Dispositivo de sustitución sensorial para la mejora de la interacción social de personas con discapacidad visual

Trabajo de grado para optar al título de

Ingeniero en Mecatrónica

Trabajo de Grado

Autor:
Jasyd David Caballero Quintero

Tutor:
MSc. Juan Martín Cáceres Tarazona

4 de Diciembre de 2020

Formando **líderes** para la construcción de un nuevo **país en paz**

Dispositivo de sustitución sensorial para la mejora de la interacción social de personas con discapacidad visual

Trabajo de grado Ingeniería Mecatrónica

Autor
Jasyd David Caballero Quintero

Tutor
MSc. Juan Martín Cáceres Tarazona
Universidad de Pamplona

Trabajo de grado para optar al título de Ingeniero en Mecatrónica

Universidad de Pamplona
Facultad de Ingenierías y Arquitectura
Pamplona, Colombia
4 de Diciembre de 2020

Dedico este trabajo a...

A mi papá, Jaime Alberto Caballero Morantes y a mi mamá Sonia Janeth Quintero Mendoza quienes me dieron ánimos en todo momento y me apoyaron en mi camino para convertirme en quien soy ahora; siempre me brindaron su amor incondicional en los buenos y malos momentos y nunca me dejaron solo.

A mi hermana, Shadya Gabriela Caballero Quintero, por su cariño y estar para mi siempre.

A mis familiares y amigos, quienes siempre creyeron en mi, me aconsejaron y me apoyaron.

Agradecimientos

Agradezco a mis padres y a mi hermana, quienes siempre estuvieron para mi en cada momento, dándome razones y ánimos para seguir adelante y nunca rendirme.

A mis familiares, que con su apoyo, me ayudaron cuando mas lo necesité. Mis abuelos José María Quintero Escalante, Gladys Isabel Mendoza, mis tias Maruja Carillo Mendoza, su esposo e hijos, Luz Marina Caballero Morantes, Claudia Isabel Quintero Mendoza, su esposo e hijo, Liliana Amparo Quintero Mendoza e hijas, Sandra Milena Quintero Mendoza, su esposo e hijos. Mi tio José Rodolfo Quintero Mendoza, su esposa e hijos. Y especialmente a mi primo Mario Andrés Niño Carrillo y a mi prima Luz Evelyng Barroso Caballero, su esposo e hijas. A todos ellos mi cariño y profundo agradecimiento por su significativa presencia y apoyo incondicional.

Y a todos mis amigos por apoyarme e inspirarme con su valiosa compañía.

Pregúntate si lo que estás haciendo hoy te acerca al lugar en el que quieres estar mañana

-Walt Disney.

Resumen

La inclusión de las personas con discapacidad visual en tareas cotidianas, ha sido impulsada por distintos desarrollos tecnológicos, aumentando la independencia de estos individuos. Con el desarrollo de este proyecto se impulsó aún más dicha inclusión, dotando a los usuarios con la capacidad de identificar expresiones faciales en una conversación, aumentando su confianza al momento de formular una respuesta, ya que tendrán en cuenta esta información adicional, siendo que el lenguaje corporal representa un gran porcentaje en las interacciones sociales. El prototipo consta de unas gafas que estarán dotadas de una cámara en el frente, un auricular en el lado izquierdo y una mini-computadora para realizar todo el procesamiento requerido. Las imágenes son tomadas por la cámara en condiciones apropiadas de luz, analizadas con redes neuronales convolucionales (CNN) para detectar la posición del rostro y estimar las expresiones faciales provenientes del interlocutor, a partir de la emoción detectada se realiza la retroalimentación sonora al auricular con una pista de audio que le permita al usuario conocer esta información adicional en su conversación.

Palabras clave: discapacidad visual, dispositivo de sustitución sensorial, visión artificial, realimentación acústica o sonora, interacción social, expresiones faciales, redes neuronales convolucionales.

Abstract

The inclusion of people with visual disability in daily tasks, has been boosted with some technologic developments, increasing the independency od these individuals. With the development of this proyect, the inclusion has been boosted more, endowing the users with the ability to identify facial expressions in a conversation, giving more confidence at the moment of giving an answer, having this additional information in consideration, since the body language represents a whole porcentage in social interaction. The prototype consists in a pair of glasses, with a camera in the front, a headphone in the left side and a tiny computer that is responsible of doing all the required processing. The pictures are took with the camera in appropriate light conditions, analyzed with convolutional neural networks (CNN) to detect the position of a face and estimate the facial expresions from the interlocutor. Depending on the detected emotion, the audio feedback is realized with an audio track, allowing the user to know this additional information in his conversation.

Keywords: Visual disability, Sensorial Substitution Device, artificial vision, audio feedback, social interaction, facial expressions, convolutional neural networks

Índice general

Índice de figuras

Índice de tablas

Índice de Códigos

1 Introducción

Los humanos somos seres sociales por naturaleza, por lo que la interacción social es un componente indispensable para el desarrollo de la personalidad y la identidad del individuo. Las personas con mas relaciones interpersonales presentan niveles reducidos de depresión y ansiedad.

La interacción social tiene dos componentes principales, el lenguaje hablado y el lenguaje corporal, siendo el último a base de señas, gestos y expresiones faciales el que representa mas del 93% de una conversación.

Los individuos con disminución total o parcial de la visión presentan dificultades en el desarrollo normal de sus habilidades sociales, mas notables cuando esta condición aparece en el nacimiento. La inhabilidad de percibir y entender el estado mental y las emociones a través del lenguaje corporal disminuye la posibilidad de una comunicación efectiva.

Para dar solución a este inconveniente, durante los últimos años se han desarrollado los denominados Dispositivos de Sustitución Sensorial (SSD), los cuales buscan a través de retroalimentaciones hápticas o sonoras, facilitar la vida a individuos con discapacidad visual, dotándolos con la habilidad de navegar por si solos, tener conversaciones mas fluidas, inclusión en la educación, entre otros.

En este trabajo se presenta el desarrollo de uno de estos dispositivos, con el fin de brindar al usuario la habilidad de identificar las expresiones faciales que presente su interlocutor en una conversación. El usuario es notificado a través de una retroalimentación sonora de la expresión detectada, esto por medio de un sistema de visión artificial montado sobre unas gafas de sol de una manera compacta. Los sistemas desarrollados anteriormente requieren de otros periféricos y cables para cumplir su propósito debido a que no existían sistemas embebidos lo suficientemente compactos para realizar tareas complejas como lo es la visión artificial. Gracias a los avances en la tecnología, hoy en día existen sistemas de bajo consumo y de tamaño reducido que pueden desarrollar dichas tareas con la eficiencia suficiente para que funcione de manera adecuada.

1.1 Justificación

Los esfuerzos por incluir a las personas con discapacidad visual en tareas de la vida cotidiana, han estado enfocados principalmente en la movilización, mediante dispositivos que guíen al individuo usando cualquier técnica de retroalimentación acústica o háptica, disminuyendo la necesidad de un acompañante, como también dispositivos que permiten al individuo tener un mayor desempeño y facilidad en el área laboral o educacional. Pero al momento de comunicarse con otra persona presentan desventajas, ya que no tienen manera de conocer e identificar el lenguaje corporal de su interlocutor, especialmente de las expresiones que realiza con su rostro. Estas acciones representan el 93% de una conversación y son altamente influyentes en el hecho de poder conocer las emociones e intenciones de la persona. Por su contra parte el lenguaje hablado solo representa el 7%. Sin esta información adicional, el usuario no tendrá la capacidad de devolver una respuesta completamente asertiva, lo cual influye negativamente en su confianza y su habilidad para entablar una conversación.

Han habido desarrollos como por ejemplo “Un asistente social para ayudar a las personas con discapacidad visual durante la interacción social: un análisis de requisitos sistemático”, que han tratado de solventar esta problemática; teniendo en cuenta estos trabajos como punto de partida, se desarrolló un sistema capaz de notificar al usuario de las expresiones faciales o gestos que presente su interlocutor durante una conversación. Imágenes tomadas de una cámara montada en unas gafas de sol modificadas, serán procesadas mediante técnicas de inteligencia artificial como las redes neuronales convolucionales, para la estimación de las expresiones faciales que haga la otra persona. Además, usando retroalimentación acústica o sonora, el usuario tendrá la capacidad de reconocer el gesto o emoción identificado.

1.2 Objetivos

1.2.1 Objetivo General

- Desarrollar un dispositivo de sustitución sensorial para la mejora de la interacción social de personas con discapacidad visual.

1.2.2 Objetivos específicos

- Diseñar el sistema de adquisición y procesamiento de imágenes.
- Implementar los algoritmos de procesamiento de imágenes en el sistema embebido.
- Desarrollar el sistema de realimentación acústica o sonora.
- Validar el funcionamiento del sistema de manera experimental

2 Marco Teórico

2.1 Discapacidad visual

La discapacidad visual se define como la dificultad que presentan algunas personas para participar en actividades cotidianas, que surge como consecuencia de la disminución, carencia o pérdida de las funciones visuales de manera congénita o adquirida. La misma es causada por factores como enfermedades y trastornos que afectan directamente a la visión. Estas limitaciones pueden ser totales en el caso de la ceguera o parciales como es el caso de la baja visión. [1]

Por lo general, una persona a lo largo de su vida experimenta al menos una enfermedad ocular, en todo el mundo, por lo menos 2.200 millones de personas padecen deficiencia visual o ceguera. Los términos de visión parcial, visión defectuosa, debilidad visual, visión subnormal y baja visión, son usados para clasificar el deterioro visual entre la visión normal y la ausencia de función visual o ceguera. el concepto de ceguera ha sufrido cambios en el marco legal, político y laboral.

Promovido por la OMS (Organización Mundial de la Salud) desde el año 2009, la ceguera corresponde a las categorías 3, 4, y 5 de severidad de la discapacidad visual (ver Tabla 2.1) indicando un porcentaje de agudeza visual menor al 5% disminuyendo hasta la no percepción de luz. Es importante enmarcar esta diferenciación, la mayoría de personas clasificadas como ciegas, presentan un resto visual que les permite de una u otra forma, desenvolverse y realizar actividades diarias. [2]

2.1.1 Retos para las personas con discapacidad visual

La pérdida total o parcial del sentido de la vista representa cambios a nivel personal, laboral y social. Las actividades y responsabilidades que una persona sana desarrollaba de forma automática, cambian convirtiéndose en dificultades que pueden provocar miedos, ansiedad o depresión.

La adaptación de estos individuos a estas situaciones es tarea compleja y varía para cada uno de ellos, se debe tener en cuenta aspectos personales como la edad, la salud o la historia biográfica, como otros relativos a su entorno social. [3]

Categoría	Agudeza visual menor a:	Agudeza visual mayor a:
0: discapacidad visual leve o sin discapacidad	No aplica	6/18 3/10 (0.3) 20/60
1: discapacidad visual moderada	6/18(metros) 3/10(0.3) 20/60(pies)	6/60(metros) 1/10(0.1) 20/400(pies)
2: discapacidad visual severa	6/60(metros) 1/10(0.1) 20/200(pies)	3/60(metros) 1/20(0.05) 20/400(pies)
3: ceguera	3/60(metros) 1/20(0.05) 20/400(pies)	1/60(metros) 1/50(0.02) 5/300(pies)
4: ceguera	1/60(metros) 1/50(0.02) 5/300(pies)	Percepción de luz
5: ceguera	No percepción de luz	
9	Indeterminado o no especificado	

Tabla 2.1: Categorías de discapacidad visual OMS.
Fuente: Escudero and Camilo, 2011

Dificultad en el aprendizaje

La inclusión de personas con discapacidad visual al área educativa es una tarea compleja, adquirir aprendizaje de calidad requiere de transformaciones técnicas, culturales y sociales, especialmente en entornos de educación superior, las universidades presentan problemas para que personas con discapacidad visual puedan ser incluidas académicamente.

Las posibilidades de aprendizaje y comunicación mejoraron a partir del año 1829, año en el que se publica un volumen, impreso en relieve lineal, donde se daba a conocer el sistema elaborado por Louis Braille, sistema de lectoescritura que lleva su nombre. Este método integra todas las letras, acentos, signos de puntuación y signos matemáticos utilizando solo 6 puntos y algunas rayas horizontales que mas adelante eliminaría. Debido a su versatilidad, se usó para reproducir todas las lenguas además de que con él era fácil adaptar las matemáticas, la música, etc. [4]

Desplazamiento autónomo

El desplazamiento autónomo es otro de los principales retos que enfrentan estos individuos, que afecta de forma indirecta a sus actividades de vida diaria, tales como

ir a estudiar o a trabajar, dirigirse a una consulta médica, entre otros. Enfrentan problemas como cambios en las rutas debido a una construcción o al clima, falta de herramientas de navegación o incluso indicaciones inútiles de personas en su entorno.[5]

Con el paso de los años, se han presentado herramientas de navegación para estos individuos, incluyendo dispositivos GPS pequeños y precisos, perros guía, aplicaciones para teléfonos inteligentes, entre otras, que permiten un desplazamiento seguro y de forma independiente. La primera herramienta que cambió el panorama por completo fue el bastón largo, construido con tubo de aluminio usado en aviones militares y diseñado por Richard E. Hoover en 1940, que hasta la fecha sigue siendo el dispositivo de uso básico para estas personas. Hoy en día las herramientas de alta tecnología asisten con pre-planeación del viaje, identificación de puntos de interés y anuncios de las direcciones que debe seguir el usuario para llegar a su destino. [6]

Interacción social

El desarrollo del individuo depende en gran medida de la vida en sociedad, ya que la comunicación e interacción con los demás le permiten aprender pautas y normas sociales de convivencia de la cultura en la que habita. Es el caso en el que las personas con discapacidad visual presentan desventajas, además de que las personas videntes sienten recelo al tratar con ellos, creando la dificultad en el normal desarrollo de las relaciones sociales. [7]

Los niños desde temprana edad obtienen información acerca del comportamiento social a través de observación e imitación. La inhabilidad de los niños con discapacidad visual para observar e imitar el comportamiento social, afecta significativamente al desarrollo de sus habilidades sociales en todos los aspectos y a su comportamiento. Obtener habilidades sociales es tan importante como el desarrollo de otras habilidades. Los padres de estos niños se dan cuenta de que sin buenas habilidades sociales y un buen comportamiento, sus niños enfrentarán dificultades adaptándose a diferentes situaciones, llegando incluso a experimentar aislamiento por parte de sus compañeros. [8]

2.1.2 Confianza social de las personas con ceguera

Existe evidencia consistente sugiriendo que la presencia de ceguera a temprana edad puede interferir en el desarrollo normal de las habilidades sociales como la de tomar cierta perspectiva además de la comprensión de estados mentales y emociones de otros y tener un efecto dramático en la evaluación de factores sociales como la integridad. Los individuos que presentan privación de la visión a temprana edad, experimentan interferencias en el normal desarrollo de las habilidades sociales, que no presentan sus compañeros videntes, sin embargo, las dificultades en interacciones sociales van disminuyendo a medida que desarrollan habilidades verbales, aumentando la percepción

del estado mental de otros. A pesar de lo mencionado anteriormente, una gran parte de la información relevante en una conversación se transmite mediante señales no verbales, las personas con mínima o nula discapacidad visual, confían principalmente en expresiones faciales o gestos corporales cuando requieren evaluar el estado emocional de otras personas, similar a los gestos realizados con la mirada, que proveen información crítica acerca de las intenciones y objetivos de las personas. En este sentido, aunque las persona con visión limitada puedan hacer inferencias sobre el estado mental de su interlocutor, la falta de señales visuales derivadas de las expresiones faciales o los gestos corporales probablemente tengan un impacto en la interpretación de las intenciones, creencias y sentimientos de una persona. [9]

2.2 Dispositivos de sustitución sensorial

Existe la posibilidad de compensar la pérdida de funciones sensoriales transmitiendo esta información faltante a través de otro sentido intacto, técnicas desarrolladas para la sustitución sensorial para personas ciegas a través del tacto y la audición, con la característica de que no se necesita un entrenamiento riguroso para que el individuo pueda adaptarse a su uso.

Una de las dificultades en el desarrollo de estos dispositivos puede ser el filtrado y selección de la información adicional a entregar en el sentido seleccionado, es necesario entregar solo información importante del entorno, para evitar una sobrecarga sensorial y redundancias; para esto se realizan investigaciones psicológicas y neuro-científicas en representaciones del entorno y los caminos más efectivos para la entrega de la información, obteniendo mejores resultados en el diseño de sistemas de sustitución sensorial.

Dichos dispositivos deben enfatizar en su usabilidad, y no interferir con otros funciones perpetúales, por esto su desarrollo de enfoca en el desarrollo de una tarea específica por lo que puede ser impráctica la entrega de detalles acerca de demasiados aspectos del entorno.

El desarrollo de estos dispositivos debe tener en cuenta ciertas consideraciones, reemplazar la vista usando otro sentido tiene sus límites, ya que el nervio auditivo cuenta con aproximadamente 30.000 fibras, mientras que el nervio óptico cuenta con más de un millón de fibras, además, medidas psicológicas muestran que la capacidad de obtención de información por parte de la visión es considerablemente más alta que la audición. Investigaciones acerca de la capacidad del sistema visual demuestran que este tiene 4 veces más ancho de banda que la percepción háptica, mientras que la capacidad de adquisición de información por parte del oído humano cae entre estas dos estimaciones.

