GOBIERNO DE MÉXICO | CONAHCYT CONSEJO NACIONAL DE HUMANIDADES CIENCIAS Y TECNOLOGÍAS | INFOTEC

BIBLIOTECA INFOTEC
VISTO BUENO DE TRABAJO TERMINAL

Maestría en Sistemas Embebidos
(MSE)

Ciudad de México, a 30 de enero de 2024

UNIDAD DE POSGRADOS
PRESENTE

Por medio de la presente se hace constar que el trabajo de titulación:

"Sistema de apoyo para la identificación de sonidos del entorno para personas con discapacidad auditiva"

Desarrollado por el alumno: **Cristian Yarib Bautista Villalpando**, bajo la asesoría del **Dr. Víctor Manuel Lomas Barrie** cumple con el formato de Biblioteca, así mismo, se ha verificado la correcta citación para la prevención del plagio; por lo cual, se expide la presente autorización para entrega en digital del proyecto terminal al que se ha hecho mención. Se hace constar que el alumno no adeuda materiales de la biblioteca de INFOTEC.

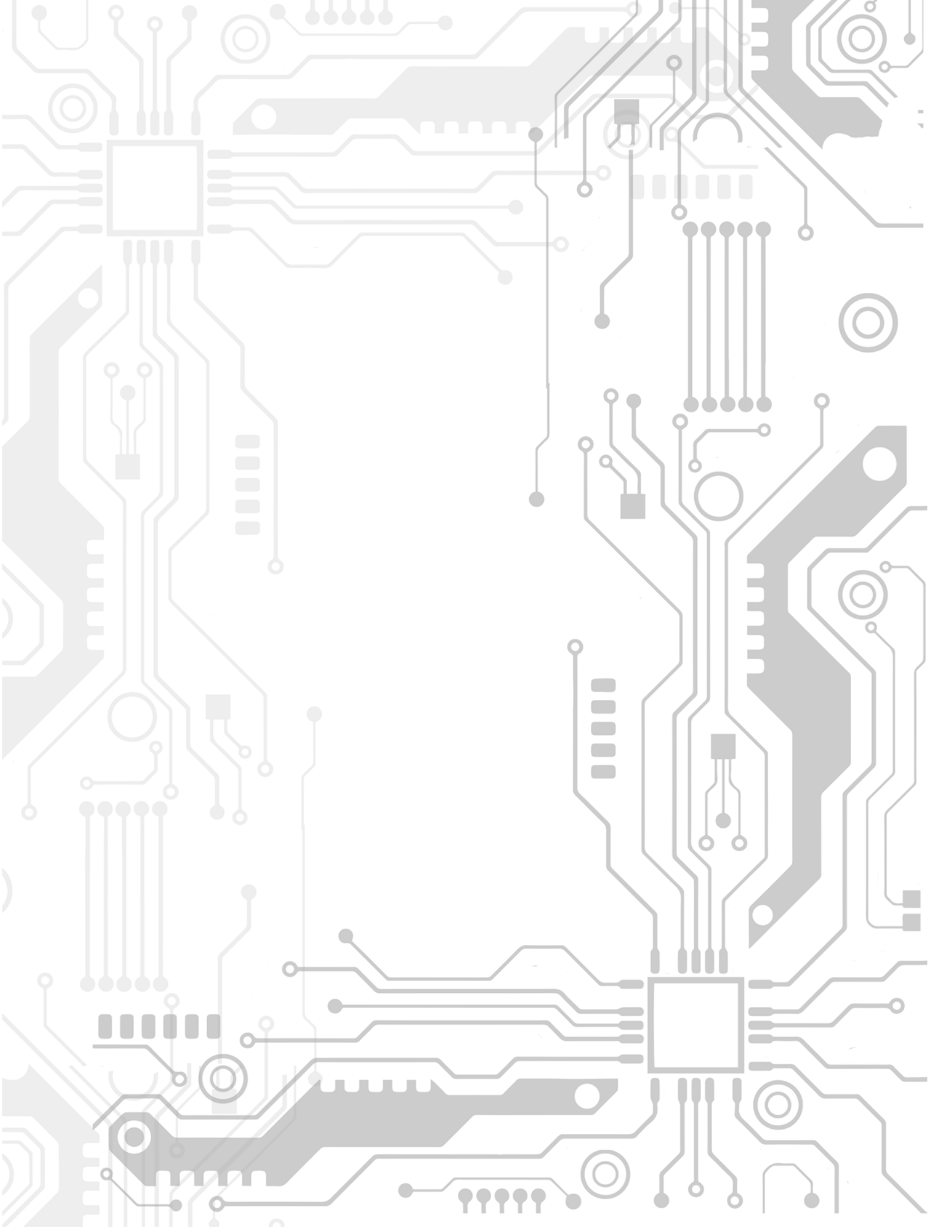
No omito mencionar, que se deberá anexar la presente autorización al inicio de la versión digital del trabajo referido, con el fin de amparar la misma.

