estos algoritmos pueden ser clasificados de la siguiente manera:

- Aprendizaje supervisado: se refiere al aprendizaje modelos a partir de un conjunto de datos, mejor conocidos como *dataset*, cuyas respuestas son conocidas con antelación y están asociadas a una etiqueta o *label*. Por ejemplo, el *dataset* pueden ser muchas fotografías de gatos y el *label* asociado el nombre de este animal. De esta manera el modelo es entrenado para generar predicciones de datos nuevos. En la figura 1.2 se representa gráficamente el aprendizaje supervisado.

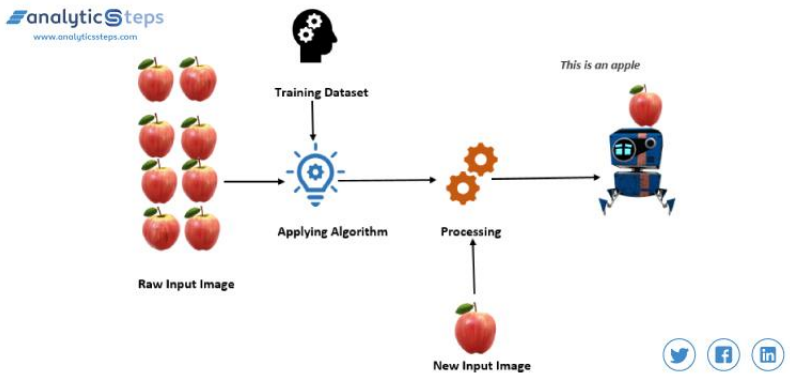

FIGURA 1.2. Aprendizaje supervisado.

- Aprendizaje no supervisado: es utilizado cuando los datos utilizados para el aprendizaje no tienen *labels*. Su objetivo principal es aprender acerca de los datos e inferir patrones sin ningún tipo de referencia sobre las respuestas esperadas. Es mayormente utilizado como parte del análisis exploratorio

de datos [https://towardsdatascience.com/understanding-the-difference-between-ai-ml-and-dl-cceb63252a6c]. En la figura 1.3 se representa gráficamente el aprendizaje no supervisado.

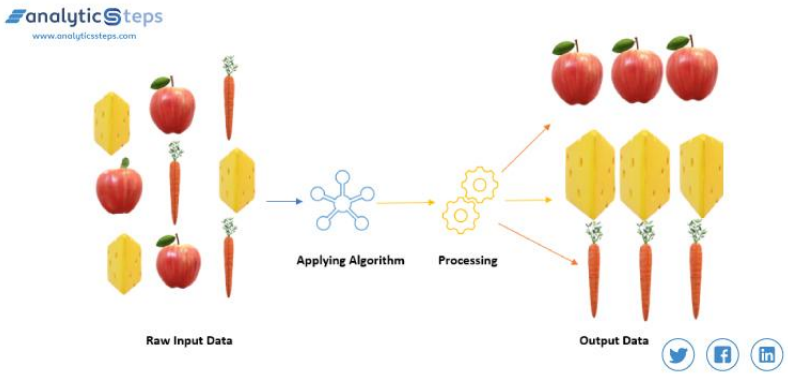

FIGURA 1.3. Aprendizaje no supervisado.

- Aprendizaje reforzado: es el aprendizaje mediante la interacción continua con el entorno con el método de prueba y error, y utiliza continuamente la retroalimentación de sus acciones y experiencias previas. Este tipo de aprendizaje utiliza recompensas si se realizan acciones correctas y penalizaciones si son incorrectas. En la figura 1.4 se representa gráficamente el aprendizaje reforzado.

FIGURA 1.4. Aprendizaje reforzado.

DL es una técnica de ML que está inspirada en la forma en la que el cerebro humano filtra información. Cómo DL procesa información de manera similar al cerebro humano sus aplicaciones son tareas que un humano generalmente realiza, como distinguir entre un peatón o un poste de luz en el caso de automóviles autónomos. El componente principal de DL son las redes neuronales artificiales, que son capas de nodos interconectados, donde existen una capa de entrada, una o varias capas ocultas y una capa de salida. En la figura 1.5 se puede observar la arquitectura de una red neuronal artificial.

Cada uno de los nodos de las capas ocultas y de salida, tienen como entrada la salida de los nodos anteriores multiplicadas por unos términos denominados pesos o *weights* y que sumados junto a otro término llamado sesgo o *bias* pasan por una función de activación no lineal para generar su salida. En la figura 1.6 se visualiza un nodo de las capas ocultas o de salida.