La falta de visión limita la posibilidad de movilización independiente, para esto ha habido numerosos intentos que han generado estrategias donde se obtiene información

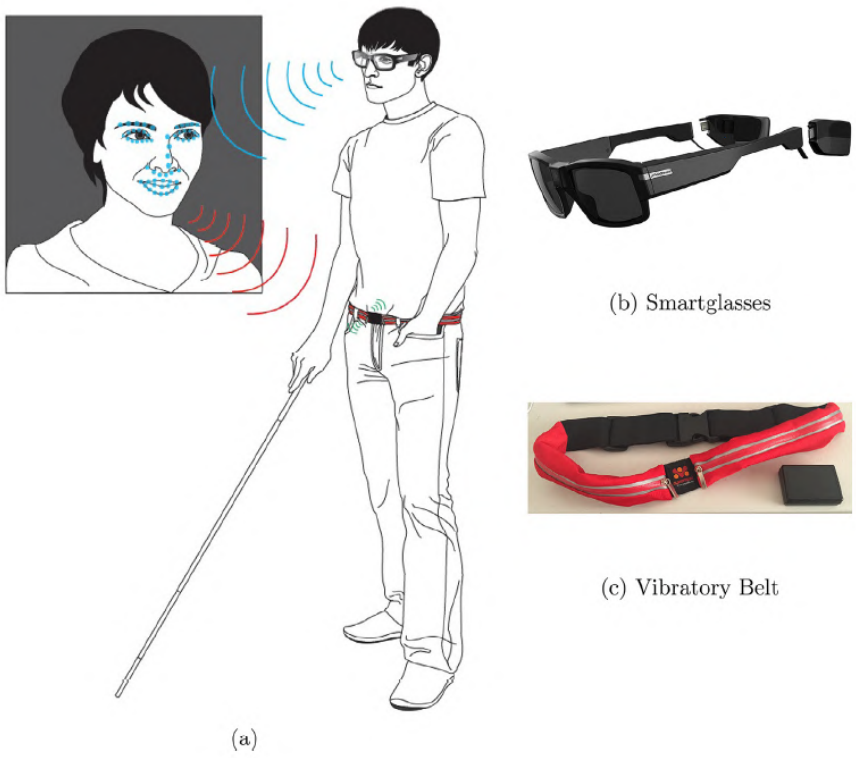

Figura 2.1: Sistema de sustitución sensorial desarrollado en "Un asistente social para ayudar a las personas con discapacidad visual durante la interacción social: un análisis de requisitos sistemático". Las gafas mostradas en (b) transmiten video a un computador, que procesa los datos y envía comandos al cinturón vibratorio mostrado en (c). El cinturón recibe los comandos del computador y vibra, dando al usuario mostrado en (a) la retroalimentación

Fuente: Park et al.

del entorno, y se entrega a través de la audición o el tacto, dotando al usuario de la capacidad de evadir obstáculos, análisis del escenario y reconocimiento de objetos[11].

Un ejemplo de estos dispositivos se evidencia en la figura 2.1, en el cual se usan unas gafas dotadas de una cámara, de la cual se toman las imágenes y se procesan para estimar la orientación de la cabeza del interlocutor, con esto, cuando se detecta que la persona asiente con la cabeza, se realiza una retroalimentación háptica por medio de un cinturón, indicando al usuario del gesto para que de una respuesta [10].

2.2.1 Consideraciones a tener en cuenta en el desarrollo de dispositivos de sustitución sensorial

- **Necesidad de una investigación básica:** para el estudio y generación de estos dispositivos, es necesario una investigación básica acerca de la cognición y

la percepción; resolver dudas acerca de cómo los sonidos y los estímulos al tacto son interpretados y qué tipo de estimulación es la más efectiva para entregar un dato acerca del entorno a un sentido en particular deben ser considerados con suficiente detalle. La respuesta a estas incógnitas es obtenida eficientemente a través de la experimentación psicológica.

- **Comodidad y facilidad de uso:** la comodidad, la facilidad de uso, la movilidad y la apariencia son aspectos a considerar si se espera que el usuario use el dispositivo en particular, estos deben dejar las manos libres o minimizar su uso en la operación del mismo, tampoco deben interferir con la habilidad de sentir el entorno directamente, evitando la sobrecarga sensorial; también deben brindar una fácil usabilidad y no restringir la movilidad, fácil de usar sin la inclusión de características innecesarias y más importante, no deben requerir de un exhaustivo entrenamiento para aprender a usar el dispositivo

- **Externalización:** en el desarrollo de un dispositivo de sustitución sensorial, el usuario debería poder llegar a sentir externalización al usar el dispositivo, es decir, que el dispositivo pueda experimentarse como una extensión del cuerpo, por ejemplo en la problemática de la representación del entorno al individuo, que este sea capaz de construir una representación mental de las sensaciones que experimenta a través de la estimulación táctil o auditiva.

2.2.2 Dispositivos de sustitución sensorial para la visión a través de la audición

Han habido intentos en el desarrollo de dispositivos de sustitución sensorial, para entregar información del ambiente al usuario a través de estimulación auditiva. Algunos usando eco localización como lo hacen Ifukube, Sasaki y Peng en [12], pero debido a la contaminación acústica, este desarrollo se vuelve impráctico, previniendo el uso del dispositivo. En cambio, los desarrollos basados en el uso de cámaras y retroalimentación sonora pueden ser más prácticos.

Una consideración importante es la manera en la que se entrega la información al usuario, con sonidos o comandos de voz. Loomis, Golledge y Klatzky mostraron en [13] que se obtiene un mejor rendimiento cuando se usan sonidos virtuales que comandos verbales. Väljamäe y Kleiner sugieren en [14] que una manera práctica de entregar información puede ser generando sonidos que correspondan a otra dimensión secuencial. Por ejemplo el brillo puede ser representado con volumen, tamaño con agudeza, dirección con timbre, etc [11].

2.3 Inteligencia artificial (AI)

La inteligencia artificial es la rama de las ciencias de la computación que lidia con el estudio y el diseño de agentes inteligentes que perciben su entorno y toman decisiones para maximizar sus posibilidades de éxito [15]. Las características que poseen estos agentes son las que nosotros asociamos con el comportamiento inteligente de los seres humanos, características tales como percepción, procesamiento del lenguaje natural, resolución de problemas y planeación, aprendizaje y adaptación al entorno [16].

El machine learning es una de las áreas de estudio de la inteligencia artificial, área la cual ha estado en constante crecimiento a lo largo de los últimos años, usado comúnmente en tareas que requieren extraer información de grandes sets de datos. [17] Su uso consiste principalmente en dos pasos, la primera fase donde se usan datos de entrada (como por ejemplo imágenes de gatos y perros para realizar una tarea de clasificación) para encontrar los parámetros que mejor solucionan la problemática, y la segunda fase, donde se usan los parámetros encontrados para realizar la tarea de clasificación, fase llamada también ”inferencia” [18].

El deep learning es un subcampo del machine learning que basa sus modelos en componentes básicos llamados ”neuronas”, las cuales están inspirados en las neuronas biológicas. Estas están organizadas en capas sucesivas y conectadas entre si, conexiones a las cuales se les atribuye un peso que es ajustado en la fase de aprendizaje. Cada neurona mapea su entrada a una salida con una función de transferencia, simulando el comportamiento de las neuronas del cerebro.

Después de la fase de aprendizaje, la red neuronal es capaz de separar los datos de entrada en jerarquías de características, representando múltiples capas de abstracción. Un ejemplo es el reconocimiento de rostros, la primera capa identifica patrones elementales como lineas, bordes y esquinas; y las capas siguientes se encargan de encontrar patrones mas grandes como labios, cejas y ojos. Existen dos tipos de redes neuronales especialmente conocidos, las redes neuronales convolucionales (especiales para tareas de visión computacional) y las redes neuronales recurrentes (usadas en aplicaciones de procesamiento de lenguaje) [18].

2.3.1 Historia de las redes neuronales artificiales

Sus inicios se remontan a 1957, cuando Frank Rosenblatt comenzó sus desarrollos en el perceptrón, la red neuronal más antigua. Un modelo capaz de predecir patrones similares a los cuales se les había presentado en su entrenamiento, pero al ser un modelo tan simple, no era capaz de resolver problemas no linealmente separables como la función XOR (OR-exclusiva).

Luego en 1960, Bernand Widroff y Marcian Hoff hicieron la primera implementación exitosa de una red neuronal a un problema de la vida real con su modelo Adaline

(ADAptative LINear Elements), usado como filtro para la eliminación de ecos en líneas telefónicas. Las redes neuronales artificiales tuvieron un fuerte declive en 1969 cuando Marvin Minsky y Seymour Papert probaron matemáticamente que el perceptrón no era capaz de resolver problemas no linealmente separables, los cuales eran relativamente fáciles, demostrando que el perceptrón era débil dado que la no-linealidad está ampliamente presente en problemas de la vida real.

El renacimiento de las neuro-redes lo trajeron John Hopfield en 1985 con su libro "Computación neuronal de decisiones en problemas de optimización" y David Rumelhart junto con G. Hinton, quienes redescubrieron el algoritmo de propagación hacia atrás (backpropagation) usado hasta el día de hoy para el entrenamiento de redes neuronales [19].

2.3.2 Redes neuronales artificiales

"Las redes neuronales artificiales (RNA) surgen como un intento para emular el funcionamiento de las neuronas de nuestro cerebro" [20]. Estas neuro-redes están enfocadas en modelar la forma de procesamiento de la información en sistemas nerviosos biológicos, principalmente basándose en el funcionamiento del cerebro humano, el cual es un sistema altamente complejo, capaz de realizar muchas operaciones de forma paralela, a diferencia de las computadoras convencionales, las cuales son de tipo secuencial o una operación a la vez.

Una red neuronal se puede describir como un procesador de información, constituido por unidades sencillas e interconectadas llamadas neuronas, además es capaz de almacenar conocimiento a través de la experiencia, tal como lo hace el cerebro humano, y tener un comportamiento no lineal, teniendo la capacidad de procesar información que presente una no-linealidad.

Las neuronas artificiales reciben entradas las cuales son multiplicadas por los pesos asociados a cada conexión, dichos pesos indican la importancia de cada entrada de la neurona, posteriormente se realiza la suma de todas estas multiplicaciones, el resultado es evaluado en una función de activación la cual puede ser del tipo escalón, sigmoidal, entre otras, siendo el valor devuelto por esta función, la salida de la neurona [21].

Dentro del cerebro se presenta una red de células (neuronas) las cuales presentan una gran cantidad de conexiones entre sí, la información que recibe cada uno de estos componentes a través de las dendritas, pasa a través del soma el cuales el órgano de cómputo y va al axón, para transmitir la información de salida a otras unidades a las que esté conectada, se estima que en el cerebro hay alrededor de cien mil millones de neuronas.

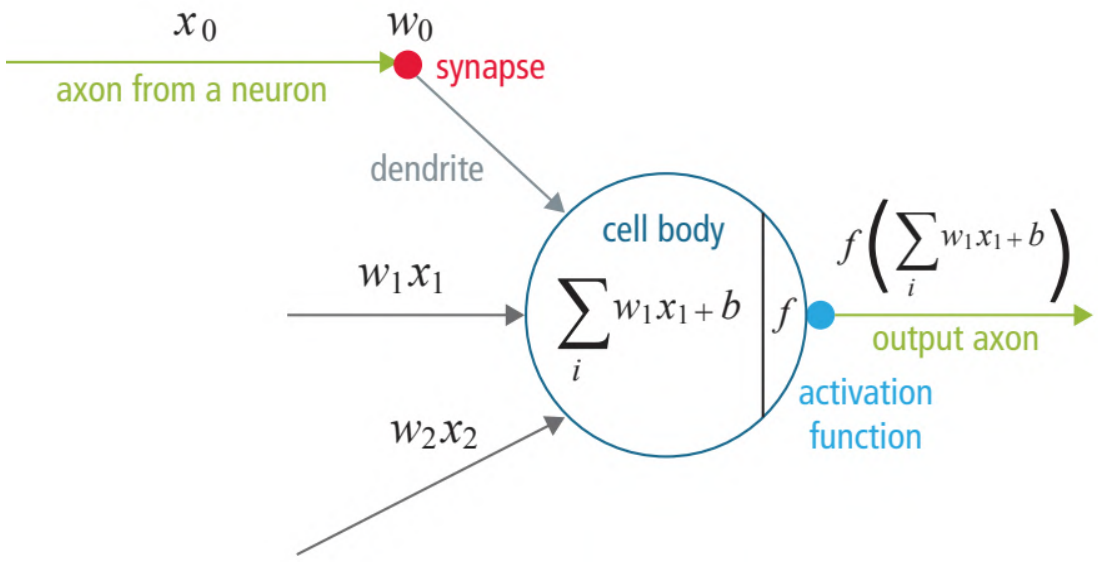

Figura 2.2: Esquema de una neurona artificial.
Fuente: Pigou et al.

La unión entre 2 neuronas o más recibe el nombre de sinapsis, esta conexión es unidireccional. Cada neurona recibe impulsos eléctricos a través de las dendritas y estás están a su vez conectadas a otras para producir la sinapsis [23]

Estructura de una red neuronal artificial

Como se ha mencionado anteriormente, una red neuronal artificial está compuesta por elementos simples, llamados neuronas; tenemos varias de estas unidades ordenadas por capas, la capa de entrada, la cual no realiza ningún proceso, su única función es entregar los valores de entrada a los siguientes nodos; las capas ocultas, reciben los valores de entrada y se ocupan de proporcionar mayor complejidad a la red, permitiendo un mejor aprendizaje, éstas capas pueden o no estar presentes en una red, dependiendo de la topología escogida; por último la capa de salida de encarga de proporcionar la salida del sistema.

Ventajas de las redes neuronales artificiales

- Cada una de las neuronas realiza un procesamiento, dependiendo de las entradas y pesos asociados a estas, además de la función de activación asignada. Dicho procesamiento se realiza de manera paralela y proporcionan una respuesta al mismo tiempo.

- Los pesos sinápticos son ajustados usando reglas de aprendizaje como el algo-

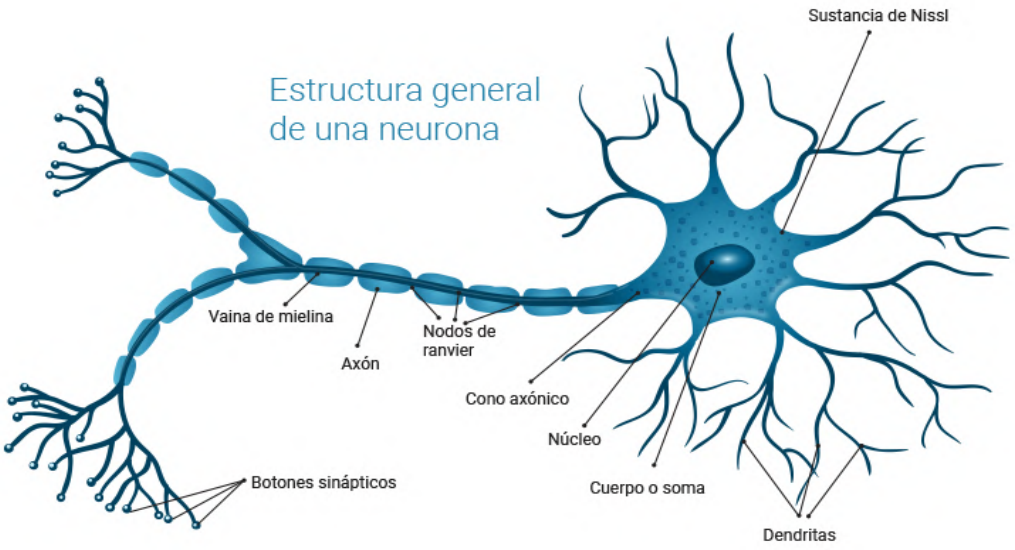

Figura 2.3: Estructura de una neurona biológica.
Fuente: Campos

ritmo de backpropagation, enseñando a la red lo que necesita para funcionar correctamente.

- Las redes neuronales son tolerantes a fallos, pueden seguir operando si parte de la red deja de funcionar, solo dejará de funcionar para los patrones en los que dicha región desempeñaba un papel importante.

- Las redes neuronales tienen la capacidad de predecir patrones que no han sido mostrados anteriormente (patrones que no fueron dados a la red en el periodo de entrenamiento). El único requisito es que el nuevo patrón sea similar a los de entrenamiento.

- La velocidad de respuesta de las redes neuronales es casi inmediata una vez que han sido entrenadas [25]

2.3.3 Redes Neuronales Convolucionales (CNN)

Las redes neuronales convolucionales han tenido gran protagonismo en la última década en gran variedad de campos relacionados con el reconocimiento de patrones, desde reconocimiento de imágenes hasta reconocimiento de voz. Debido al reducido número de parámetros requeridos, los desarrolladores se han enfocado en aumentar el tamaño de los modelos para dar solución a tareas complejas que no eran posibles con las redes neuronales artificiales clásicas.

Otro aspecto importante acerca de las CNNs es obtener características abstractas cuando la entrada se propaga a las capas mas profundas. Por ejemplo, en la clasificación de imágenes, los bordes son detectados por las primeras capas, las formas simples son detectadas en las capas secundarias y las características de alto nivel como las caras, son detectadas en las capas mas profundas como se puede ver en la figura 2.4[26].

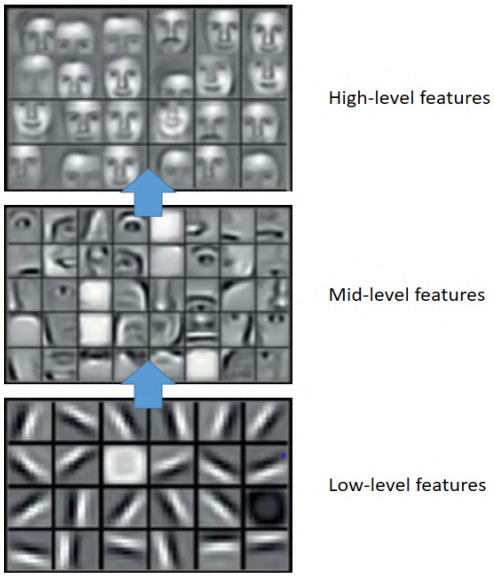

Figura 2.4: Características aprendidas de una red neuronal convolucional.
Fuente: Albawi et al.