Sin más por el momento, aprovecho la ocasión para enviar un cordial saludo.

Mtro. Carlos Josué Lavandeira Portillo
Director Adjunto de Innovación y Conocimiento

CJLP/jah

C.c.p. Felipe Alfonso Delgado Castillo.- Gerente de Capital Humano.- Para su conocimiento.
Cristian Yarib Bautista Villalpando.- Alumno de la Maestría en Sistemas Emebebidos.- Para su conocimiento.



MSE
MAESTRÍA EN SISTEMAS EMBEBIDOS

INFOTEC CENTRO DE INVESTIGACIÓN E INNOVACIÓN EN TECNOLOGÍAS DE LA INFORMACIÓN Y COMUNICACIÓN

DIRECCIÓN ADJUNTA DE INNOVACIÓN Y CONOCIMIENTO
GERENCIA DE CAPITAL HUMANO
POSGRADOS

“Sistema de apoyo para la identificación de sonidos del entorno para personas con discapacidad auditiva”

IMPEMENTACIÓN DE UN PROYECTO
Que para obtener el grado de MAESTRO EN SISTEMAS EMBEBIDOS

Presenta:

Cristian Yarib Bautista Villalpando

Asesor:

Dr. Víctor Manuel Lomas Barrie

Aguascalientes, Ags, noviembre, 2023.

CONACYT

INFOTEC POSGRADOS

Agradecimientos

El presente trabajo es resultado no solo de mi esfuerzo, sino de múltiples personas y circunstancias que se prestaron para que pudiera llevarse a cabo.

Doy gracias a la vida, el universo, Dios, la entropía o cualquier fuerza superior que nos recuerda lo vulnerables y bendecidos que somos, y que nuestro paso temporal por este espacio es una oportunidad de ser felices y mientras tanto aportar, aunque sea un poco, a la felicidad de los demás.

Quiero agradecer a mi esposa Mauren; nuestro camino de casados comenzó casi a la par de estos estudios de maestría, por lo que fue una acompañante constante de este proceso. Gracias por su consejo (que al ser médica fue tanto técnico como alentador), su paciencia, cariño y su también sacrificio por apoyar esta meta personal.

Gracias a mi familia; a mi madre Martha y mi padre Bernabé (cuya presencia sigue viva) por formarme con disciplina y voluntad, además de tener su apoyo constante a lo largo de mi camino; a mis hermanos Lily, Lalo y Charly, cuyas vidas son una inspiración para mí; su visión del mundo y su tesón para superar los momentos más complicados son enseñanza constante.

Agradezco a mis amigos de vida, que directa e indirectamente me motivaron a emprender esta travesía. Personas íntegras y con deseo constante de superación. Uno toma empuje de los que le rodean y crece cuando los otros crecen.

Agradezco a mi asesor el Dr. Víctor que, a un correo de distancia y una charla por videollamada, accedió a ser mi asesor con confianza ciega. Agradezco su tiempo y disposición constante; su guía, ideas y voluntad por encontrar alternativas para el éxito del proyecto.

A mi compañero de maestría Darío, que cuando nada más quedamos dos, reafirmamos que íbamos a terminar. Gracias por su apoyo.

Agradezco a INFOTEC y las personas que lo conforman, quienes estuvieron dispuestas a escuchar y resolver las inquietudes que me acontecieron a lo largo de la maestría. A las y los docentes que aportaron su conocimiento y asesoría en diferentes aspectos.

Finalmente, agradezco al lector que llegue a este trabajo de tesis por su interés en ella. Espero que pueda encontrar claridad y aportaciones para sus futuros proyectos.