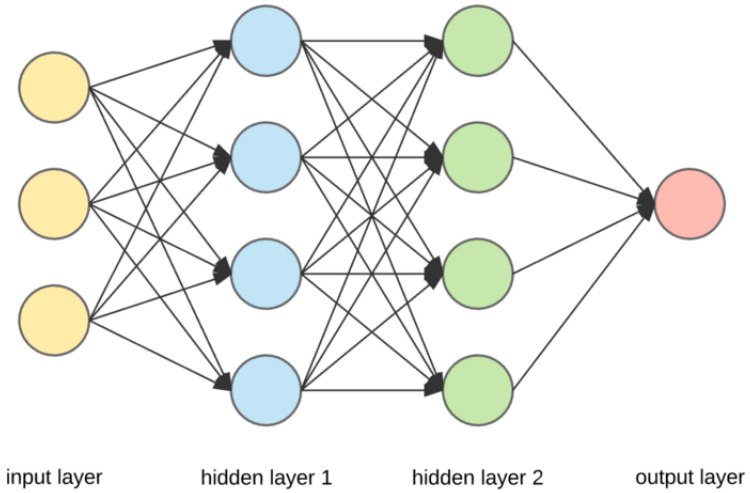

FIGURA 1.5. Arquitectura de una red neuronal artificial.

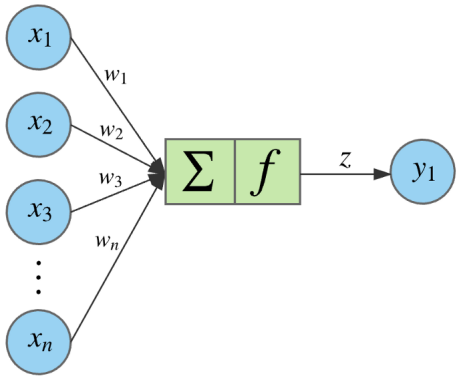

FIGURA 1.6. Nodo de una red neuronal artificial.

1.2. Redes neuronales convolucionales

También conocidas como CNN (*Convolutional Neural Networks*, Redes Neuronales Convolucionales) o ConvNet, son un algoritmo de DL que están orientadas a recibir como entrada una, asignarle *weights* y *biases* entrenables a varios aspectos/objetos en la imagen para poder diferenciarlas unas de otras. Su uso reduce el pre procesamiento de las imágenes de entrada con respecto a otros modelos de clasificación, ya que los filtros necesarios son incorporados en su arquitectura y tienen la habilidad de ser entrenados.

Computacionalmente una imagen puede ser muy difícil de procesar, esto depende del espacio de colores donde se encuentra [ref] y las dimensiones que posee. Por ejemplo una imagen RGB (*Red Green Blue*, Rojo Verde Azul) y de dimensiones 1920x1080 pixeles, tiene un tamaño de 6220800 bytes. El objetivo principal de las CNN es reducir la dimensionalidad de las imágenes de entrada, de tal forma que sean más fáciles de procesar y no pierdan sus características o *features* principales que son críticas para obtener una buena predicción.

La arquitectura de una CNN es independiente del tipo de aplicación, donde las capas que lo componen son elegidas en función de los objetivos que se persiguen. En la figura 1.7 se puede observar la arquitectura de una CNN para clasificar dígitos escritos a mano.

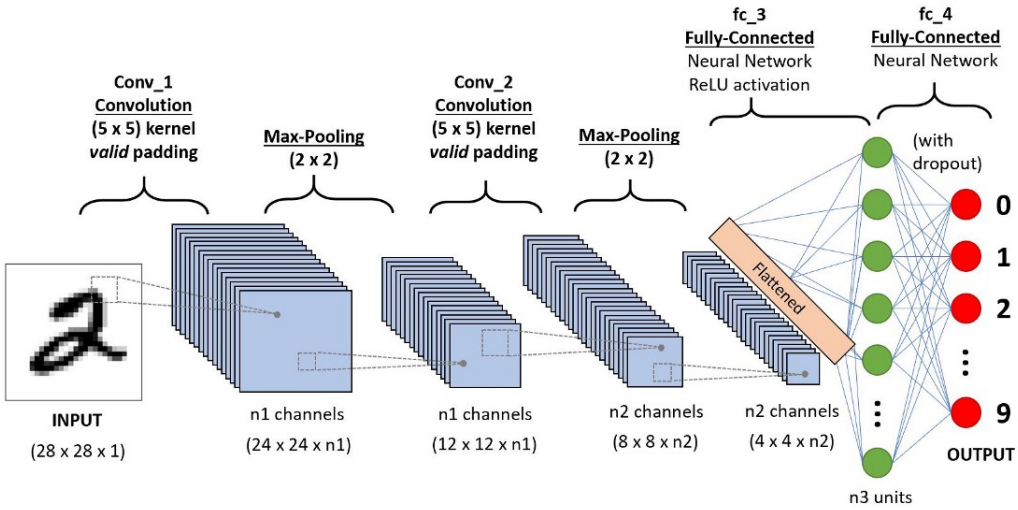

FIGURA 1.7. CNN para clasificar dígitos escritos a mano.