2.3.4 Estructura de una CNN

La arquitectura de una CNN está compuesta por varios bloques como capas de convolución (convolutional layers), capas de agrupación (pooling layers) y capas conectadas. Típicamente una CNN consiste en la repetición de bloques con varias capas de convolución y capas de agrupación seguidos de una o mas capas conectadas [27].

Convolutional layer

Las capas de convolución son los elementos principales en una CNN que desarrolla la extracción de patrones, que consiste en una combinación de operaciones lineales y no lineales, que son las convoluciones y las funciones de activación.

Una Convolución es un tipo de operación lineal usada para extracción de características, donde un arreglo de números llamado kernel, es aplicado a la entrada, que es un arreglo de números llamado tensor. El valor de una posición dada del tensor de

salida, es obtenido mediante la suma de los productos de cada uno de los elementos del kernel y cada una de las posiciones del tensor de entrada, a este tensor de salida se le llama mapa de características (ver figura 2.5). El proceso se repite múltiples veces para obtener varios mapas de características, difiriendo cada uno por el kernel usado, estos varian de tamaño, los mas comunes son 3×3, 5×5 y 7×7. La operación de convolución está definida por el tamaño del kernel y la profundidad del mapa de características [27].

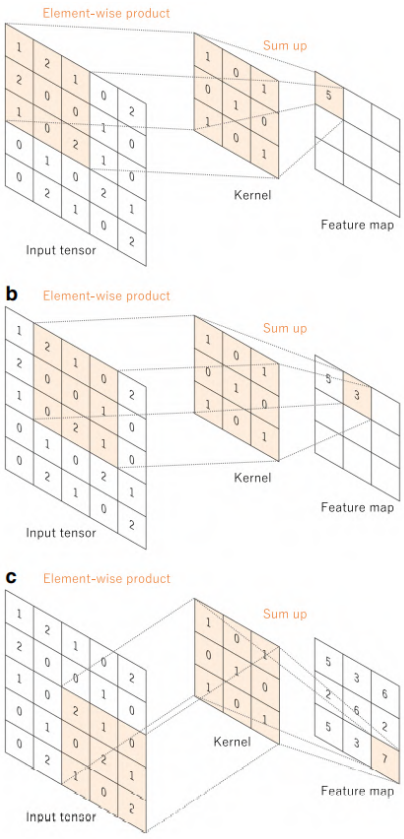

Figura 2.5: Ejemplo de una operación de convolución con un kernel de tamaño 3×3, con desplazamiento de valor 1.

Fuente: Yamashita et al.

Non-linear layers

Las salidas de operaciones lineales como las convoluciones son luego pasadas funciones de activación no lineales. La capa ReLU es una de las mas usadas ya que tiene la ventaja de entrenar las CNN mucho más rápido. En ReLU se implementa la funcion $y = max(x, 0)$, el tamaño de entrada y salida son el mismo. Esta incrementa las propiedades no lineales de la función de decisión y de la red en general sin afectar los

campos receptivos de la capa convolucional [22].

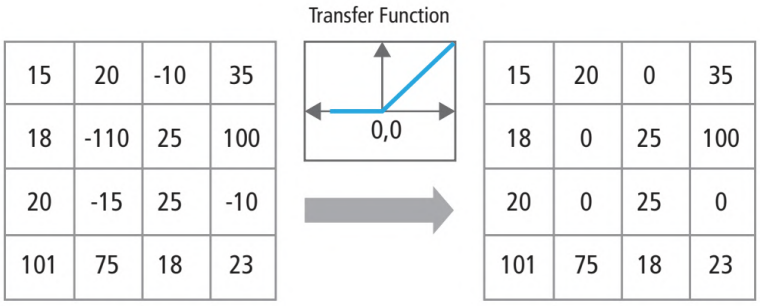

Figura 2.6: Representación de la funcionalidad de ReLU.
Fuente: Pigou et al.

Pooling layers

Las capas de agrupación (Pooling layers), se encargan de reducir la resolución de los mapas de características. haciéndolos robustos frente a ruido y distorsión. Hay dos tipos de estas capas, agrupación máxima (Max pooling) y agrupación promedio (Average pooling). La figura 2.7 muestra el proceso de agrupación que realizan estos dos tipo de capas, la entrada tiene un tamaño de 4 ×4, y la salida en ambos casos es de 2×2. La entrada es dividida en matrices de tamaño 2 ×2. En el caso de max pooling, se toma cada una de las subdivisiones de la entrada y se toma el valor máximo de los 4 valores y en el caso de average pooling, el resultado es el promedio entre los 4 valores del arreglo [22]

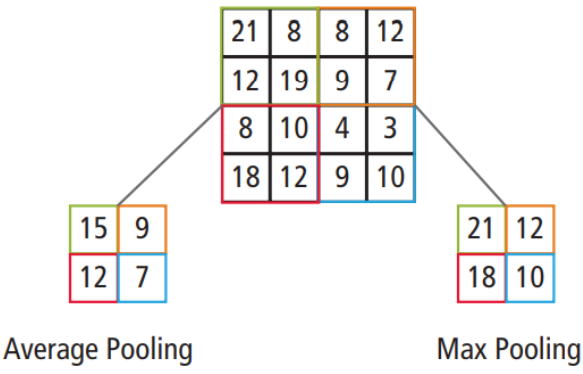

Figura 2.7: Representación del max pooling y el average pooling.
Fuente: Pigou et al.

Fully-connected layers

Las capas completamente conectadas (Fully connected layers) son la última estación en la topología de una CC, consiste en un arreglo multicapa de neuronas (ver figura 2.8) [28]. Esta capa tiene conexiones con todas las activaciones de la capa previa. Sus activaciones se computan con operaciones de matrices teniendo en cuenta el desplazamiento "bias" [29].

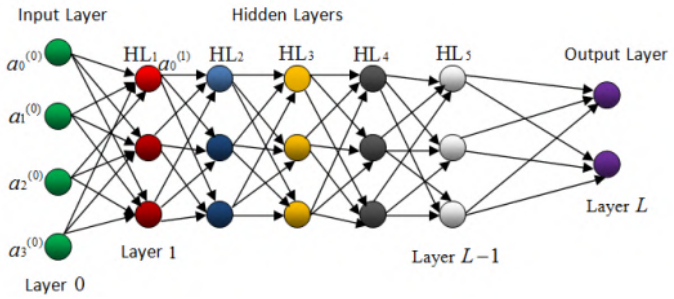

Figura 2.8: Fully-connected layer
Fuente: Sakib et al.

2.3.5 MobileNet

La MobileNet es un modelo creado especialmente para aplicaciones de visión artificial para teléfonos móviles y sistemas embebidos, por lo tanto el modelo no requiere de altas prestaciones en cuanto a hardware para realizar los cálculos de forma rápida y en tiempo real.

La arquitectura usa convoluciones separables en profundidad el cual es un tipo de convolución mas rápida que las usada en otras arquitecturas.

MobileNet puede ser usada en varias tareas de reconocimiento de imágenes tales como reconocimiento de puntos de referencia, detección de objetos (MobileNet puede clasificar hasta 1000 objetos) y reconocimiento de rostros[30].

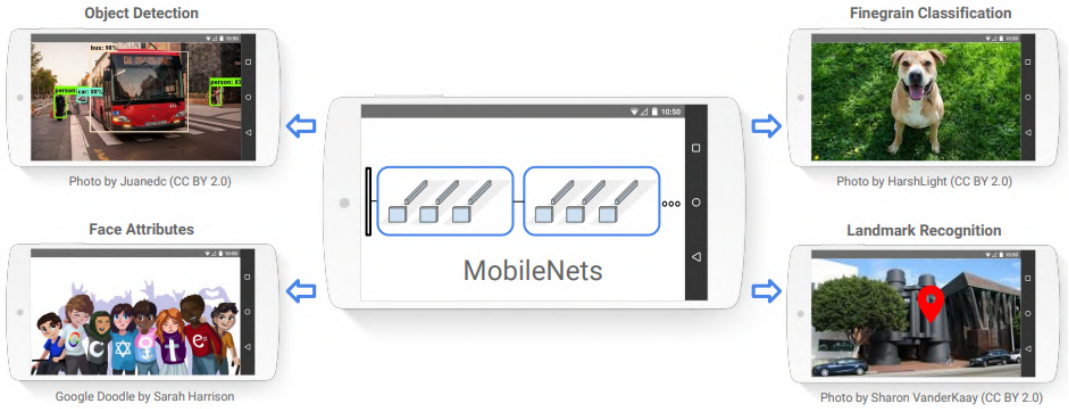

Figura 2.9: El modelo MobileNet puede ser aplicado a varias tareas de reconocimiento de forma eficiente.

Fuente: Howard et al.

2.3.6 You Only Look Once (YOLO)

YOLO es una arquitectura capaz de detectar objetos de distintas clases en una imagen "viéndola" una sola vez, lo que le permite ser mas rápida que otras arquitecturas desarrolladas anteriormente. La rapidez de YOLO le permite detectar objetos en tiempo real, llegando hasta los 30 FPS a un pequeño costo de exactitud.

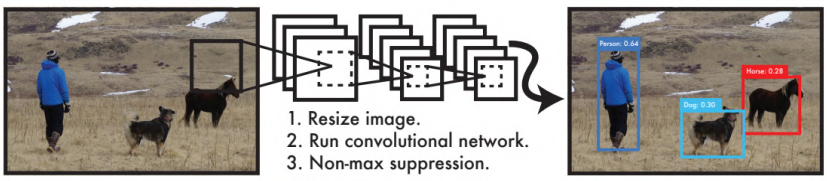

Figura 2.10: El sistema de detección YOLO.

Fuente: Redmon

La detección de objetos se lleva a cabo dividiendo la imagen de entrada con una cuadrícula de cierto tamaño S. Posteriormente, para cada una de las celdas se predicen N cuadros delimitadores y se calcula la probabilidad para cada uno de ellos. Después se eliminan los cuadros que están por debajo de un límite de probabilidad definido y de los restantes se toman los que tengan la máxima probabilidad con el fin de eliminar varios cuadros que estén sobre el mismo elemento, dejando el que lo encierre de la manera mas exacta. El proceso puede visualizarse de forma gráfica con la figura 2.11[31].

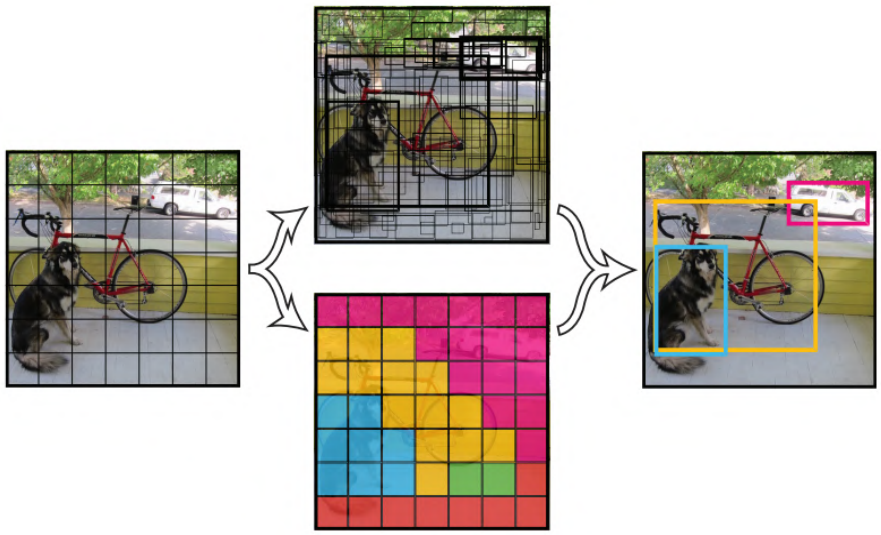

Figura 2.11: Imagen con grilla de entrada, cuadros delimitadores, mapa de probabilidades y detección del algoritmo YOLO.

Fuente: Redmon

3 Metodología

3.1 Componentes

En esta sección se describen los componentes electrónicos seleccionados para el desarrollo del trabajo, teniendo en cuenta principalmente su tamaño, características técnicas y precio.

3.1.1 Placas de desarrollo

Existen distintas placas de desarrollo disponibles en el mercado. Para este proyecto se requiere una de tamaño reducido y que sea capaz de desarrollar tareas de visión artificial, con el fin de obtener un prototipo compacto, por lo que se tuvieron en cuenta las siguientes placas.

Característica	**Raspberry pi zero w**	**Sipeed M1n**	**OpenMV Cam H7**
Alimentación	5V	3.3V	5V
Frecuencia de reloj	1GHz	400MHz/600MHz	480MHz
Número de cores	1	2	1
RAM	512MB	6MB	1MB
Dimensiones	65mm x 30mm	30mm x 22mm	35.5mm x 44.4mm
Precio (COP)	$66.000	$56.000	$234.000

Tabla 3.1: Comparativa: placas de desarrollo
Fuente: Autor.

3.1.2 Sipeed M1n

Figura 3.1: Sipeed M1n & camera
Fuente: Seeed.

Debido a su bajo costo, su alto poder de procesamiento para tareas de visión artificial y sus dimensiones, se escogió la Sipeed M1n. Es un kit de desarrollo diseñado por Sipeed, el kit incluye el módulo M1n, un adaptador USB tipo C para la programación del mismo y una cámara de baja resolución.

El módulo cuenta con todo lo necesario para el desarrollo de proyectos de tamaño reducido y que requieran de tareas de inteligencia artificial, ofreciendo un gran desempeño por un bajo precio y bajo consumo de energía. Cuenta con pines GPIO configurables, de los cuales se puede usar cualquiera para comunicación I2C, I2S, UART, etc.

El módulo M1n incluye el chip para inteligencia artificial K210 que tiene en su interior una CPU (Central Processing Unit), una NPU (Neural Processing Unit) concebido con la estructura de una red neuronal, optimizado para utilizar menos batería y una APU (Audio Processing Unit) especial para el pre-procesamiento de audio.

Dicho módulo puede ser programado usando el IDE de Arduino o el IDE de Maixpy usando micropython. [32]

Características principales

- Múltiples funcionalidades: Detección facial, Reconocimiento de objetos, Espectograma FFT, etc.
- CPU: procesador RISC-V 64 bit dual-core con una frecuencia estándar de 400MHz (overclockeable).
- Reconocimiento de imágenes: QVGA@60FPS / VGA@30FPS.
- Reconocimiento de voz: Soporta un arreglo con hasta 8 micrófonos.

- Soporte para frameworks como Keras, Tensorflow y Darknet.
- Periféricos: FPIOA, UART, GPIO, SPI, I2C, I2S, WDT

Procesador de IA - K210

Figura 3.2: K210 SoC
Fuente: Sipeed.

Es un SoC (Sistema en Chip) que integra todos los componentes necesarios para desarrollar tareas de visión y escucha computacional con un bajo consumo de potencia (0.3W). Todos los componentes que lo conforman se pueden observar en la figura 3.3.

El K210 es capaz de realizar detección y reconocimiento de rostros y de imágenes basándose en redes neuronales convolucionales usando su KPU (acelerador de CNN), además de obtener el tamaño y coordenadas del objetivo detectado en tiempo real.

El chip también tiene la capacidad de reconocimiento de audio. Incluye un procesador de audio de alto rendimiento, para una matriz de micrófonos, siendo capaz de estimar la localización de la fuente y del reconocimiento de voz [33].

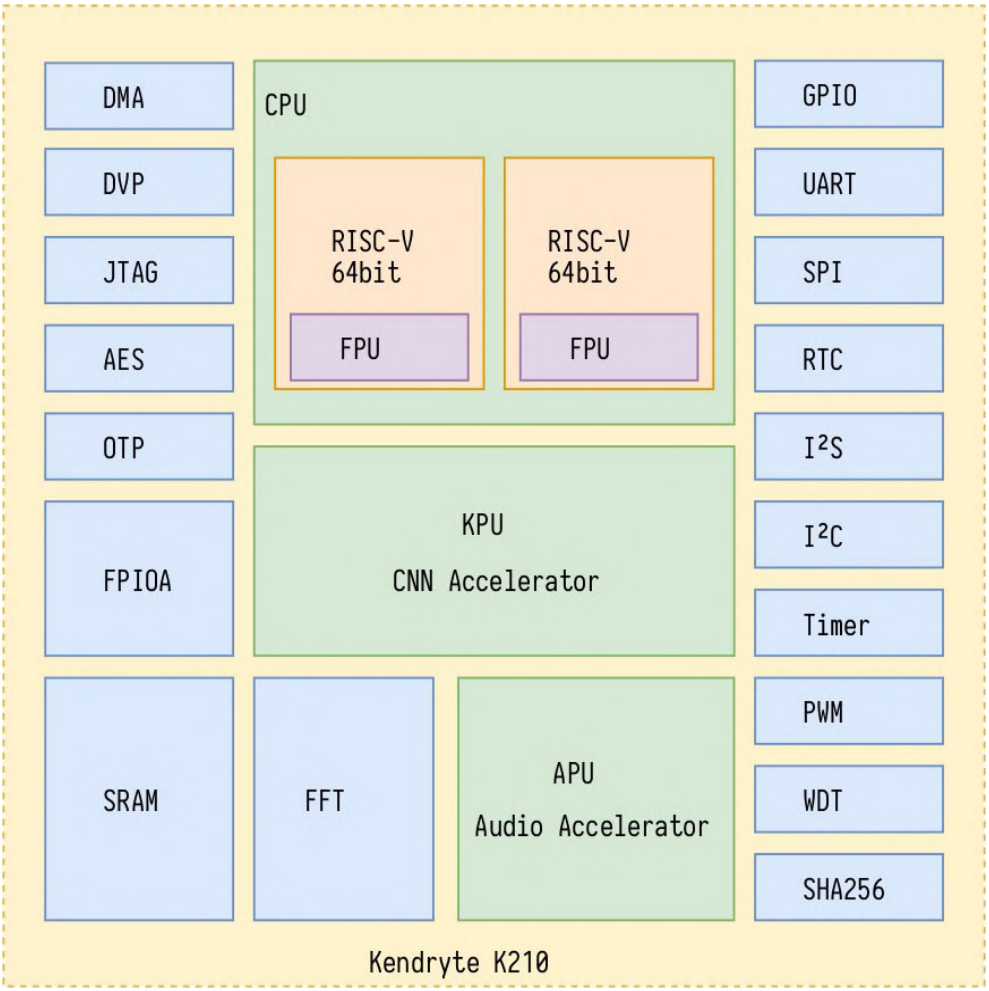

Figura 3.3: Arquitectura del K210
Fuente: Wu.