Tabla de contenido

Índice de ilustraciones

Índice de cuadros

Siglas y abreviaturas

ABC	Architecture Business Cycle (ciclo empresarial de la arquitectura)
ANN	Artificial Neural Networks (redes neuronales artificiales)
CNN	Convolutional Neural Networks (redes neuronales convolucionales)
DHH	Deaf and Hard of Hearing (sordos y con problemas de audición)
DOA	Direction of Arrival (dirección de llegada del sonido)
HCI	Human-Computer Interaction (interacción humano-computadora)
MEMS	Micro-Electromechanical System (sistema micro-electromecánico)
ML	Machine Learning (aprendizaje de máquina)
PCB	Printed Circuit Board (tarjeta de circuito impreso)
SAISE	Sistema de Apoyo para la Identificación de Sonidos del Entorno para personas con discapacidad auditiva
SE	Sistema Embebido
TBD	To Be Determined (a determinar)
WBS	Work Breakdown Structure

Glosario

“D”

Dataset: Conjunto de datos utilizados para entrenar, validar o probar métodos de aprendizaje supervisado**.**

“H”

Hipoacusia: Incapacidad total o parcial para escuchar sonidos en uno o ambos oídos**.**

Háptico: Todo aquello referido al contacto, especialmente cuando éste se usa de manera activa (*Huitzil Muñoz, 2011*).

“T”

Tactón: Patrón vibratorio específico.

Introducción

La hipoacusia es la incapacidad total o parcial para escuchar sonidos en uno o ambos oídos *(National Library of Medicine, 2022)*. Esto conlleva diferentes dificultades para la persona que lo padece y que limitan su capacidad de desarrollar actividades cotidianas, incluso las más simples.

La Organización Mundial de la Salud (OMS) estima que, en el 2018, 6.12% de la población mundial y 6.18% de la población en Latinoamérica tienen pérdida de la audición discapacitante y, de ellos, 7% (34 millones) es población pediátrica. Se calcula que, en el año 2050, más de 900 millones de personas (una de cada 10), sufrirá pérdida de audición discapacitante. Cerca de dos tercios de estas personas viven en países en vías de desarrollo. Además, se ha calculado que el déficit auditivo sin tratamiento representa un costo global de 750,000 millones de dólares anualmente, lo cual incluye costos del sector salud (excluyendo costo de los dispositivos para la audición), apoyo educativo, pérdida de productividad y costos sociales. *(Zavala-Vargas & García, 2018)*

En México, aproximadamente el 4.9% de la población sufre de alguna discapacidad. De este valor, el 22% está relacionado con discapacidades auditivas aun usando algún dispositivo de asistencia. *(INEGI, 2020).*

Para poder disminuir el impacto que provoca la discapacidad auditiva en la vida cotidiana de quien la padece, existen diferentes opciones como los audífonos retroauriculares e intraauriculares, implantes cocleares y complementos de estos como sistemas de frecuencia modulada y el bucle magnético; así como ayudas visuales como la subtitulación y páneles de avisos luminosos. No todas estas opciones son de carácter portable. De las investigaciones tecnológicas recientes y que ahondan en alternativas de los ya mencionados, hay estudios que abordan el uso de dispositivos portables y la preferencia que provocan en los usuarios sordos o hipoacúsicos. La investigación de *(Findlater et al., 2019),* refleja que las personas usuarias se inclinan por dispositivos de detección de “advertencias sonoras” así como para apoyo en la comunicación, tanto a nivel háptico como visual.

Es por lo anterior que se propone el Sistema de Apoyo para la Identificación de Sonidos del Entorno para personas con discapacidad auditiva (SAISE), que consiste en un dispositivo portable y discreto en forma de audífonos (o simil), que contiene dos micrófonos MEMS (uno en cada auricular), capaz de reconocer cuatro sonidos característicos del entorno relacionados a “emergencia” (claxon, grito, sirenas y llanto de bebé) y ofrecer una respuesta háptica identificativa a cada uno (patrón de vibración o *tactón*), que permita al usuario ser consciente del entorno actual y reaccionar en caso de ser necesario. Además, el SE aporta información sobre la dirección proveniente del sonido mediante la intensidad de las vibraciones.

Para lograr el comportamiento descrito se recurrió a métodos de aprendizaje automático supervisado, haciendo principal hincapié en los algoritmos de redes neuronales artificiales (ANN; por sus siglas en inglés) y en su optimización para poder ser ejecutados dentro de dispositivos de bajos recursos computacionales (lejanos a la nube o cómputo de grandes capacidades) como lo son los SE, tendencia conocida como *Machine Learning at the edge (Subramani & Araújo, 2022*). Se proponen las cuatro clases mencionadas como punto de partida para esta investigación, buscando un equilibrio entre el tiempo de creación de los conjuntos de datos necesarios para el entrenamiento y la exactitud esperada del sistema, con la posibilidad de ir enriqueciendo en un futuro el dispositivo, aunque ya fuera del alcance de este documento.