En la arquitectura de la figura 1.7 se pueden observar tres capas principales para construir una CNN: capa de convoluciones, capa de *pooling* y capa *fully-connected*.

1.2.1. Capa de convoluciones

Esta capa es la encargada de aplicar la operación de convolución sobre las imágenes de entrada para encontrar patrones que más adelante permitirán clasificarlas. La convolución de una imagen con un *kernel* no es más que la aplicación del operador punto entre ambos. Este tipo de capas se definen por:´

- El número de los *kernels* o filtros que se aplican a la imagen, que es el número de matrices por las que se van a convolucionar las imágenes de entrada.
- El tamaño de los *kernels*, donde casi siempre tienen dimensiones cuadradas e impares como 3x3 o 5x5.
- El *stride* o paso, se refiere a la forma en como el *kernel* recorre la imagen.

En la figura 1.8 se puede observar como un

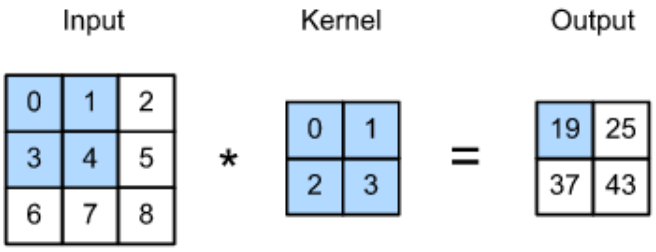

FIGURA 1.8. Convolución de una entrada de 3x3 con un *kernel* de 2x2 y *stride* de 1.

1.2.2. Capa de *pooling*

Similar a la capa de convoluciones, tiene el objetivo de reducir la dimensionalidad de los *features* obtenidos mediante las convoluciones aplicadas en la capa

anterior, para reducir el poder computacional requerido en un principio. Existen dos tipos de dos tipos: *max pooling* y *Average pooling*. El primero retorna el valor máximo de una porción de la imagen cubierta por el *kernel* y el segundo el valor promedio o *average*. *Max pooling* también funciona como supresor de ruido al mismo tiempo que reduce la dimensionalidad. Mientras que *average pooling* solo sirve para reducir la dimensionalidad. En la figura 1.9 se pueden observar estos tipos de *pooling*.

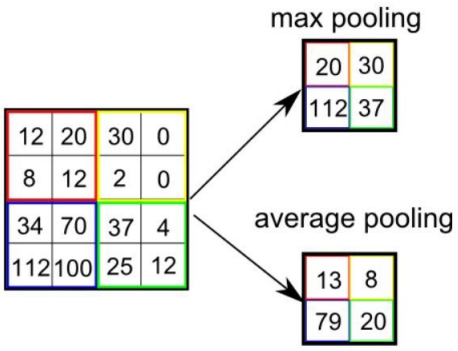

FIGURA 1.9. Tipos de *pooling*.

1.2.3. Capa *fully-connected*

También conocida como capa lineal o FC (*Fully Connected*, Totalmente Conectada), es simplemente una red neuronal artificial como la mostrada en la sección anterior y se utiliza después de que las capas de convolución y *pooling* desglosan los *features* más importantes presentes en la imagen de entrada la CNN. La capa FC brinda las probabilidades finales para cada *label* esperado.

1.3. Vision artificial

La visión artificial o *computer vision* es un campo científico interdisciplinario que se encarga de cómo los sistemas computacionales pueden obtener un entendimiento de alto nivel de imágenes y videos digitales, para comprender y automatizar tareas como lo haría un sistema de visión humano. Las tareas que ejecuta un sistema de visión artificial son de adquisición, procesamiento, análisis y entendimiento de imágenes. En la figura 1.10 se pueden apreciar los componentes de un sistema de visión artificial y un sistema de visión humano.

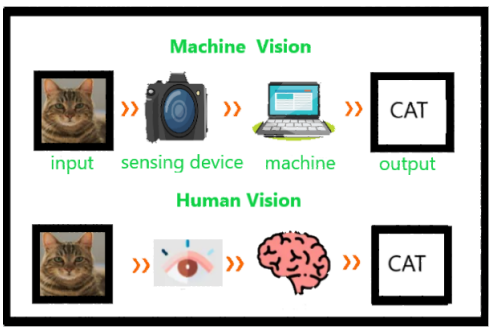

FIGURA 1.10. Componentes de un sistema de visión artificial y un sistema de visión humano.