3.1.3 DFPlayer Mini

El DFPlayer Mini es un reproductor MP3 de bajo costo con una salida simplificada que puede ser conectada directamente a un parlante de baja potencia (ver figura 3.4). Este puede ser usado a través de botones para subir/bajar el volumen, cambiar de canción, etc. Pero en conjunto con una tarjeta de control y el uso de comunicación UART se puede usar de manera automática para proyectos que requieran de la reproducción de audio [35].

Figura 3.4: Reproductor de MP3 DFPlayer Mini
Fuente: DFRobot.

Especificaciones

- Tasas de muestreo soportadas (kHz): 8/11.025/12/16/22.05/24/32/44.1/48.
- Salida DAC de 24 bits.
- Soporta SD/USB de hasta hasta 32GB de almacenamiento con sistemas de archivos FAT16 o FAT32.
- Variedad de modos de control: modo de control I/O, modo comunicación serial, modo de control con botones AD.
- Los datos de audio son ordenados por carpeta, soporta hasta 100 y cada carpeta puede contener hasta 255 canciones.
- Nivel de volumen ajustable (30 niveles).
- Cuenta con 6 pre-configuraciones de ecualizador.

3.1.4 Conversor DC/DC Pololu

Este regulador DC-DC, puede entregar un voltaje de salida ajustable desde 2.2V hasta 5.25V tomando un voltaje de entrada entre 0.5V Y 5.5v, con una alta eficiencia y apagado automático por calentamiento (ver figura 3.5). Compatible con tarjetas perforadas y protoboards por el espaciamiento de sus pines de 2.54mm [36].

Figura 3.5: Convertidor DC/DC Pololu
Fuente: DidacticasElectrónicas.

Especificaciones

- Voltaje de entrada: 0.5V 5.25V.
- Salida de voltaje ajustable: 2V hasta 5.25V.
- Corriente de entrada máximo de 1.2A.
- Dimensiones: 11.5mm * 15.3mm * 2.5mm
- Apagado automático por calentamiento.

3.1.5 Módulo de carga TP4056

Módulo que integra el chip TP4056, diseñado para cargar baterias tipo LiPo (ver figura 3.6), de carga lineal ajustable para celdas de hasta 1A, con un sistema de protección de carga. Su voltaje de entrada va desde 4.5V hasta 5.5V en DC y carga la celda conectada hasta los 4.2VDC ± 1.5%. Incluye dos LEDs indicadores, uno para carga y otro para carga completa de la batería.

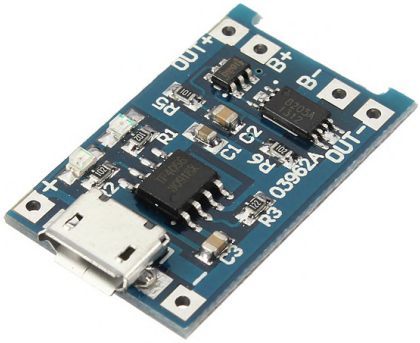

Figura 3.6: Módulo de carga TP4056
Fuente: electrónca.

Especificaciones

- Método de carga: Lineal.
- Corriente de carga: 1A ajustable por cambio de resistencia.
- Precisión de carga: 1.5%.
- Protección contra sobre-corrientes: 3A.
- Interface de entrada: conector micro-USB.
- Temperatura de operación: -10°C hasta 85°C.
- Dimensiones: 26mm * 17mm * 10mm [37]

3.2 Diseño y construcción de las PCB

Se utilizó el sofware Eagle de Autodesk bajo la licencia de educación para el desarrollo del esquemático y del circuito impreso, en la cual se montaron el DFPlayer y el convertidor DC/DC Pololu, con los respectivos espacios para los pines de alimentación

y comunicación del módulo Sipeed y la toma de alimentación proveniente del módulo de carga TP4056.

3.2.1 Esquema electrónico

En el esquema electrónico se presentan de una manera sencilla, las interconexiones entre los distintos componentes que conforman el dispositivo. En el esquemático se incluye el DFPlayer para la reproducción de los audios de retroalimentación acústica y el convertidor DC/DC Pololu. Además se usaron regletas tipo macho para las conexiones externas con el fin de garantizar un espaciado estándar entre los componentes.

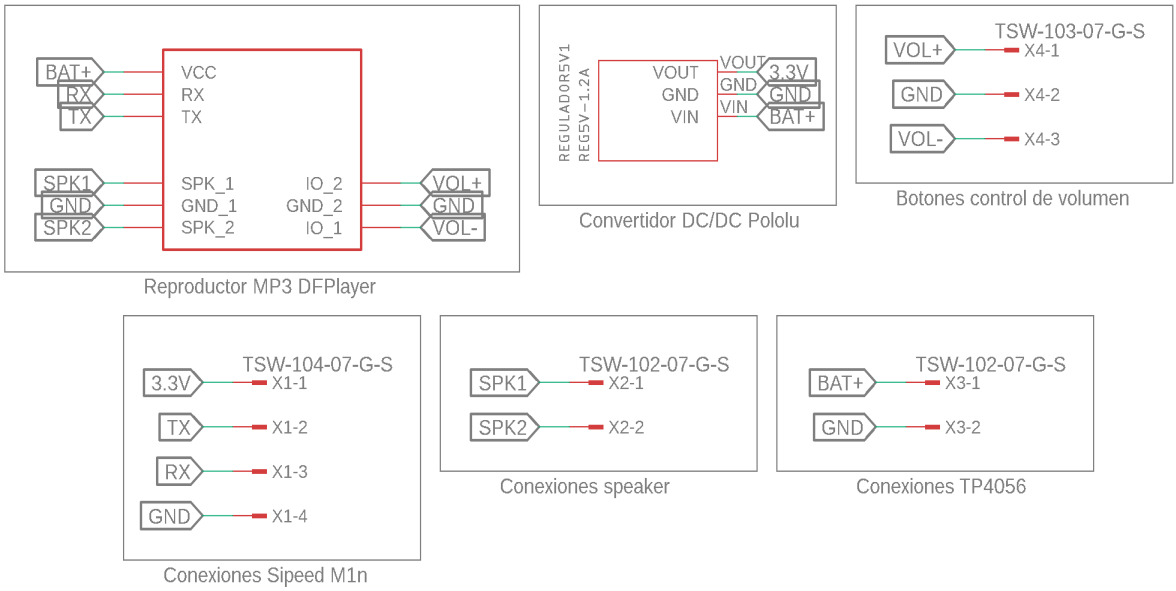

Figura 3.7: Esquemático de la placa
Fuente: Autor

3.2.2 Diseño de la PCB

Basándose en el esquemático mostrado en la figura 3.7, se diseñó el circuito impreso (PCB) con el fin de tener acceso a los botones y espacio para la micro SD del DFPlayer y que ocupara el menor espacio posible, al final se obtuvieron unas dimensiones de 25.4mm * 44.45mm como se evidencia en la figura 3.8. Además se exportó un modelo 3D de la placa (ver figura 3.9) usando Fusion 360 con el fin de diseñal la carcasa con las medidas exactas.

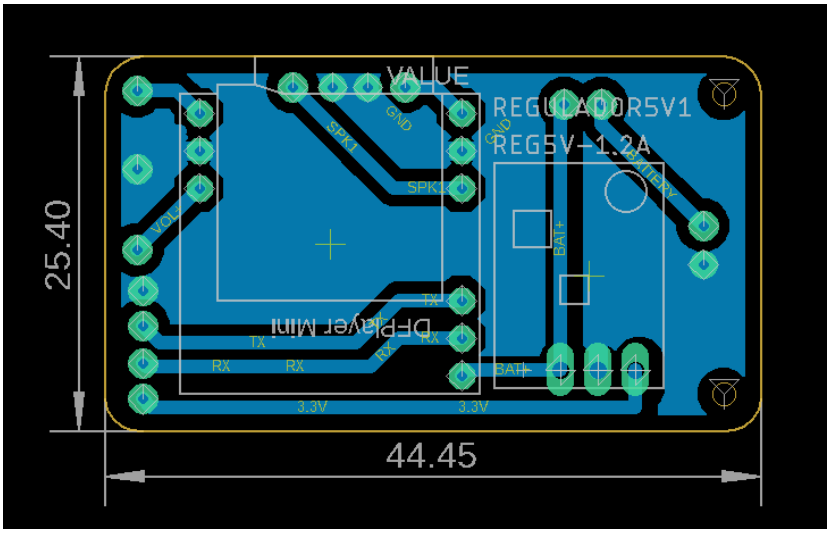

Figura 3.8: PCB de la placa
Fuente: Autor

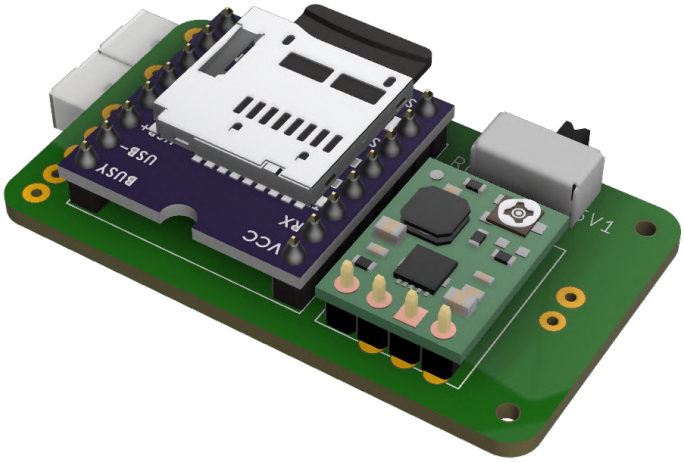

Figura 3.9: Representación 3D de la PCB
Fuente: Autor

Terminado el diseño de la PCB en el software Eagle, se inicia el proceso de fabricación de la PCB en una baquela virgen usando el método de planchado, en el cual se imprime el diseño previamente aplicando el efecto espejo, sobre papel termo-transferible (como por ejemplo el usado para fotografías o acetato) usando una impresora láser. Luego de esto se plancha hasta que la capa con la impresión quede adherida sobre la baquela,

para posteriormente sumergirla sobre ácido férrico, con el fin de eliminar todo el cobre excepto en las partes donde está la tinta de tóner adherida. Después de esto se perforan los orificios para los componentes y cableado usando una broca de diámetro apropiado. Por último se insertan cada uno de los componentes en la baquela y se fijan usando crema, estaño y cautín. En la figura 3.10 se observa la PCB terminada.

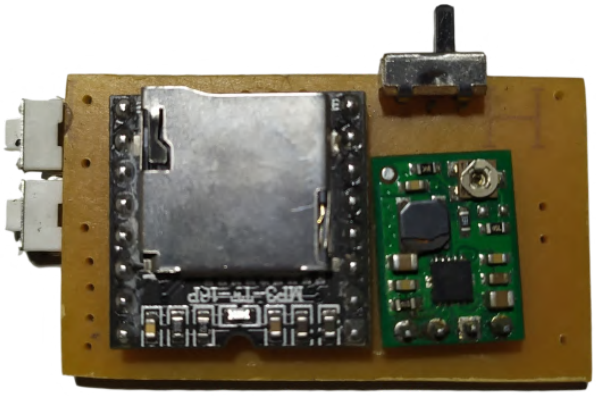

Figura 3.10: PCB tras el proceso de planchado y soldado de componentes
Fuente: Autor

3.3 Construcción del prototipo

El prototipo fue construido usando unas gafas de sol comerciales como base, para el diseño de las piezas 3D que hicieron de soporte mecánico para los componentes mencionados en la sección anterior. Además se tuvieron en cuenta modelos 3D de las placas a usar para obtener diseños con medidas exactas y que todo encajara perfectamente.

3.3.1 Diseño de piezas 3D

Las piezas 3D fueron diseñadas en el software Fusion 360 de la empresa Autodesk bajo licencia de educación. Es un software CAD, CAM y CAE alojado en la nube, que sirve para diseñar diversos tipos de productos. Combina el diseño industrial y mecánico, simulación, colaboración y maquinado, todo en uno [38].

- **Soporte para la Sipeed M1n** Se diseñó una carcasa aparte para la tarjeta de desarrollo, ya que esta solo es compatible con la cámara incluida en el kit, y esta cuenta con una flex de muy corta longitud. Fue diseñada usando encajes para un uso mínimo de tornillos y fácil ensamble. Además se usaron dos imanes de neodimio con el fin de que las gafas pudieran dividirse, para que el usuario tuviera facilidad al colocárselas o quitárselas (Ver figura 3.11).

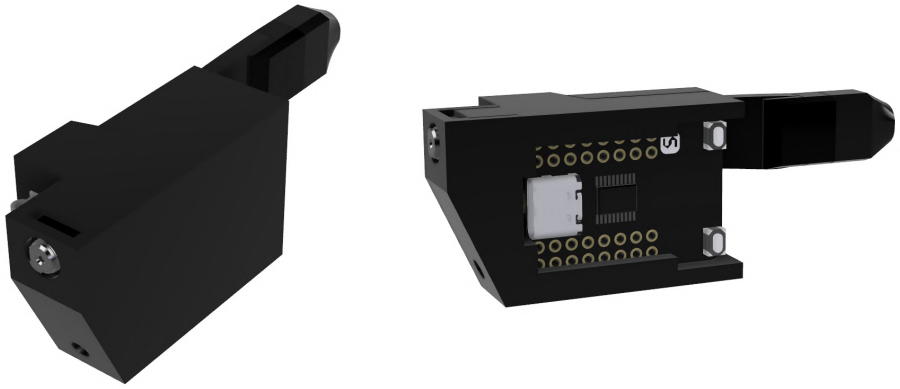

Figura 3.11: Ensamble carcasa Sipeed M1n
Fuente: Autor

- **Soporte para DFPlayer, convertidor DC/DC, batería y cargador TP4056**
  Se diseñó una carcasa con todos los cortes necesarios para tener acceso a la tarjeta micro SD contenedora de los audios, botones y al puerto micro USB tipo B para la carga del dispositivo. Cuenta con orificios en los que se insertarán mangueras de diálisis, que harán las veces de conductos para el cableado de alimentación y comunicación con el módulo Sipeed (Ver figura 3.12).

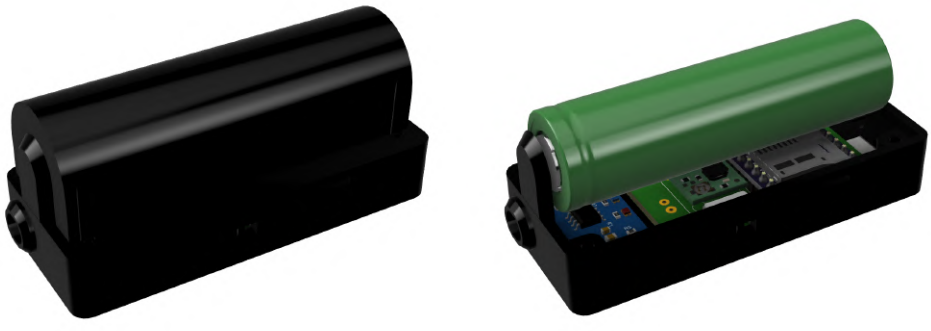

Figura 3.12: Ensamble carcasa PCB's y batería
Fuente: Autor

- **Soporte para el auricular** Se diseñó una pieza que hará las veces de carcasa para el auricular que realizará la retroalimentación acústica o sonora (ver figura 3.13), este irá sobre la manguera de diálisis mencionada anteriormente, de tal manera que quede ubicado detrás del oído del usuario.

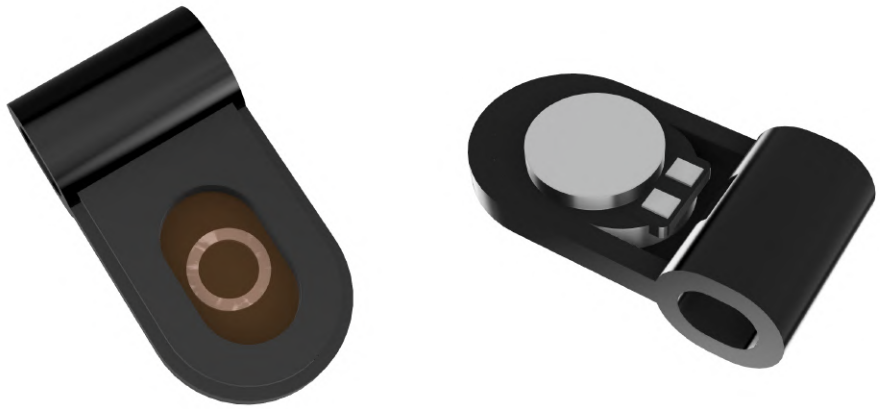

Figura 3.13: Carcasa auricular
Fuente: Autor

3.3.2 Ensamble

El ensamble del prototipo fue realizado con el software Fusion 360. Este ayudó a comprobar que todos los elementos diseñados tuvieran las medidas exactas para acoplarse con los demás elementos impresos, las placas desarrolladas y las gafas originales. También se obtuvo una renderización del modelo para visualizar el estado final del prototipo (ver figura 3.14). En la figura 3.15 se puede ver el resultado final y en la figura 3.16 se puede ver el dispositivo en uso.

Características del dispositivo	
Dimensiones	16cm * 24cm * 4.5cm
Peso	110g
Autonomía	5 horas
Cargador requerido	5V/1A (min)

Tabla 3.2: Características del dispositivo.
Fuente: Autor.

Figura 3.14: Ensamble 3D del prototipo
Fuente: Autor

Figura 3.15: Ensamble del prototipo
Fuente: Autor

Figura 3.16: Dispositivo en uso
Fuente: Autor

3.4 Sipeed M1n

En esta sección se describe el entrenamiento de la red neuronal convolucional a cargar en la tarjeta Sipeed M1n y el proceso de conversión de dicha red al formato requerido por la tarjeta. Además se describen los algoritmos para la implementación de esta red, la detección de rostros que se realiza con un modelo ya entrenado ofrecido por la empresa Sipeed y la comunicación UART con el módulo MP3 DFPlayer.

Para el entrenamiento de la red neuronal se usó la librería Keras, por la simplicidad que brinda a la hora de crear y entrenar modelos.

Para acelerar el proceso de entrenamiento de las redes neuronales, se siguió el procedimiento descrito en la guía provista en la página oficial de tensorflow para el uso de la tarjeta gráfica o GPU [39].

3.4.1 Dataset para el entrenamiento de la CNN

El conjunto de datos usado para el entrenamiento de la red neuronal convolucional fue el provisto por la plataforma Kaggle para el reto de reconocimiento de expresiones faciales realizado en 2013. En la figura 3.17 se pueden observar algunas imágenes del dataset.

Este conjunto de datos consta de imágenes en escala de grises con un tamaño de 48 x 48 píxeles.donde se registran caras mas o menos centradas y que ocupan el mismo espacio en cada imagen.

El conjunto de datos para el entrenamiento consta de 28.709 imágenes, mientras que el conjunto de datos para la validación consta de 3.589 imágenes en las cuales se presentan 7 emociones (Enojado, Disgustado, Asustado, Feliz, Triste, Sorprendido, Neutral) [40].

Figura 3.17: Imágenes del dataset FER-2013
Fuente: Kaggle

En el desarrollo del proyecto se incluyeron 5 de las emociones (Enojado, Feliz, Triste, Sorprendido, Neutral), ya que son las emociones mas frecuentes en una conversación. Además el uso de más clases aumenta el número de parámetros de la red neuronal, provocando que el espacio que esta ocupa incremente, el cual es limitado en la tarjeta.

Para aumentar el número de imágenes se usó la clase ImageDataGenerator de la librería Keras, con el cual se ingresa cada imagen y se obtienen imágenes con efecto espejo en ambos ejes, rotación sobre el eje z determinados grados, etc.

3.4.2 Entrenamiento de la CNN

La arquitectura elegida para el desarrollo del proyecto fue la MobileNet debido al poco espacio que usa y su alta precisión. Se usó un modelo pre-entrenado y se ajustaron algunos parámetros usando el método de transfer-learning, con el cual se conservan los parámetros de las primeras capas, encargadas de clasificar características simples como lineas y figuras simples, y se entrenan solo las capas superiores, encargadas de clasificar características mas específicas como partes del cuerpo, texturas, partes de animales o plantas, etc [41].

Parámetros de entrenamiento de la MobileNet

Se usó un modelo pre-entrenado de MobileNet con un tamaño de entrada de 128 x 128, por lo cual era necesario escalar las imágenes del dataset a dicho tamaño. Además se usó solo el modelo base, configurando la red para que no se tuvieran en cuenta las últimas capas, teniendo la posibilidad de agregar otras para la clasificación final entre las 5 nuevas clases.

Se establecieron las primeras 70 capas como no entrenables, lo cual indica que sus pesos no cambiarán durante el proceso de entrenamiento y desde la capa número 70 en adelante, se establecieron como entrenables.

Después de las respectivas modificaciones, la arquitectura de la red resultante es la que se describe en la tabla 3.3.

Tipo de capa	Forma de salida	Parámetros
input_1 (InputLayer)	[(None, 128, 128, 3)]	0
conv_pw_13_relu (MobileNet)	(None, 4, 4, 768)	0
global_average_pooling2d	(None, 768)	0
dense (Dense)	(None, 256)	196864
dropout (Dropout)	(None, 256)	0
dense_1 (Dense)	(None, 128)	32896
dense_2 (Dense/Softmax)	(None, 5)	645

Tabla 3.3: Sumario de la arquitectura MobileNet
Fuente: Autor.

la arquitectura de la red terminó con un total de 2.063.381 parámetros, de los cuales 1.279.109 son entrenables y 784.272 no son entrenables.

Entrenamiento de la red

Para el entrenamiento se usó el optimizador Adam con un factor de aprendizaje de 0.01 y 40 épocas. Además se implementaron funciones para guardar el modelo (en formato '.h5') con mejor desempeño durante el entrenamiento, de manera que si la precisión del modelo no mejoraba transcurrida cierta cantidad de iteraciones, el proceso se detenía o se reducía el factor de aprendizaje. En el código 3.1 se explica a detalle el funcionamiento por medio de comentarios.

Código 3.1: Algoritmo para el entrenamiento de la MobileNet

```
1
2 import keras
3 from keras import backend as K
4 from keras.layers.core import Dense, Activation
5 from keras.optimizers import Adam
6 from keras.metrics import categorical_crossentropy
7 from keras.preprocessing.image import ImageDataGenerator
8 from keras.preprocessing import image
9 from keras.models import Model
10 from keras.applications import imagenet_utils
11 from keras.layers import Dense, Flatten, Dropout, GlobalAveragePooling2D
12 from keras.applications import MobileNet
13 from keras.applications.mobilenet import preprocess_input
14 import numpy as np
```

```
15
16
17 train_dataset_directory = 'train'
18 validation_dataset_directory = 'validation'
19 batch_size = 32
20 name_exported_model = 'facial_expression_classification_model'
21 num_classes = 5 # 5 classes
22
23
24 # We have to pre process the image to pass to the model
25 def prepare_image(file):
26     img = file
27     img_array = image.img_to_array(img)
28     img_array_expanded_dims = np.expand_dims(img_array, axis = 0)
29     return keras.applications.mobilenet.preprocess_input(img_array_expanded_dims)
30
31 # Base model for our DNN, it will download the pretrained 'weights' file without including the top layers
32 base_model = keras.applications.mobilenet.MobileNet(
33     input_shape = (128, 128, 3),
34     alpha = 0.75,
35     depth_multiplier = 1,
36     dropout = 0.001,
37     pooling = 'avg',
38     include_top = False, # This argument must be True if we want to predict using the entire network
39     weights = "imagenet",
40     classes = 1000
41 )
42
43
44 # Modiying the output of the model, adding custom layers
45 x = base_model.output
46 x = Dense(256, activation='relu')(x)
47 x = Dropout(0.001)(x)
48 x = Dense(128, activation='relu')(x)
49
50 preds = Dense(num_classes, activation = 'softmax')(x)
51
52 # Redine the entire model
53 model = Model(inputs = base_model.input, outputs = preds)
54
55 # Print the name of the output tensor, neccesary to know for the conversion to .kmodel format
56 print(model.layers[-1].output)
57
58 # Set no trainable layers
59 for layer in model.layers[:70]:
60     layer.trainable = False
61
62 # Set trainable layers
63 for layer in model.layers[70:]:
64     layer.trainable = True
65
66
67 # Generating training data
68 train_data_generator = ImageDataGenerator(
69     preprocessing_function = prepare_image,
70     horizontal_flip = True
71 )
72
73 # Obtaining training data from the specified directory
74 train_generator = train_data_generator.flow_from_directory(
75     train_dataset_directory,
76     target_size = (128, 128),
77     color_mode = 'rgb',
78     batch_size = batch_size,
```

```
    class_mode = 'categorical',
    shuffle = True
)

# Generating validation data
validation_data_generator = ImageDataGenerator(preprocessing_function = prepare_image)

# Obtaining validation data from the specified directory
validation_generator = validation_data_generator.flow_from_directory(
    validation_dataset_directory,
    target_size = (128, 128),
    color_mode = 'rgb',
    batch_size = batch_size,
    class_mode = 'categorical',
    shuffle = True
)

# Print the model summary
model.summary()

# Specify the optimizer and metrics for the training
model.compile(
    optimizer = Adam(lr = 0.01),
    loss = 'categorical_crossentropy',
    metrics = ['accuracy']
)

from keras.optimizers import RMSprop, SGD, Adam
from keras.callbacks import ModelCheckpoint, EarlyStopping, ReduceLROnPlateau

# Creating the callbacks for early stopping and reducing learning rate

# Checkpoint: save the model in each epoch
checkpoint = ModelCheckpoint(name_exported_model + '.h5',
                             monitor = 'val_loss',
                             mode = 'min',
                             save_best_only = True,
                             verbose = 1)

# Model accuracy not improving for 6 rounds, then stop
early_stop = EarlyStopping(monitor = 'val_loss',
                           min_delta = 0,
                           patience = 6,
                           verbose = 1,
                           restore_best_weights = True)

# Model accuracy not improving, then reduce the learning rate
reduce_lr = ReduceLROnPlateau(monitor = 'val_loss',
                              factor = 0.2,
                              patience = 3,
                              verbose = 1,
                              min_delta = 0.0001)

callbacks = [checkpoint, early_stop, reduce_lr]

nb_train_samples = 24123
nb_validation_samples = 5919
epochs = 40

# Train the model
history = model.fit_generator(
    train_generator,
    steps_per_epoch = nb_train_samples // batch_size,
    epochs = epochs,
```

```
    callbacks = callbacks,
    validation_data = validation_generator,
    validation_steps = nb_validation_samples // batch_size
)
```

Verificación del funcionamiento de la red

Después del entrenamiento, se cargó el modelo usando el código 3.2 para verificar su correcto funcionamiento usando imágenes tomadas de la cámara web. Para el reconocimiento de rostros y obtención de la posición de los mismos se usó el método del clasificador en cascada, disponible mediante la librería de open-cv.

Código 3.2: Algoritmo para la verificación del funcionamiento del modelo.

```
import keras
from keras.models import load_model
from time import sleep
from keras.preprocessing.image import img_to_array
from keras.preprocessing import image
import cv2
import numpy as np

face_classifier = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_default.xml')
classifier = load_model('facial_expression_classification_model.h5')

# Labels shown in the screen
class_labels = ['Angry', 'Happy', 'Neutral', 'Sad', 'Surprise']

# Setting WebCam
cap = cv2.VideoCapture(0)
preds_arg_max_before = -1
preds_arg_max = -1
label = 'None'

while True:
    # Take image from the WebCam
    ret, frame = cap.read()

    # Convert image from BGR to grayscale
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    # Convert image from BGR to RGB
    color = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)

    # Find faces in the grayscale image
    faces = face_classifier.detectMultiScale(gray, 1.3, 5)

    for (x, y, w, h) in faces:
        # Draw a rectangle in the image
        cv2.rectangle(frame, (x, y), (x + w, y + h), (255, 0, 0), 2)

        # Cut the face in the image
        roi_color = color[y : y + h, x : x + w, :]

        # Resize the image to use it in the MobileNet
        roi_color = cv2.resize(roi_color, (128, 128), interpolation = cv2.INTER_AREA)

        if np.sum([roi_color]) != 0:
            # Preprocess the resized image to use the MobileNet
```

```
            roi = img_to_array(roi_color)
            roi = np.expand_dims(roi, axis = 0)
            roi = keras.applications.mobilenet.preprocess_input(roi)

            # Predict the output with the MobileNet
            preds = classifier.predict(roi)
            preds_arg_max_before = preds_arg_max
            preds_arg_max = preds.argmax()

            if preds_arg_max == preds_arg_max_before:
                label = class_labels[preds_arg_max]

            print(preds)

            # Put label in the image
            label_position = (x, y)
            cv2.putText(frame, label, label_position, cv2.FONT_HERSHEY_SIMPLEX, 2, (0, 255, 0), 3)

        else:
            cv2.putText(frame, 'No face found', (20, 60), cv2.FONT_HERSHEY_SIMPLEX, 2, (0, 255, 0), 3)

    # Show image
    cv2.imshow('Emotion detector', frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

cap.release()
cv2.destroyAllWindows()
```

3.4.3 Preparación de la Sipeed M1n

La Sipeed M1n requiere de la conversión de los modelos obtenidos del entrenamiento al formato '.kmodel' para poder ser cargados a la KPU. En este caso, el modelo no debe sobrepasar los 2.5MB, ya que este comparte el espacio de almacenamiento junto con el firmware de la tarjeta y el archivo '.kmodel' encargado de la detección de rostros.

Conversión del modelo obtenido al requerido por la tarjeta

Por medio del entrenamiento de la red neuronal realizado con el código 3.1, se obtiene el modelo en formato '.h5', el cual se transforma a formato '.tflite' mediante el código 3.3.

Código 3.3: Algoritmo para la conversión de formato '.h5' a '.tflite'

```
from keras.models import load_model
import tensorflow as tf

# Model's name in the current directory
model_name = 'facial_expression_classification_model'

# Load the model
model = load_model(model_name + '.h5')

# Print the summary of the model
model.summary()

# Print the name of the output tensor
```

```
output_tensor = model.layers[-1].output
print(output_tensor)

# Convert model.h5 to model.tflite
tf.compat.v1.disable_eager_execution()
converter = tf.compat.v1.lite.TFLiteConverter.from_keras_model_file(model_name + '.h5', output_arrays = ['dense_2/Softmax'])

tfmodel = converter.convert()
file = open(model_name + '.tflite', 'wb')
file.write(tfmodel)
file.close()
print('tflite file created...')
```

Una vez convertido el modelo al formato '.tflite', es necesario convertirlo al formato '.kmodel' haciendo uso de el compilador de redes neuronales Nncase, software desarrollado por Sipeed para usar modelos creados con keras, tensorflow o pytorch, en los chips K210. En este caso se usó la versión 0.2.0 Beta 4, con soporte para kmodel versión 4.

Nncase es un programa de consola, no tiene interfaz gráfica. Para convertir el modelo, se organizaron los archivos necesarios como se muestra en el árbol de directorios de la figura 3.18. El modelo en formato '.tflite' dentro de la carpeta model, ruta en la cual se exportará el archivo '.kmodel' y las imágenes de inferencia dentro de la carpeta imgs, usadas por nncase durante el proceso de conversión. Las imágenes fueron escaladas usando una herramienta gratuita online para que coincidieran con el tamaño de entrada de la red (128 x 128).

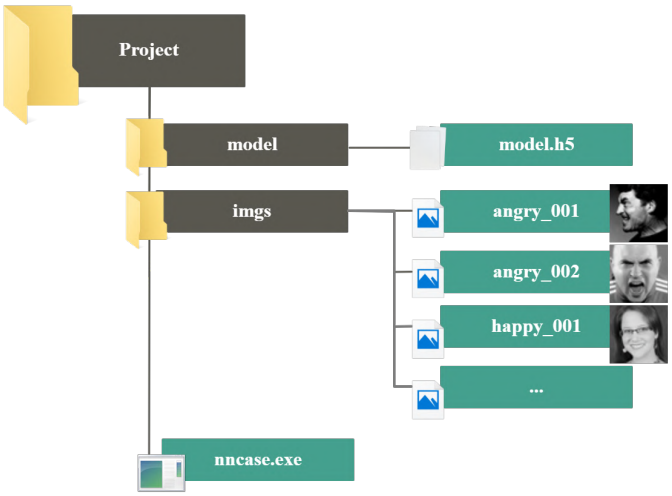

Figura 3.18: Árbol de directorios conversión tflite a kmodel usando nncase
Fuente: Autor.

Después de tener organizados los archivos correctamente, se procede a la conversión del modelo usando el código 3.4 dentro de una ventana de comandos de Windows,

donde se especifica la ruta del archivo '.tflite' de entrada, la ruta del archivo '.kmodel' de salida, el formato de entrada, el formato de salida y el directorio donde se encuentran las imágenes de inferencia. Tras el proceso de conversión se obtiene el modelo con un peso aproximado de 2.1MB.

Código 3.4: Comando para la conversión de '.tflite' a '.kmodel'

```
ncc compile model\model.tflite model\model.kmodel -i tflite -o kmodel --dataset imgs
```

Carga de los archivos a la tarjeta

Tras la obtención del archivo '.kmodel', se procede a cargar los archivos a la Sipeed M1n. Para esto es necesario el software 'k-flash', desarrollado también por la empresa Sipeed.

Primero que todo es necesario comprimir los archivos en el formato '.kgpkg', archivo en el cual están el firmware de la tarjeta en formato '.bin', el archivo '.kmodel' encargado de la detección de rostros, el archivo '.kmodel' encargado de la clasificación de expresiones faciales y el archivo de configuración 'flash-list.json', en el cual se especifica en qué dirección de memoria se cargará cada uno de los archivos mencionados anteriormente (ver código 3.5).