1.3.1. Detección facial

Uno de los campos de estudio más importantes de la visión artificial es la detección facial. La detección facial puede ser considerada como un caso particular de la detección de objetos y tiene los objetivos de detectar y localizar todos los rostros humanos contenidos en una imagen digital. En la figura 1.11 se puede observar una imagen procesada por un sistema de detección facial.

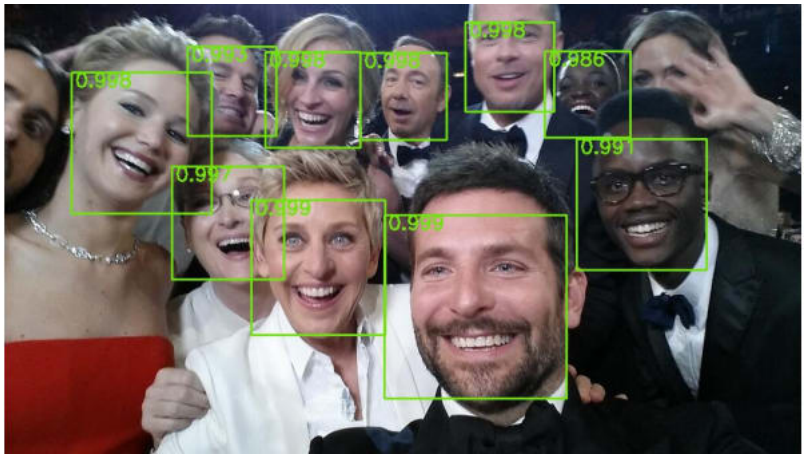

FIGURA 1.11. Imagen procesada por un sistema de detección facial.

Hoy en día, muchos dispositivos comerciales y profesionales como smartphones, tablets y robots, utilizan la detección facial como primer paso para otro tipo de aplicaciones más complejas, entre las que destacan: reconocimiento facial, computación afectiva y grabación de vídeo inteligente.

1.4. Servicios en la nube

El término servicios en la nube hace referencia a un amplio rango de servicios ofrecidos bajo demanda a compañías y usuarios a través de internet. Estos servicios están diseñados para proveer de una manera fácil y asequible acceso a aplicaciones y recursos, sin la necesidad de una infraestructura o hardware propios.

Los servicios en la nube son administrados totalmente por proveedores de computación en la nube [ref]. Estos se encuentran disponibles para los usuarios desde los servidores de los proveedores, por lo que no es necesario que una empresa aloje aplicaciones en sus propios servidores.

De manera general, existen tres tipos básicos de servicios en la nube [ref]:

- SaaS (*Software as a Service*, Software como un Servicio): en este servicio el proveedor sólo proporciona el software o aplicaciones en la nube mediante internet. Los clientes tienen acceso a través de APIs (*Application Programming Interface*, Interfaz de Programación de Aplicaciones) o a través de la web, que les permite interactuar de manera sencilla, sin la necesidad de gestionar, instalar ni actualizar el software.
- IaaS (*Infrastructure as a Service*, Infraestructura como un Servicio): este servicio implica la contratación de una infraestructura de hardware a un tercero,

donde varios cliente comparten los recursos de una máquina física. El proveedor proporciona a sus clientes el acceso a los recursos computacionales necesarios para almacenar o ejecutar tareas que pueden incluir servidores, redes, *backup*, *firewalls*, entre otros.

- PaaS (*Platform as a Service*, Plataforma como un Servicio): es un servicio que se encuentra conceptualmente entre SaaS e IaaS al eliminar la parte física de la infraestructura y ofrece una plataforma donde los cliente pueden crear, desarrollar, gestionar y distribuir sus aplicaciones. El proveedor es el encargado de la gestión y mantenimiento de la plataforma y permite que los clientes se dediquen exclusivamente al desarrollo.

En la figura 1.12 se puede observar las diferencias entre estos servicios en función de los elementos que pueden gestionar.

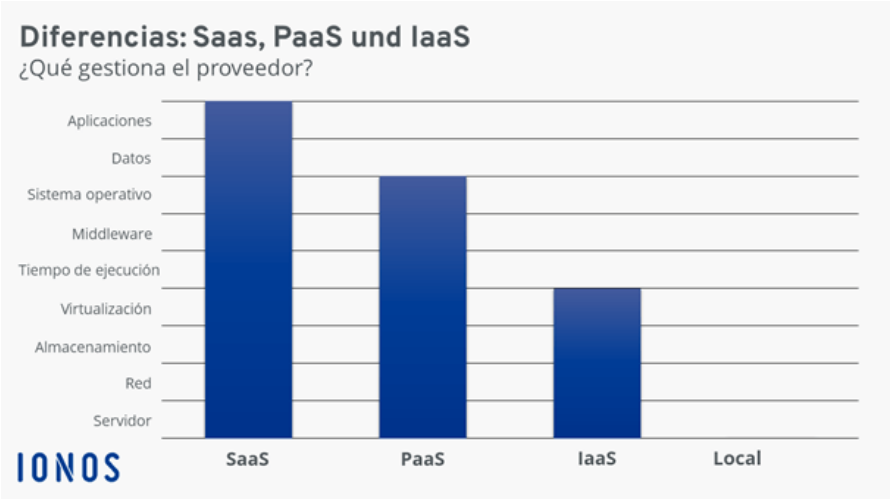

FIGURA 1.12. Capacidades de SaaS, IaaS y PaaS[1].