Código 3.5: Archivo de configuración 'flash-list.json'

```
{
  "version": "0.1.0",
  "files": [
    {
      "address": 0,
      "bin": "maixpy_firmware.bin",
      "sha256Prefix": true
    },
    {
      "address": 0x300000,
      "bin": "face_detection.kmodel",
      "sha256Prefix": false
    },
    {
      "address": 0x500000,
      "bin": "facial_expression_recognition.kmodel",
      "sha256Prefix": false
    }
  ]
}
```

Después de editar el archivo 'flash-list.json', se comprimen los 4 archivos usando

cualquier software de compresión en el formato '.zip', formato el cual debe cambiarse a '.kfpkg' haciendo click derecho sobre el archivo y luego en la opción 'cambiar nombre'.

Por último se carga el archivo a la tarjeta con k-flash, en su interfaz gráfica se selecciona el archivo '.kfpkg', se selecciona la tarjeta 'Sipeed Maixduino', el puerto COM de la tarjeta y se da click en 'Download', el proceso se describe también en la figura 3.19.

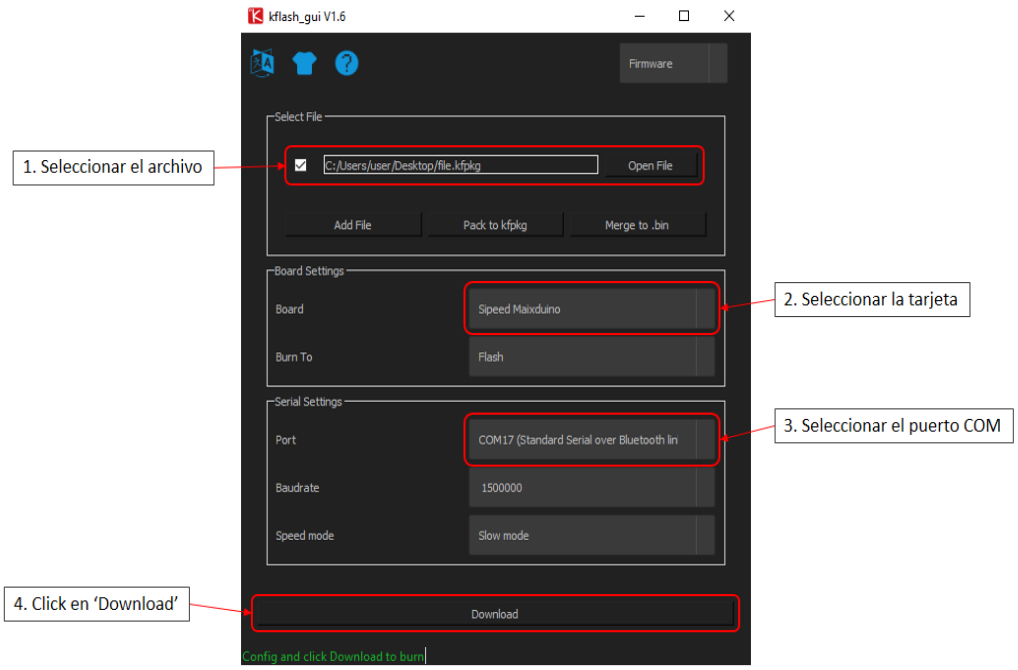

Figura 3.19: Carga del archivo '.kfpkg' a la tarjeta Sipeed M1n
Fuente: Autor.

3.4.4 Algoritmos

Comunicación de la tarjeta Sipeed M1n con el módulo DFPlayer

Para realizar la comunicación entre la Sipeed M1n y el módulo MP3 DFPlayer se utilizó el protocolo UART, el cual es configurable en cualquier conjunto de pines de la Sipeed (en este proyecto se usó el pin 20 para la transmisión (TX) y el pin 21 para la recepción (RX)). La tabla 3.4 muestra el formato del comando que debe enviarse al módulo MP3 para realizar la acción deseada (pausar el audio, cambiar de canción, bajar el volumen, etc) y la tabla 3.5 muestra los distintos comandos que se pueden enviar. En este proyecto solo se usaron los comandos "start" para inicializar la comunicación

con el módulo, ”set volume” para establecer un nivel de volumen apropiado y ”Specify tracking” para especificar la pista a reproducir.

Formato: $S VER Len CMD Feedback para1 para2 checksum $O		
$S	Bit inicio 0x7E	Cada comando empieza con $
VER	Version	Información de la versión
Len	El número de bits tras ’Len’	Checksum es no contado
CMD	Comandos	Indicar la operación
Feedback	Comando de retroalimentación	Si es requerido
para1	Parámetro 1	MSB de la cola
para2	Parámetro 2	LSB de la cola
checksum	checksum	Acumulación y verificación
$O	Bit final	Bit final 0xEF

Tabla 3.4: Formato para el control serial - DFPlayer
Fuente: DFRobot.

CMD	**Descripción de la función**	**Parámetros (16 bit)**
0x01	Siguiente pista	
0x02	Pista anterior	
0x03	Especificar pista	0-2999
0x04	Incrementar volumen	
0x05	Decrementar volumen	
0x06	Especificar volumen	0-30
0x07	Especificar EQ	0/1/2/3/4/5
0x08	Especificar modo reproducción	repetir/random

Tabla 3.5: Comandos de control serial - DFPlayer
Fuente: DFRobot.

Código 3.6: Comandos DFPlayer usando comunicación UART

```
# Init communication with DFPlayer
cmd_init = bytearray([0x7E, 0xFF, 0x06, 0x0C, 0x01, 0x00, 0x00, 0xFE, 0xEE, 0xEF])

# Set volume
cmd_vol = bytearray([0x7E, 0xFF, 0x06, 0x06, 0x01, 0x00, 0x14, 0xFE, 0xE0, 0xEF])

# Play a specified track
cmd_1st_song = bytearray([0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x01, 0xFE, 0xF6, 0xEF])
cmd_2nd_song = bytearray([0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x02, 0xFE, 0xF5, 0xEF])
cmd_3rd_song = bytearray([0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x03, 0xFE, 0xF4, 0xEF])
cmd_4th_song = bytearray([0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x04, 0xFE, 0xF3, 0xEF])
cmd_5th_song = bytearray([0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x05, 0xFE, 0xF2, 0xEF])
```

Algoritmo de la Sipeed M1n

El algoritmo cargado a la tarjeta funciona como de describe a continuación:

1. Se cargan las librerías necesarias y se declaran los pines a usar para la comunicación, que en este caso son el pin 20 y el pin 21.

2. Se declaran los comandos descritos en la seccíón anterior para el manejo del módulo MP3 DFPlayer y se inicializa la comunicación.

3. Configurar la cámara mediante la librería 'sensor'.

4. Cargar y configurar los modelos para la detección de rostros y la clasificación de expresiones faciales.

5. Dentro del bucle while, se captura la imagen de la cámara y se evalua con el modelo de detección de rostros.

6. Si un rostro es detectado, se recorta la imagen, se escala al tamaño requerido por el modelo (128 x 128) y se evalúa con el modelo de clasificación de expresiones faciales.

7. Enviar el comando al DFPlayer dependiendo de la salida obtenida en el paso anterior.

A continuación se muestra el algoritmo completo en el código 3.7 y la dinámica del funcionamiento en la figura 3.20.

Código 3.7: Algoritmo cargado a la Sipeed M1n

```
import sensor, image, lcd
import KPU as kpu
from Maix import GPIO
from fpioa_manager import *
from machine import UART
import time

# Configuration of the UART protocol
```

```
fm.register(20, fm.fpioa.UART2_TX) # Declare the pin for TX uart
fm.register(21, fm.fpioa.UART2_RX) # Declare the pin for RX uart

uart = UART(UART.UART2, 9600, 8, None, 1, timeout = 1000, read_buf_len = 4096)

# Commands for DFPlayer
cmd_init = bytearray([0x7E, 0xFF, 0x06, 0x0C, 0x01, 0x00, 0x00, 0xFE, 0xEE, 0xEF])
cmd_vol = bytearray([0x7E, 0xFF, 0x06, 0x06, 0x01, 0x00, 0x14, 0xFE, 0xE0, 0xEF]) # 14 in HEX is 20 in DEC
cmd_1st_song = bytearray([0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x01, 0xFE, 0xF6, 0xEF])
cmd_2nd_song = bytearray([0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x02, 0xFE, 0xF5, 0xEF])
cmd_3rd_song = bytearray([0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x03, 0xFE, 0xF4, 0xEF])
cmd_4th_song = bytearray([0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x04, 0xFE, 0xF3, 0xEF])
cmd_5th_song = bytearray([0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x05, 0xFE, 0xF2, 0xEF])

cmd_dict = {'Angry' : cmd_1st_song, 'Happy' : cmd_2nd_song, 'Sad' : cmd_3rd_song, 'Surprise' : cmd_4th_song}

# Init communication with DFPlayer
uart.write(cmd_init)
uart.read()
uart.write(cmd_vol)
uart.read()

lcd.init(freq=15000000)
# Reset and initialize the sensor
sensor.reset()
# Set pixel format to RGB565 (or GRAYSCALE)
sensor.set_pixformat(sensor.RGB565)
# Set frame size to QVGA (320x240)
sensor.set_framesize(sensor.QVGA)
# Wait for settings take effect.
sensor.skip_frames(time = 2000)
# Flip the image vertically
sensor.set_vflip(1)
# Flip the image horizontally
sensor.set_hmirror(1)
# Create a clock object to track the FPS.
clock = time.clock()

# Loading face detect model
task_detect_face = kpu.load(0x300000)
# Loading face expresion classify model
task_classify_face = kpu.load(0x500000)

a = kpu.set_outputs(task_classify_face, 0, 1, 1, 5)

anchor = (1.889, 2.5245, 2.9465, 3.94056, 3.99987, 5.3658, 5.155437, 6.92275, 6.718375, 9.01025)
a = kpu.init_yolo2(task_detect_face, 0.5, 0.3, 5, anchor)

# Facial expression labels
labels = ['Angry', 'Happy', 'Neutral', 'Sad', 'Surprise']

while(True):
    # Update the FPS clock.
    clock.tick()
    # Take a picture and return the image.
    img = sensor.snapshot()
    # Perform the face detection task
    detected_face = kpu.run_yolo2(task_detect_face, img)

    if detected_face:
        for i in detected_face:
            # Cut the detected face
```

```
71              face = img.cut(i.x(), i.y(), i.w(), i.h())
72              # Resize the image
73              face_128 = face.resize(128, 128)
74              a = face_128.pix_to_ai()
75              # Perform the facial expression classification task
76              fmap = kpu.forward(task_classify_face, face_128)
77              plist = fmap[:]
78              # Obtain the label with max probability
79              pmax = max(plist)
80              print("%s: %s" % (labels[plist.index(pmax)], pmax))
81              label = labels[plist.index(pmax)]
82
83              # Send command to DFPlayer
84              if label != 'Neutral':
85                  uart.write(cmd_dict[label])
86                  uart.read()
87                  time.sleep(2)
```

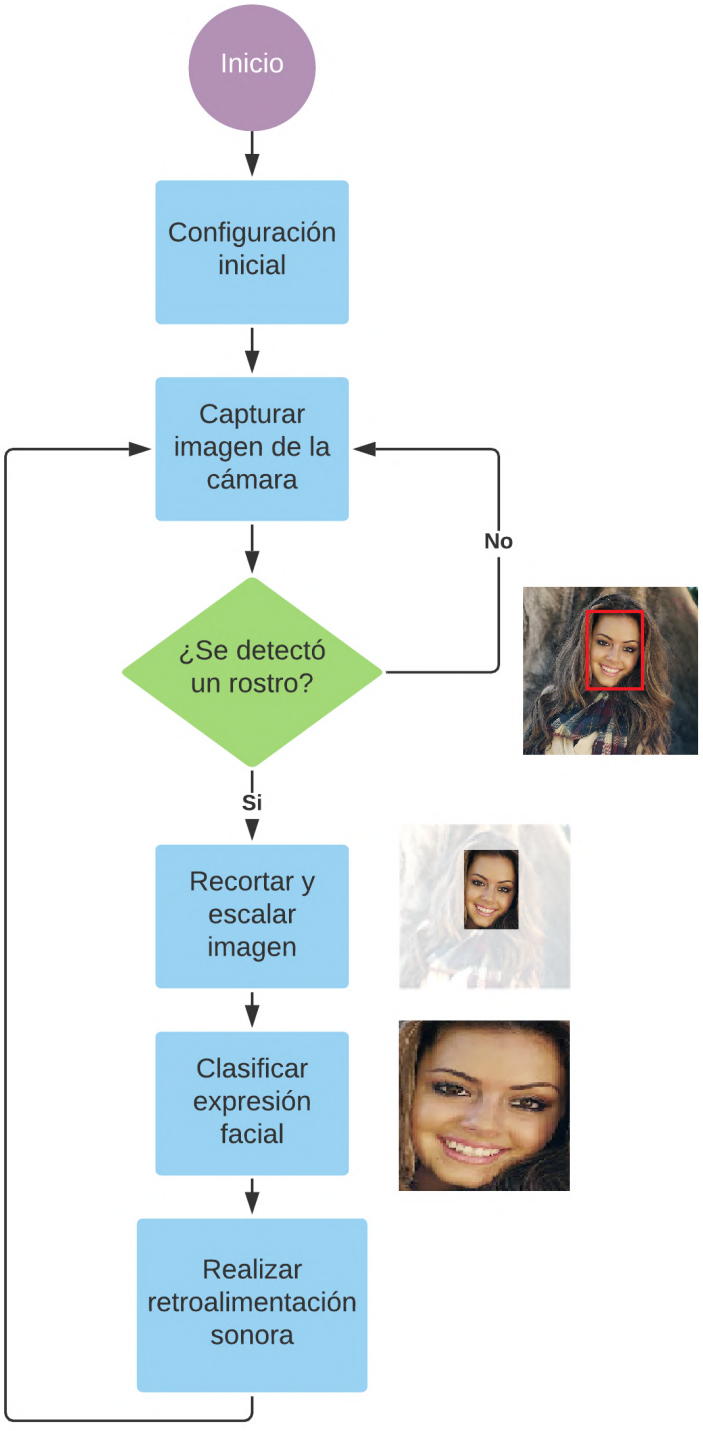

Figura 3.20: Diagrama de flujo: algoritmo del dispositivo
Fuente: Autor.

4 Resultados

4.1 La arquitectura MobileNet

En esta sección se exponen los resultados obtenidos del entrenamiento de la red neuronal usando la librería Keras/Tensorflow además de las pruebas de funcionamiento de la misma usando OpenCV para la verificación en el PC para posteriormente usar la tarjeta Sipeed y realizar la verificación final usando el prototipo junto con su retroalimentación sonora.

4.1.1 Entrenamiento

Tras numerosos intentos con distintas configuraciones de las capas agregadas a la arquitectura MobileNet se obtuvieron las gráficas que representan la precisión de la red (Figura 4.1) y costos de la red (Figura 4.2) a medida que el modelo iba siendo entrenado.

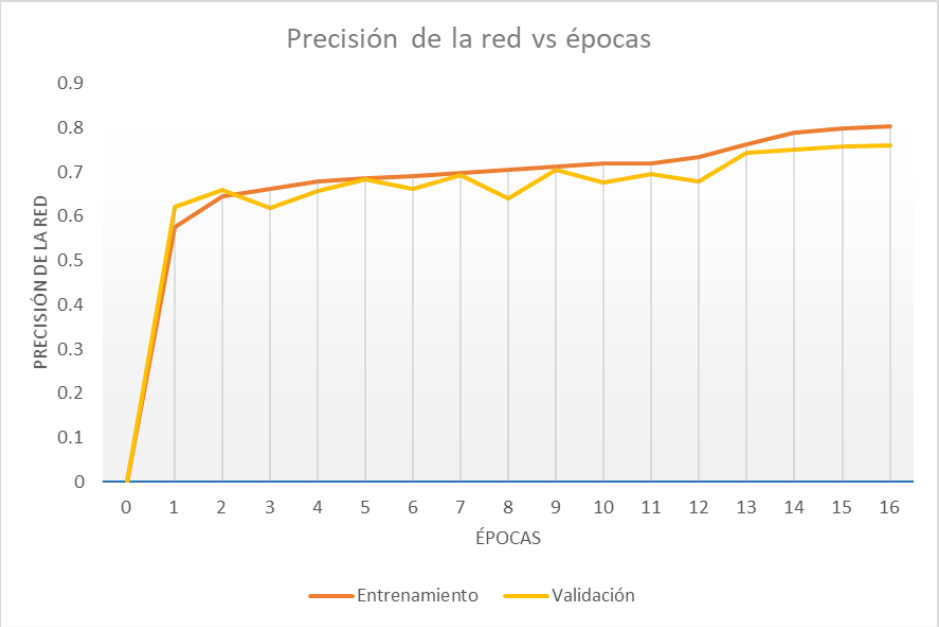

Figura 4.1: Gráfico precisión de MobileNet vs épocas
Fuente: Autor.

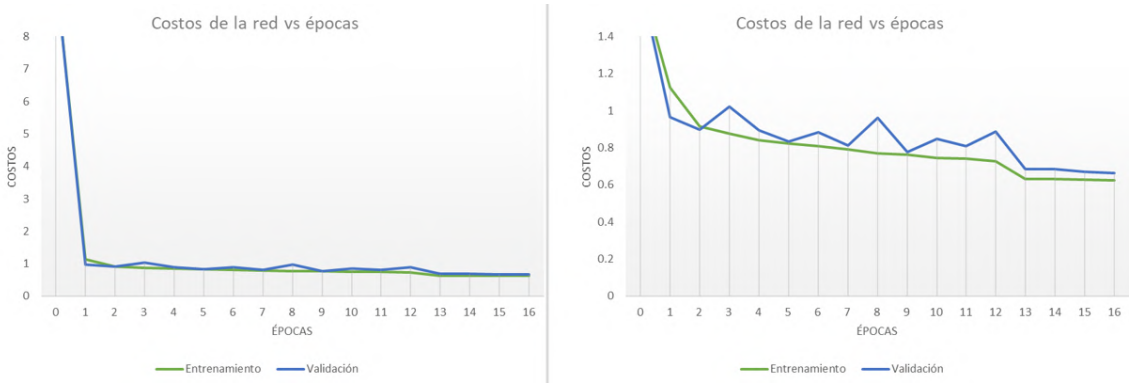

Figura 4.2: Gráfico costos de MobileNet vs épocas
Fuente: Autor.

El entrenamiento del modelo finalizó de manera anticipada debido a que la precisión del mismo no estaba aumentando tras la época número 16. La precisión de la red es al rededor de un 80% lo que es un valor aceptable para reconocimiento de imágenes. Sin embargo, dicho valor puede incrementarse usando otras configuraciones y parámetros de entrenamiento para la red neuronal, pero esto implica un aumento del espacio de almacenamiento que el modelo ocupa, el cual es limitado en el sistema embebido implementado en el prototipo.

4.1.2 Verificación del funcionamiento usando OpenCV

Una vez obtenido el modelo entrenado, se procedió a verificar su correcto funcionamiento con ayuda de la librería de visión computacional OpenCV para desarrollar la detección de rostros usando la cámara web e imágenes con las distintas expresiones faciales. En la figura 4.3 se puede observar cada una de las expresiones faciales correctamente clasificadas por el modelo y en la figura 4.4 se muestra el proceso para la identificación de la expresión facial junto con mas ejemplos para la verificación.

Figura 4.3: Verificación del funcionamiento de la red con OpenCV
Fuente: Autor.

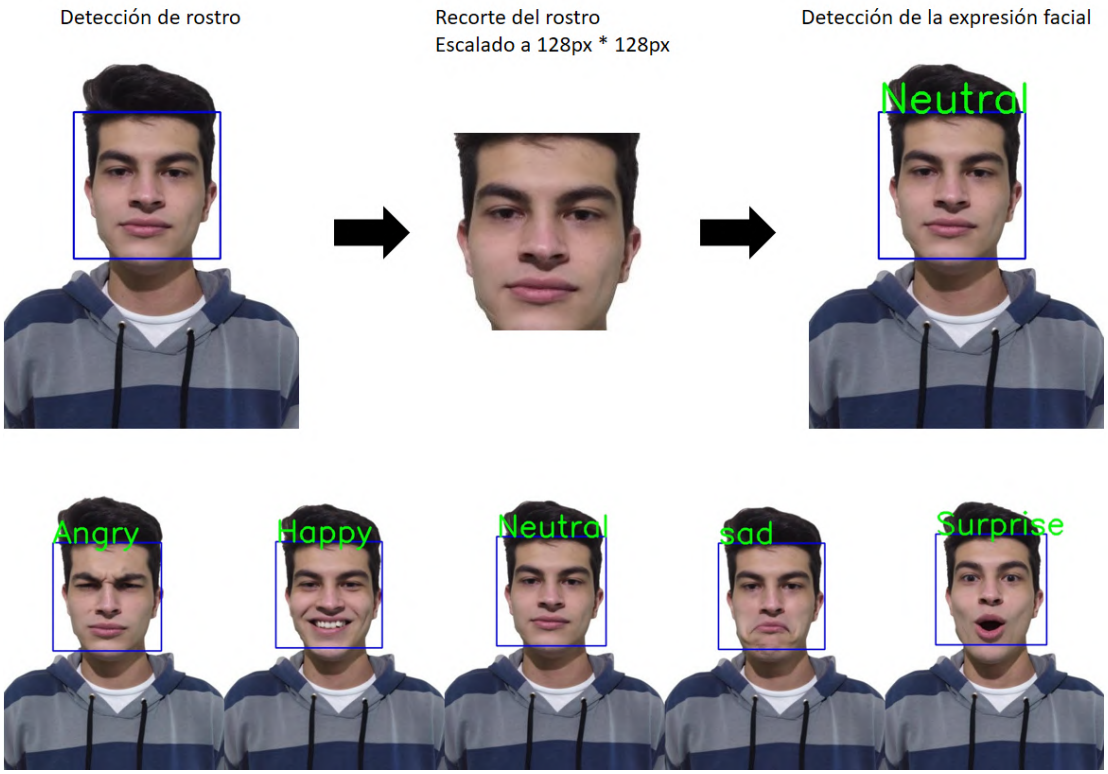

Figura 4.4: Funcionamiento de la red con OpenCV
Fuente: Autor.

4.1.3 Verificación del funcionamiento del modelo en el dispositivo

Tras entrenar el modelo con el mejor desempeño, se convirtió usando la herramienta nncase con el procedimiento descrito en la sección 3.4.3 y se cargó a la Sipeed M1n para verificar el funcionamiento del dispositivo en un entorno real. Tras varias pruebas se evidenció una fuerte influencia de la iluminación del entorno, el dispositivo funciona de manera ineficiente cuando no se cuenta con un buen nivel de luz, por lo que es recomendable usarlo solo cuando se tiene un nivel de luminosidad apropiado.

Siguiendo la recomendación mencionada en el párrafo anterior, se realizaron pruebas con el dispositivo en 10 individuos (5 de sexo femenino y 5 de sexo masculino) de edades comprendidas entre los 18 y los 55 años para verificar la robustez del modelo cargado a la tarjeta. Debido a que el firmware cargado a la tarjeta no permite la comunicación entre esta y el entorno de desarrollo para la visualización de las imágenes y resultados de la red neuronal, se optó por usar la retroalimentación acústica para conocer la salida de la red neuronal.

Las tablas 4.1 y 4.2 muestran la información recopilada del los usuarios de sexo femenino y de sexo masculino respectivamente. Los resultados fueron clasificados en aciertos y fallos para cada una de las 5 emociones, realizando 5 pruebas por cada una de ellas. Dicha clasificación se realizó con el fin de determinar el porcentaje de acierto para cada emoción y validar el correcto funcionamiento del dispositivo.

Se puede observar que el modelo se desempeña en una emoción de manera eficaz para cierto usuario mientras que para otro se desempeña de manera pobre, por ejemplo para el usuario 5 la emoción ”enojado” presenta 0% de acierto mientras que para el usuario 4 muestra un 80% de acierto.

Usuario	Expresión facial	Aciertos	Fallos
1 Edad: 17	Feliz	5	0
	Enojado	2	3
	Triste	1	4
	Sorprendido	4	1
	Neutral	5	0
2 Edad: 22	Feliz	3	2
	Enojado	3	2
	Triste	2	3
	Sorprendido	5	0
	Neutral	5	0
3 Edad: 23	Feliz	3	2
	Enojado	5	0
	Triste	3	2
	Sorprendido	2	3
	Neutral	5	0
4 Edad: 27	Feliz	4	1
	Enojado	4	1
	Triste	3	2
	Sorprendido	2	3
	Neutral	5	0
5 Edad: 55	Feliz	4	1
	Enojado	0	5
	Triste	1	4
	Sorprendido	2	3
	Neutral	5	0

Tabla 4.1: Resultados de las pruebas realizadas a cada usuario del sexo femenino
Fuente: Auto

Usuario	Expresión facial	Aciertos	Fallos
6 Edad: 20	Feliz	5	0
	Enojado	4	2
	Triste	5	0
	Sorprendido	3	2
	Neutral	5	0
7 Edad: 22	Feliz	5	0
	Enojado	3	2
	Triste	1	4
	Sorprendido	3	2
	Neutral	0	5
8 Edad: 24	Feliz	4	1
	Enojado	0	5
	Triste	2	3
	Sorprendido	4	1
	Neutral	4	1
9 Edad: 30	Feliz	5	0
	Enojado	5	0
	Triste	0	5
	Sorprendido	5	0
	Neutral	5	0
10 Edad: 30	Feliz	5	0
	Enojado	2	3
	Triste	5	0
	Sorprendido	0	5
	Neutral	0	5

Tabla 4.2: Resultados de las pruebas realizadas a cada usuario del sexo masculino.
Fuente: Autor.

En la figura 4.5 se puede observar que la emoción que presenta mayor tasa de éxito entre los usuarios de sexo femenino es la de ”neutral” seguida de la de ”feliz”, mientras que la que obtuvo el menor valor fue la de ”enojado”. En el sexo opuesto (figura 4.6) ocurre que la expresión facial con mas tasa de éxito es la de ”feliz” y la que cuenta con el menor éxito es la de ”enojado” al igual que en el caso de las mujeres. Esto se debe a que la expresión facial ”feliz” es muy distintiva de las demás por el color blanco de los dientes cuando el usuario sonríe, mientras que la expresión facial ”enojado” es la

que mas se confunde con la de ”neutral” por lo que dicha expresión debe realizarse de forma exagerada para que haya mas nivel de distinción.

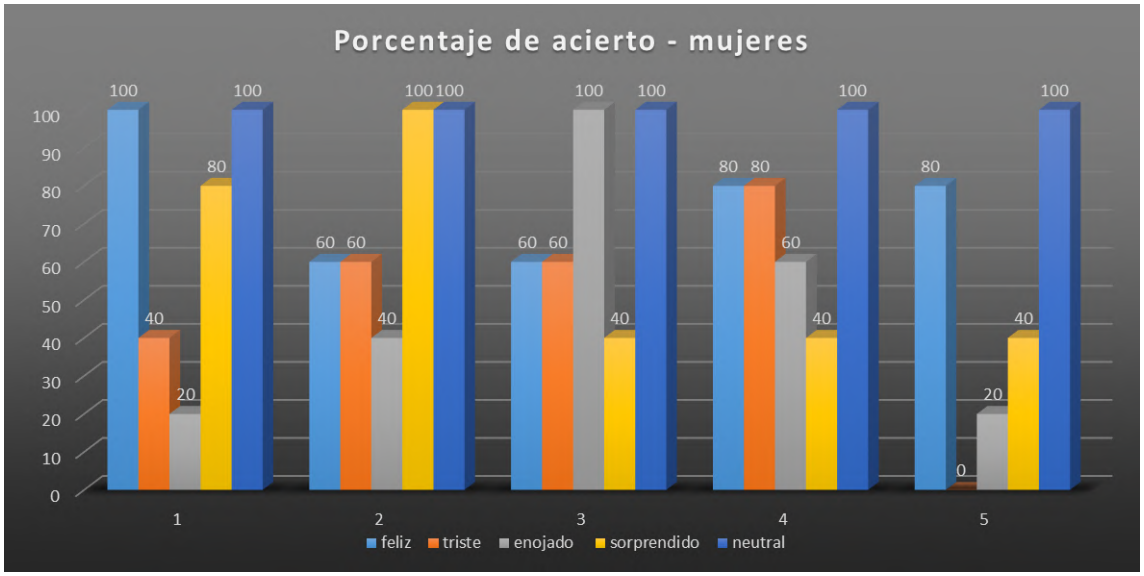

Figura 4.5: Diagrama de barras porcentaje de acierto para cada emoción (mujeres)
Fuente: Autor.

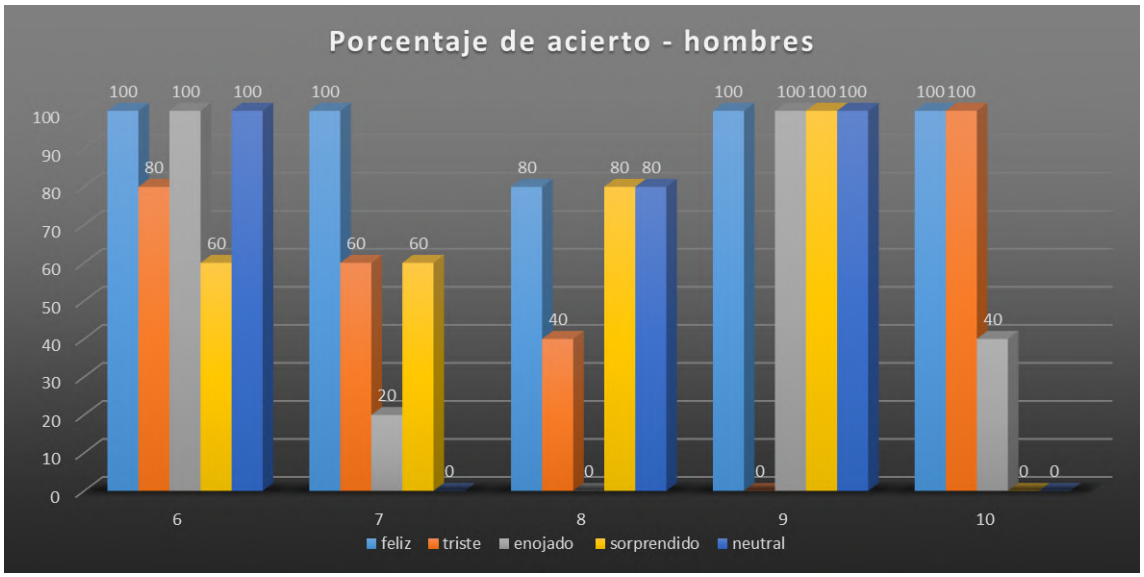

Figura 4.6: Diagrama de barras porcentaje de acierto para cada emoción (hombres)
Fuente: Autor.

Indicadores estadísticos del experimento

En la tabla 4.3 se presentan los indicadores estadísticos de media, mediana y moda para cada una de las 5 emociones de toda la población. En esta se puede apreciar que todas las expresiones faciales superan un promedio de porcentaje de éxito del 50% llegando a un 86% por parte de la expresión facial "feliz", seguida por un 78% para la expresión facial "neutral". En la figura 4.7 se puede observar el diagrama de barras de los indicadores estadísticos.

Índice estadístico	Feliz	Triste	Enojado	Sorprendido	Neutral
Moda	100%	60%	20%	40%	100%
Media	86%	52%	50%	60%	78%
Mediana	90%	60%	40%	60%	100%

Tabla 4.3: Índices estadísticos del experimento (hombres y mujeres)
Fuente: Autor.

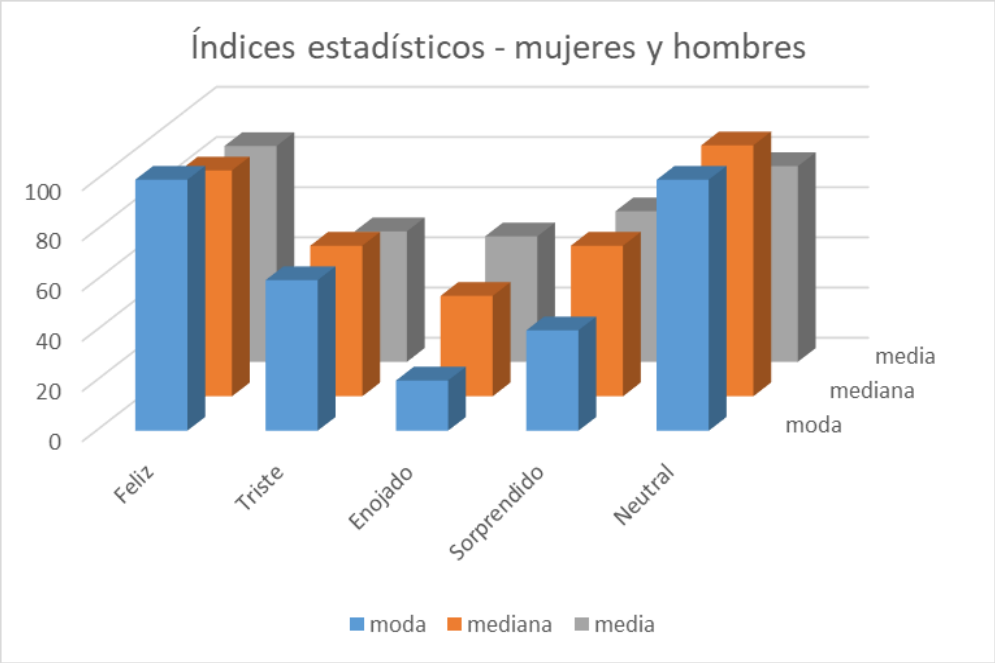

Figura 4.7: Diagrama de barras índices estadísticos (hombres y mujeres)
Fuente: Autor.

Así mismo se tienen las tablas 4.4 y 4.5 junto con los diagramas de barras de la figura 4.8 y 4.9 que representan los índices estadísticos para los usuarios de sexo femenino y masculino respectivamente.

Índice estadístico	Feliz	Triste	Enojado	Sorprendido	Neutral
Moda	60%	60%	20%	40%	100%
Media	76%	48%	48%	60%	100%
Mediana	80%	60%	40%	40%	100%

Tabla 4.4: Índices estadísticos del experimento (mujeres)
Fuente: Autor.