1.5. Motivación

Gracias a la amplia gama de platarformas de hardware y la disponibilidad de bibliotecas de codigo abierto para implementar AI, ML y DL, ademas de la difusion de informacion en foros y sitios web especializados, es posible desarrollar sistemas de vision artificial personalizados para distintos tipos de arquitecturas [nwengineeringllc.com/article/computer-vision-in-embedded-systems-and-ai-platforms.php].

Estas bibliotecas de código para implementar visión artificial no suelen ser aptas para cualquier dispositivo, ya que son, en la mayoría de los casos, muy grandes en tamaño y requieren de una capacidad de procesamiento lo suficientemente grande para ejecutarse en tiempo real. Pero los constantes esfuerzos de las empresas para ofrecer *frameworks* optimizados que reducen el tamaño y poder de procesamiento necesarios para ejecutar estos algoritmos, la integración de aceleradores de hardware para redes neuronales y DSP (*Digital Signal Processor, Procesador Digital de Señales*) en SoCs actuales (*System on a Chip*, Sistema en un Chip) y

[1]Imagen tomada de: https://expansionba.com.ar/2020/05/23/medidas-para-amortiguar-el-costo-energetico-en-pymes/

el estudio de nuevas y mejoradas arquitecturas para visión artificial basadas en CNN, permiten el desarrollo de dispositivos embebidos con la capacidad de ejecutar modelos de visión artificial. Algunas de sus aplicaciones más importantes son: industria 4.0, seguridad, vehículos autónomos y robótica.

La motivación principal de este trabajo fue desarrollar un sistema embebido de bajo costo económico, bajo consumo energético y de código abierto, que integre algoritmos de DL para visión artificial enfocado a la tarea de detección facial.

Una motivación adicional fue integrar otra tecnología actual como es IoT (*Internet of Things*, Internet de las Cosas), para trabajar en conjunto con los algoritmos de visión artificial. Así las aplicaciones que se pueden obtener son más versátiles a la hora de su implementación en entornos urbanos.

1.6. Estado del arte

Como antecedente existe un trabajo denominado Ä New IoT Combined Face Detection of People by Using Computer Vision for Security Applicationde ... El *paper* donde se describe su trabajo presenta el desarrollo de un dispositivo electrónico que tiene como componentes principales una Raspberry Pi 3, un sensor PIR (*Passive Infra Red*, Pasivo Infrarrojo) y una cámara. Su objetivo principal es detectar personas con ayuda del sensor de movimiento PIR, fotografiarlas y aplicar el algoritmo de detección facial Haar Cascade [ref], para posteriormente guardar una imagen del rostro detectado y visualizarla en un teléfono móvil con ayuda de la aplicación Telegram. En la figura 1.13 se puede observar el diagrama en bloques del sistema descrito anteriormente.

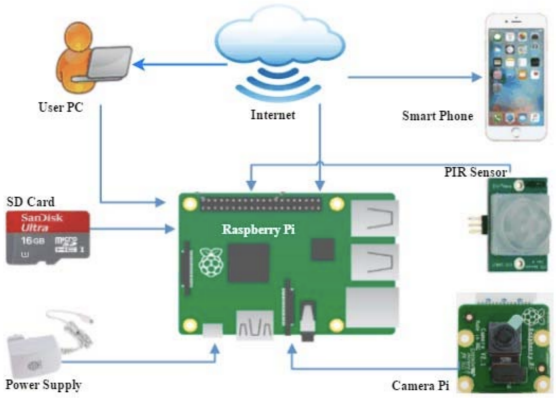

FIGURA 1.13. Diagrama en bloques del sistema propuesto por ...

1.7. Objetivos y alcance

El objetivo principal de este trabajo fue desarrollar un sistema embebido con la capacidad de ejecutar modelos de AI para detectar y localizar rostros humanos de imágenes digitales capturadas por su cámara.