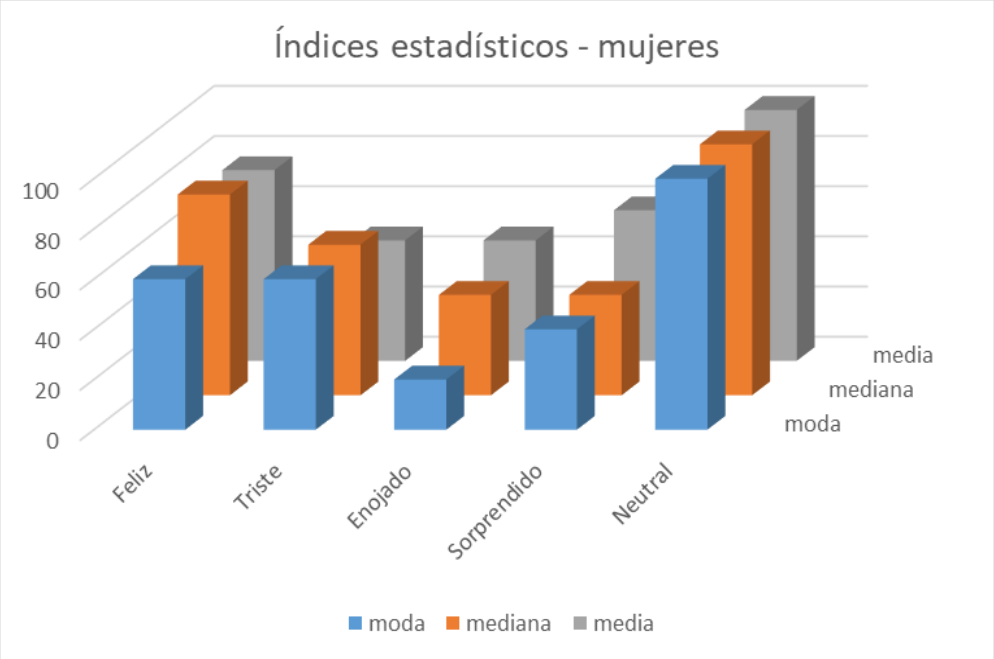

Figura 4.8: Diagrama de barras índices estadísticos (mujeres)
Fuente: Autor.

Índice estadístico	Feliz	Triste	Enojado	Sorprendido	Neutral
Moda	100%	60%	40%	60%	80%
Media	96%	56%	52%	60%	56%
Mediana	100%	60%	40%	60%	80%

Tabla 4.5: Índices estadísticos del experimento (hombres)
Fuente: Autor.

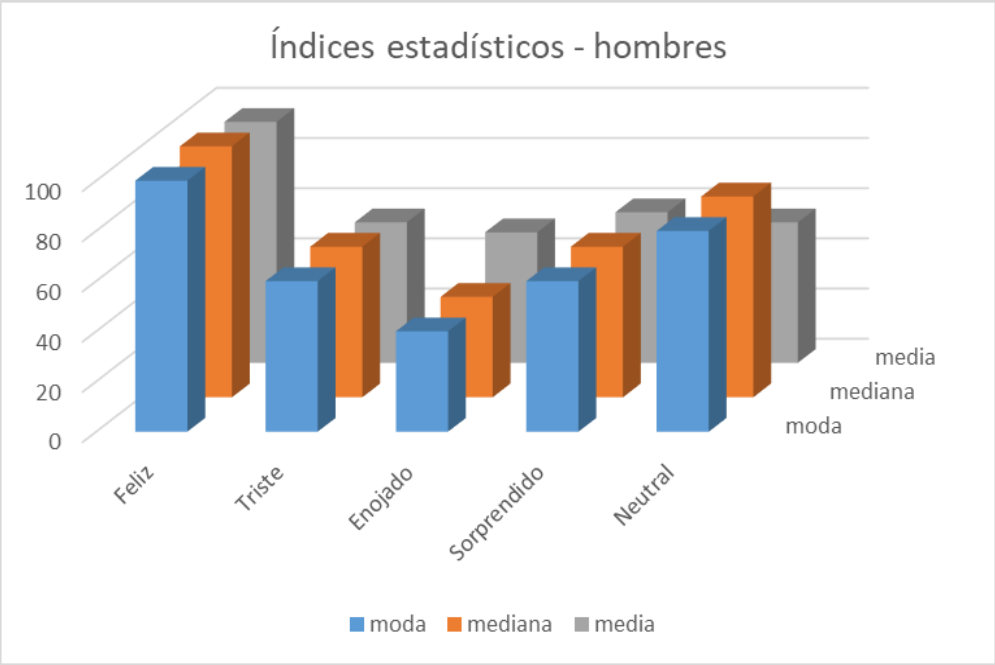

Figura 4.9: Diagrama de barras índices estadísticos (hombres)
Fuente: Autor.

Con los resultados obtenidos se evidencia que a pesar de que el modelo cargado a la tarjeta cuenta con un porcentaje de precisión alto, hay momentos en los que el dispositivo no se desempeña de la mejor manera, esto es debido a que la cámara cuenta con muy baja resolución, usando una mejor cámara se obtendrían resultados con tasas de acierto mayores.

4.2 Prueba del dispositivo en un individuo con discapacidad visual

Finalmente se realizaron pruebas del dispositivo en un individuo con discapacidad visual principalmente para conocer sus opiniones en cuanto a las características del mismo, tales como la comodidad, que es uno de los aspectos mas importantes si se quiere que el usuario use el dispositivo por un largo periodo de tiempo. En la figura 4.10 se puede observar al usuario usando el dispositivo.

Figura 4.10: Usuario con discapacidad visual usando el dispositivo.
Fuente: Autor.

La opinión dada por el usuario fue la siguiente:

... Me gustó como funciona el dispositivo, se escucha bien el audio, no es pesado, se siente cómodo al usarlo y no le cambiaría nada.

5 Conclusiones

Teniendo en cuenta los resultados obtenidos, se determina que el dispositivo se desempeña de manera efectiva pese a las limitaciones de hardware si se cumple el principal requerimiento de contar con un buen nivel de iluminación en el entorno en el que se use, recalcando que el funcionamiento se dará de la mejor manera cuando se use con luz ambiente.

La principal limitante es el uso de una cámara con baja resolución sin posibilidad de reemplazo, esto debido que el sistema embebido implementado solo es compatible con esta referencia. Teniendo esto en cuenta se puede decir que la red neuronal convolucional MobileNet entrenada con el método de transfer-learning y cargada a la tarjeta de desarrollo Sipeed M1n se desempeñó de manera eficaz en el dispositivo.

Dicho modelo presentó la mejor tasa de acierto promedio para la expresión facial ”feliz” con un 86% y la menor tasa promedio para la expresión facial ”enojado” con un 50% por la similitud que presenta con la expresión ”neutral” y ”triste”. Además la tasa de éxito promedio para ambos sexos es semejante, con valores de 66.4% y 64% para individuos de sexo femenino y de sexo masculino respectivamente.

También se evidenció que el dispositivo cumple con los requerimientos para la creación de un dispositivo de sustitución sensorial al recibir críticas positivas por el usuario con discapacidad visual con quien se realizó el experimento, presentando un buen nivel de comodidad. El método de retroalimentación de audio implementado para dar a conocer las expresiones faciales detectadas se desempeñó de manera correcta sin obstruir el normal funcionamiento del sentido del oído.

Finalmente, el desempeño general del dispositivo puede mejorarse simplemente cambiando los componentes del hardware por unos de mayor calidad, manteniendo la lógica de funcionamiento junto con la retroalimentación de audio. Dicho cambio contribuiría a la reducción del peso del sistema y a un aumento en la comodidad que brinda al usuario al usarlo.

Bibliografía

[1] María Arias. Autor: María Elisa Arias Roura. pages 1–70, 2010. URL http://dspace.ucuenca.edu.ec/bitstream/123456789/2835/1/te4148.pdf.

[2] Suárez Escudero and Juan Camilo. Discapacidad visual y ceguera en el adulto: revisión de tema. *Medicina U.P.B.*, 30(2):170–180, 2011. ISSN 0120-4874.

[3] Javier Checa, Benito Pura, Díaz Veiga, Rafael Pallero González, Almudena Cacho González, Carmen Calvo, Novell Javier, Checa Benito, Miguel Díaz, Salabert Pura, Jorge Luis González Fernández, Luis González, García José, Luis González Sánchez, Rafael Pallero González María, Victoria Puig, Samaniego María, Victoria Quílez, and García Coordinación. *Psicología y ceguera: Manual para la intervención psicológica en el ajuste a la discapacidad visual*. 2003. ISBN 8448401239.

[4] Bruno Liesen. El braille: origen, aceptación y difusión. pages 5–35, 01 2002.

[5] Yuhang Zhao, Elizabeth Kupferstein, Hathaitorn Rojnirun, Leah Findlater, and Shiri Azenkot. The Effectiveness of Visual and Audio Wayfinding Guidance on Smartglasses for People with Low Vision. *Conference on Human Factors in Computing Systems - Proceedings*, (February), 2020. doi: 10.1145/3313831.3376516.

[6] Michele A. Williams, Amy Hurst, and Shaun K. Kane. "Pray before you step out": Describing personal and situational blind navigation behaviors. *Proceedings of the 15th International ACM SIGACCESS Conference on Computers and Accessibility, ASSETS 2013*, 2013. doi: 10.1145/2513383.2513449.

[7] Rita M. Párra. Realidad De Las Personas Con Discapacidad Visual Y Escolaridad Inconclusa En "San Pablo" De Manta Y Propuesta De Guía De Estrategias Metodológicas Para Potenciar El Aprendizaje De Lectoescritura. page 144, 2015. URL https://dspace.ups.edu.ec/bitstream/123456789/10032/1/UPS-GT000857.pdf.

[8] Norshidah Mohamad Salleh and Khalim Zainal. How and why the visually impaired students socially behave the way they do. *Procedia - Social and Behavioral Sciences*, 9:859–863, 2010. ISSN 18770428. doi: 10.1016/j.sbspro.2010.12.249. URL http://dx.doi.org/10.1016/j.sbspro.2010.12.249.

[9] C. Ferrari, T. Vecchi, L. B. Merabet, and Z. Cattaneo. Blindness and social trust: The effect of early visual deprivation on judgments of trustworthiness. *Consciousness and Cognition*, 55(September):156–164, 2017. ISSN 10902376. doi: 10.1016/j.concog.2017.08.005.

[10] Sungman Park, Yeongtae Jung, and Joonbum Bae. An interactive and intuitive control interface for a tele-operated robot (AVATAR) system. *Mechatronics*, 55 (August):54–62, 2018. ISSN 09574158. doi: 10.1016/j.mechatronics.2018.08.011. URL `https://doi.org/10.1016/j.mechatronics.2018.08.011`.

[11] Árni Kristjánsson, Alin Moldoveanu, Ómar I. Jóhannesson, Oana Balan, Simone Spagnol, Vigdís Vala Valgeirsdóttir, and Rúnar Unnthorsson. Designing sensory-substitution devices: Principles, pitfalls and potential 1. *Restorative Neurology and Neuroscience*, 34(5):769–787, 2016. ISSN 18783627. doi: 10.3233/RNN-160647.

[12] T. Ifukube, T. Sasaki, and C. Peng. A blind mobility aid modeled after echolocation of bats. *IEEE Transactions on Biomedical Engineering*, 38(5):461–465, 1991. doi: 10.1109/10.81565.

[13] Roberta Klatzky, Dinesh Pai, and Eric Krotkov. Perception of material from contact sounds. *Presence: Teleoperators and Virtual Environment*, 9:399–410, 04 2000. doi: 10.1162/105474600566907.

[14] Aleksander Väljamäe and Mendel Kleiner. Spatial sound in auditory vision substitution systems. volume 1, 05 2006.

[15] Molly Y. Zhou and William F. Lawless. An Overview of Artificial Intelligence in Education. *Encyclopedia of Information Science and Technology, Third Edition*, (1):2445–2452, 2014. doi: 10.4018/978-1-4666-5888-2.ch237.

[16] Gheorghe Tecuci. Artificial intelligence. *Wiley Interdisciplinary Reviews: Computational Statistics*, 4(2):168–180, 2012. ISSN 19395108. doi: 10.1002/wics.200.

[17] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*, volume 9781107057135. 2013. ISBN 9781107298019. doi: 10.1017/CBO9781107298019.

[18] Villani C. WHAT IS ARTIFICIAL INTELLIGENCE? Villani mission on artificial intelligence. (March), 2018.

[19] Damián Jorge Matich. Redes Neuronales: Conceptos Básicos y Aplicaciones. *Historia*, page 55, 2001. URL `ftp://decsai.ugr.es/pub/usuarios/castro/Material-Redes-Neuronales/Libros/matich-redesneuronales.pdf`.

[20] Bravo Caicedo and Jesús Lopez. *Redes Neuronales Artificiales*. 2010. ISBN 970-15-0571-9. doi: 10.1016/S0210-5691(05)74198-X.

[21] Fernando Izaurieta and Carlos Saavedra. Redes Neuronales Artificiales. *Charlas de fisica*, pages 1–15, 1999. ISSN 02105691. doi: 10.1016/S0210-5691(05)74198-X.

[22] Lionel Pigou, Sander Dieleman, Pieter Jan Kindermans, and Benjamin Schrauwen. Sign language recognition using convolutional neural networks. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8925:572–578, 2015. ISSN 16113349. doi: 10.1007/978-3-319-16178-5_40.

[23] Ernesto Varela and Edwin Campbells. Redes Neuronales Artificiales: Una Revisión del Estado del Arte, Aplicaciones y Tendencias Futuras. *Investigación y Desarrollo en TIC*, 2(1):18–27, 2011. URL `http://publicaciones.unisimonbolivar.edu.co/rdigital/inovacioning/index.php/identic/article/viewFile/21/29`.

[24] Mario Campos. Inspiraci n biol gica de las redes neuronales artificiales, 2020. URL `https://medium.com/soldai/inspiraci%C3%B3n-biol%C3%B3gica-de-las-redes-neuronales-artificiales-9af7d7b906a`.

[25] Eduardo Rivera. Introduccion a redes neuronales artificiales, 2007. URL `file:///C:/Users/Juan/Downloads/4.Introduccionaredesneuronalesartificiales(1).pdf`.

[26] Saad Albawi, Tareq Abed Mohammed, and Saad Al-Zawi. Understanding of a convolutional neural network. *Proceedings of 2017 International Conference on Engineering and Technology, ICET 2017*, 2018-January(April 2018):1–6, 2018. doi: 10.1109/ICEngTechnol.2017.8308186.

[27] Rikiya Yamashita, Mizuho Nishio, Richard Kinh Gian Do, and Kaori Togashi. Convolutional neural networks: an overview and application in radiology. *Insights into Imaging*, 9(4):611–629, 2018. ISSN 18694101. doi: 10.1007/s13244-018-0639-9.

[28] Ivars Namatēvs. Deep Convolutional Neural Networks: Structure, Feature Extraction and Training. *Information Technology and Management Science*, 20(1): 40–47, 2018. ISSN 2255-9094. doi: 10.1515/itms-2017-0007.

[29] Shadman Sakib, Ahmed, Ahmed Jawad, Jawad Kabir, and Hridon Ahmed. An Overview of Convolutional Neural Network: Its Architecture and Applications. *ResearchGate*, (November), 2018. doi: 10.20944/preprints201811.0546.v1. URL `https://www.researchgate.net/publication/329220700`.

[30] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. MobileNets: Efficient convolutional neural networks for mobile vision applications. *arXiv*, 2017.

[31] E. Redmon. Pinel Et Esquirol: Quelques Commentaires Sur Les Debuts D'Une Amitie. *Annales Medico-Psychologiques*, 134 II(1):59–61, 1976. ISSN 00034487.

[32] Seeed. Sipeed m1n module ai development kit based on k210 (risc-v). url https://www.seeedstudio.com/Sipeed-M1n-Module-AI-Development-Kit-based-on-K210-p-4491.html.

[33] Sipeed. Maixpy wiki. url https://maixpy.sipeed.com/en/.

[34] Elaine Wu. Get started with k210: Hardware and programming environment. url https://www.seeedstudio.com/blog/2019/09/12/get-started-with-k210-hardware-and-programming-environment/.

[35] DFRobot. Dfr0299 dfplayer mini. url https://wiki.dfrobot.com/DFPlayer$_M ini_S KU_D FR$0299.

[36] DidacticasElectrónicas. Convertidor dc/dc pololu. url https://www.didacticaselectronicas.com/index.php/fuentes-adaptadores/fuentes-variables/conversor-dc-dc-pololu-ajust-elevador-5v-1-2a-fuente-elevador-step-up-boost-conversor-dc-dc-5-voltaje-variable-alimentaci%C3%B3n-poder-pololu-detail.

[37] Sigma electrónca. Módulo de carga tp4056. url https://www.sigmaelectronica.net/producto/tar-tp4056prot/.

[38] Spartan Geek. Fusion 360, el software de ingeniería mecánica de autodesk. url https://spartangeek.com/blog/fusion-360-software-de-ingenieria: :text=Fusion%20360%20es.

[39] Tensorflow. Tensorflow - compatibilidad con gpu. url https://www.tensorflow.org/install/gpu.

[40] Kaggle. Challenges in representation learning: Facial expression recognition challenge. url https://www.kaggle.com/c/challenges-in-representation-learning-facial-expression-recognition-challenge/data.

[41] Dimitry. Image recognition with k210 boards and arduino ide/micropython. url https://www.instructables.com/Transfer-Learning-With-Sipeed-MaiX-and-Arduino-IDE/.

[42] Stuart Rusell and Peter Norvig. *Inteligencia Artificial. Un Enfoque Moderno*. 2004. ISBN 0137903952. URL `http://scholar.google.com/scholar?hl=en{&}btnG=Search{&}q=intitle:Inteligencia+Artificial:+un+enfoque+moderno{#}0`.

[43] Sami Abboud, Shlomi Hanassy, Shelly Levy-Tzedek, Shachar Maidenbaum, and Amir Amedi. EyeMusic: Introducing a 'visual' colorful experience for the blind using auditory sensory substitution. *Restorative Neurology and Neuroscience*, 32 (2):247–257, 2014. ISSN 18783627. doi: 10.3233/RNN-130338.