El alcance de este trabajo incluyó:

- Construcción de un prototipo de pruebas
- Desarrollo e implementación de los modelos de AI necesarios

- Implementación de los servicios en la nube necesarios

1.8. Requerimientos

Los requerimientos planteados para este trabajo fueron:

1. Requerimientos funcionales
   a) El sistema debe detectar y contar todos los rostros existentes de las imágenes obtenidas por su cámara con ayuda de las técnicas de procesamiento de imágenes pyramid image y slidding window, y modelos de DL que alcancen una precisión de al menos 80 %.
   b) El sistema debe conectarse a una red Wi-Fi existente a través de algún mecanismo de aprovisionamiento de credenciales de red.
   c) El sistema debe establecer comunicación con los servidores de AWS.
   d) El sistema debe ser alimentado mediante dos baterías AA de litio.
   e) El sistema debe poseer mecanismos de seguridad implementados tanto en hardware como en firmware para evitar su manipulación incorrecta.
   f) El sistema debe funcionar solamente si se detecta movimiento en el sector donde se encuentra instalada.
2. Requerimientos no funcionales
   a) El sistema debe tener un costo de desarrollo igual o menor a US$200.
   b) El sistema debe tener documentación adecuada sobre su uso y desarrollo.

Capítulo 2

Introducción específica

Este capitulo expone una descripción detallada del sistema, del hardware utilizado y las herramientas de software necesarias en el desarrollo del trabajo. Se abarcan la descripcion del sistema y sus componentes, los *frameworks* y modelos utilizados para detección facial y las herramientas utilziadas en la web.

2.1. Diagrama general del sistema

El sistema desarrollado en este trabajo consta de varios componentes de hardware que interconectados entre si son capaces de cubrir . En la figura 2.1 se muestra el diagrama en bloques del sistema.

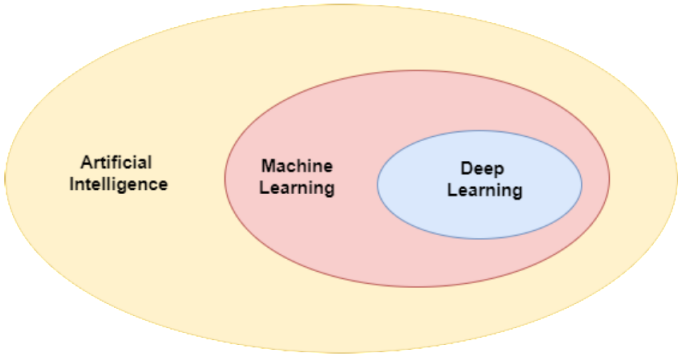

FIGURA 2.1. Diagrama en bloques del sistema.

2.1.1. Placa de desarrollo ESP32-S3-DevKitC-1

El componente central del sistema es la tarjeta de desarrollo ESP32-S3-DevKitC-1-N8R8 de la empresa Espressif. Tiene como componente central el modulo ESP32-S3-WROOM-1-N8R8 y varios otros componentes que simplifican el proceso de desarollo de aplicaiones para IoT. En la figura 2.2 se observa una fotografia de la tarjeta con el detalle de sus componentes.

El modulo ç-WROOM-1-N8R8 es un potente modulo MCU (*Microcontroller Unit*, Unidad de Microcontrolador) de doble nucleo que incorpora Wi-Fi y BLE (*Bluetooth Low Energy*, Bluetooth de Baja Energia) y tiene un amplio conjunto de perifericos. Sus especificaciones tecnicas mas relevante son:

- SoC embebido: ESP32-S3R8
- Procesador: Xtensa LX7 doble nucleo de 32 bits hasta 240 Mhz

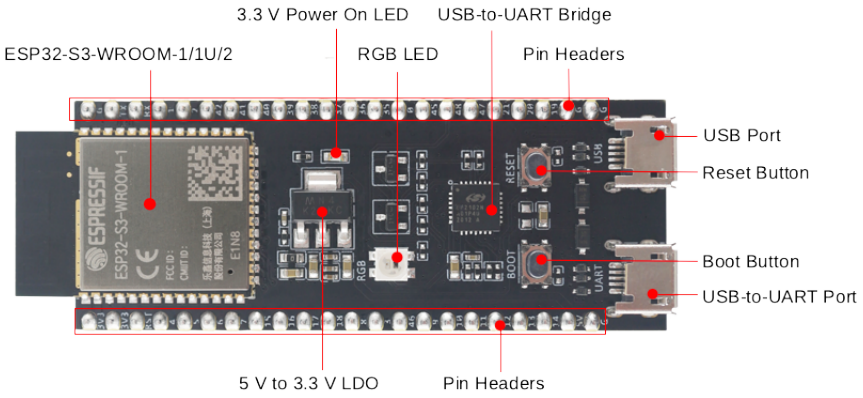

FIGURA 2.2. Componentes del ESP32-S3-DevKitC-1.

- ROM: 384 KB
- SRAM: 512 KB
- Pines: 41
- Flash: 8 MB
- PSRAM (*Pseudostatuc RAM*, RAM Pseudiestatica): 8 MB
- Tipo de antena: PCB
- Wi-Fi: 802.11 b/g/n hasta 150 Mbps
- Bluetooth: Bluetooth 5 y Bluetooth *mesh*
- Perifericos: GPIO, I2C, SPI, interfaz LCD, interfaz de camara, UART, I2S, USB, PWM, ADC, sensor tactil, sensor de temperatura, timer y *watchdogs*
- Temeperatura: –40 65 °C

En el mercado existen muchos fabricantes que ofrecen tarjetas de desarrollo de caracteristicas tecnicas que podrian haber sido utilizadas para el desarrollo de este trabajo. Sin ir muy lejos, Espressif, fabricante de la ESP32-S3-DevKitC-1-N8R8, tiene toda una familia de modulos y tarjetas muy similares entre si. La eleccion de esta tarjeta en particular responde a los siguientes criterios:

- Costo: Espressif ofrece en todos sus SoCs, modulos y tarjetas, un costo muy contenido por la gran cantidad de caracteristicas ofrecidas.
- Redes neuronales: la serie de SoCs ESP32-S3 ofrece soporte para instrucciones vectoriales, que acelera las tareas de computacion de redes neuronales. Esta fue la caracteristica mas importante al momento de la eleccion de esta tarjeta.
- Memoria: como el trabajo implicaba el uso de una camara y por tanto el manejo de *buffers* de memoria de gran tamano para manipular las imagenes obtenidas, la cantidad de memoria externa que ofrecia esta tarjeta la hizo optima para la aplicacion.

2.1.2. Sensor de movimiento PIR

Un sensor de movimiento PIR basa su funcionamiento al detectar diferencias en la energia IR (*Infrared*, Infrarojo) en el campo de vision del sensor. Debido a que la senal de salida generada por el sensor es muy pequena, es necesario aplicar etapas de amplificacion y filtrado para elevar el nivel de tension de la senal de salida y al mismo tiempo filtrar el ruido que puede generar eventos falsos positivos. Esta salida analogica luego se debe convertir en una senal digital mediante la operacion de comparacion de ventanas y se puede utilizar, por ejemplo, como una interrupcion en un MCU. En la figura 2.3 se muestra el diagrama en bloques del sensor de movimiento PIR.

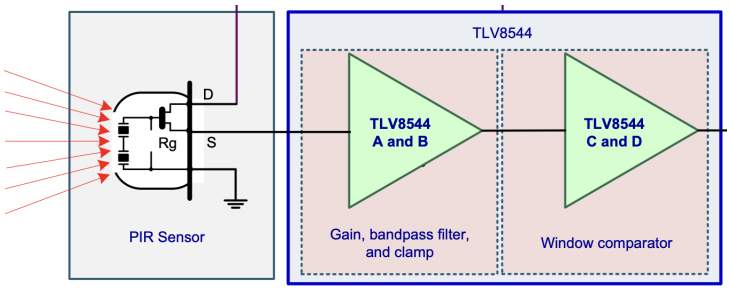

FIGURA 2.3. Diagrama en bloques del sensor de movimiento PIR.

Sensor IRA-S230ST01

Amplificador operacional TVL8544

2.1.3. Cámara OV2640

Otro de los componentes principales del sistema es la camara, que permite obtener imagenes en un formato digital que posteriormente deben ser procesadas por los algoritmos de DL. Para este trabajo se utilizo el modulo ESP-LyraP-CAM. Este modulo integra un CCM (*Compact Camera Module*, Modulo de Camara Compacto) con un sensor OV2640 en conjunto con dos reguladores de tension para su correcto funcionamiento. En la figura 2.4 se puede observar unas fotografias del modulo ESP-LyraP-CAM y sus componentes.

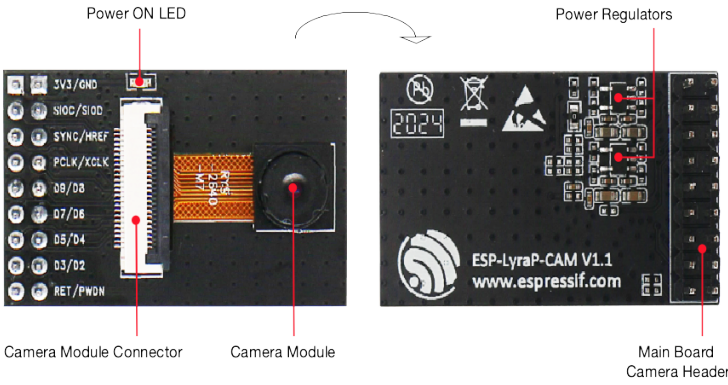

FIGURA 2.4. Diagrama en bloques del del modulo ESP-LyraP-CAM.

El OV2640 de la empresa OmniVision es un sensor CMOS (*Complementary Metal-Oxide-Semiconductor*, Semiconductor de Oxido de Metal Complementario) de 2

MP, cuenta con una interfaz de comunicacion compatible con DVP (*Digital Video Port*, Puerto de Video Digital), soporta codificacion JPEG (*Joint Photographic Experts Group*, Grupo Unido de Expertos en Fotografia) y es de bajo consumo energetico. En la tabla 2.1 se muestran las caracterisiticas tecnicas mas importantes del OV2640.

TABLA 2.1. Tabla de especificaciones del OV2640

Caracteristica	**Descripcion**
Tamano de matriz	1600x1200 (UXGA)
Fuente de alimentacion	*Core*: 1.3 V ± 5 % *Analog* 2.5 3.0 V I/O: 1.7 V - 3.3 V
Consumo energetico	*Free running*: 125 mW *Standby*: 600 µA
Formato de imagen del sensor	1/4″
Tasa de transferencia maxima	1600×1200 a 15 fps SVGA a 30 fps CIF a 60 fps
Sensibilidad	0.6 / Lux-sec
SNR	40 dB
Rango dinamico	50 dB
Tamano de pixel	2.2x2.2 µm
Formato de salida	YUV/RGB/MJPEG

Los criterios para utilizar este modulo como camara del sistema son los siguientes:

- Costo: los modulos con el sensor OV2640 tienen un costo muy reducido en comparacion con otros disponibles en el mercado.
- Bajo consumo energetico: como se mostro en la tabal 2.1 el consumo energetico del modulo en modo *standby* es lo suficientemente bajo como para funcionar alimentado por baterias.
- Disponibilidad de codigo: al ser un modulo que ya lleva mucho tiempo en el mercado existen muchas bibliotecas de codigo para utilizarlo, lo que simplifica en gran medida el tiempo de desarrollo de *firmware*.

2.2. MTCNN (*Multi-Task Cascaded Convolutional Networks*, Redes Convolucionales en Cascada Multitarea)

2.3. TensorFlow

2.4. AWS (*Amazon Web Services*, Servicios Web de Amazon)

2.5. Grafana

Capítulo 3

Diseño e implementación

3.1. Análisis del software

La idea de esta sección es resaltar los problemas encontrados, los criterios utilizados y la justificación de las decisiones que se hayan tomado.

Se puede agregar código o pseudocódigo dentro de un entorno lstlisting con el siguiente código:

```
\begin{lstlisting}[caption= "un epígrafe descriptivo"]
las líneas de código irían aquí...
\end{lstlisting}
```

A modo de ejemplo:

```
#define MAX_SENSOR_NUMBER 3
#define MAX_ALARM_NUMBER  6
#define MAX_ACTUATOR_NUMBER 6

uint32_t sensorValue[MAX_SENSOR_NUMBER];
FunctionalState alarmControl[MAX_ALARM_NUMBER]; //ENABLE or DISABLE
state_t alarmState[MAX_ALARM_NUMBER];           //ON or OFF
state_t actuatorState[MAX_ACTUATOR_NUMBER];     //ON or OFF

void vControl() {

  initGlobalVariables();

  period = 500 ms;

  while(1) {

    ticks = xTaskGetTickCount();

    updateSensors();

    updateAlarms();

    controlActuators();

    vTaskDelayUntil(&ticks, period);
  }
}
```

CÓDIGO 3.1. Pseudocódigo del lazo principal de control.

Capítulo 4

Ensayos y resultados

4.1. Pruebas funcionales del hardware

La idea de esta sección es explicar cómo se hicieron los ensayos, qué resultados se obtuvieron y analizarlos.

Capítulo 5

Conclusiones

5.1. Conclusiones generales

La idea de esta sección es resaltar cuáles son los principales aportes del trabajo realizado y cómo se podría continuar. Debe ser especialmente breve y concisa. Es buena idea usar un listado para enumerar los logros obtenidos.

Algunas preguntas que pueden servir para completar este capítulo:

- ¿Cuál es el grado de cumplimiento de los requerimientos?
- ¿Cuán fielmente se puedo seguir la planificación original (cronograma incluido)?
- ¿Se manifestó algunos de los riesgos identificados en la planificación? ¿Fue efectivo el plan de mitigación? ¿Se debió aplicar alguna otra acción no contemplada previamente?
- Si se debieron hacer modificaciones a lo planificado ¿Cuáles fueron las causas y los efectos?
- ¿Qué técnicas resultaron útiles para el desarrollo del proyecto y cuáles no tanto?

5.2. Próximos pasos

Acá se indica cómo se podría continuar el trabajo más adelante.