Como metodología se planteó un modelo en espiral con tres iteraciones, dentro de las cuales se encuentras las etapas del ciclo de vida de sistemas (SDLC por sus siglas en inglés): planeación, análisis, construcción y finalmente pruebas y evaluación.

Los resultados obtenidos fueron el diseño e implementación de un prototipo portable de los audífonos (sin ser un producto comerciable) capaz de detectar los diferentes sonidos propuestos con una exactitud promedio de alrededor del **77% entre las clases**, y con un peor escenario de *score F1* de super clase (es o no es emergencia) mayor al **92%**. Esto se logró mediante la creación de un conjunto de datos propio para el proyecto para posteriormente realizar el entrenamiento y

prueba de una serie de arquitecturas de CNN que lograran el despeño buscado y cumplieran las limitaciones de memoria del microcontrolador utilizado. También se logró la determinación de **cinco variantes de dirección** de llegada de sonido (DOA por sus siglas en inglés). Además, el sistema fue probado por una persona con sordera total cuya retroalimentación dio pauta para mejoras en el diseño.

El trabajo futuro para esta investigación pudiera continuar en la creación de un conjunto de datos de entrenamiento que no sea de audios existentes, sino que sea información y características capturadas propiamente por los micrófonos MEMS, para que la red aprenda de las características de los mismos; la posibilidad de crear un diseño de conjunto de datos estereofónico; la implementación de redes neuronales recurrentes para obtener métricas comparativas de desempeño; la creación de una interfaz donde la persona usuaria pueda seleccionar y personalizar características del sistema: selección de sonidos a detectar, sensibilidad de la detección, actualización de nuevos sonidos desde la nube, entre otros. Muchas posibilidades se abren a partir del presente trabajo y vale la pena mencionarlas.

El documento está organizado de la siguiente manera: Capitulo 1, donde se podrá encontrar el planteamiento del problema, el estado del arte, objetivos y metas, así como otras herramientas de análisis del proyecto. En el Capítulo 2 se encuentra la metodología; el análisis y diseño del sistema embebido. El Capítulo 3 refleja las implementaciones realizadas y los resultados obtenidos; ahí se podrán encontrar detalles de los algoritmos, modelos utilizados, pruebas, diagramas, diseños finales, y rendimiento. Finalmente, el documento cierra con las conclusiones donde se puede encontrar la discusión y más detalle sobre trabajo futuro.

Capítulo 1
Planteamiento del Proyecto

Capítulo 1. Planteamiento del Proyecto

El presente capítulo aborda diversos temas como antecedentes, estado del arte, análisis del Ciclo Empresarial de la Arquitectura (ABC por sus siglas en inglés), planteamiento del problema, objetivos, metas y alcances. Se describen investigaciones relacionadas a dispositivos de apoyo para personas con discapacidad auditiva, así como la aplicación de aprendizaje automático en sistemas embebidos. Se muestra información de una década previa a la actualidad.

1.1 Antecedentes y estado del arte

Existen en la actualidad varios desarrollos tecnológicos enfocados en el apoyo a la discapacidad auditiva y que están enfocadas a su distinto nivel de pérdida ya sea leve, moderado, grave o profunda.

Dentro de estos dispositivos se encuentran: despertadores de vibración, equipos auxiliares de amplificación para TV y música, intercomunicadores, sistemas de alerta con destellos visuales, aparatos auditivos retroauriculares e intraauriculares, implante coclear, bucle auditivo y otros. De estos dispositivos, la mayoría son ya comercializados con varios años en el mercado, pero son pocas las soluciones que hacen uso del aprendizaje automático para la clasificación de sonidos, esto debido a que anteriormente los dispositivos de cómputo aún estaban limitados en capacidades para poder utilizar este tipo de algoritmos, más aún en alguna aplicación portable. Con el continuo avance de la reducción del transistor en el diseño de circuitos integrados, este tipo de perspectivas de aplicación es cada vez más factible.

Muchos de los dispositivos en el mercado están enfocados a la pérdida leve o moderada, como lo son los aparatos auditivos; por otro lado, tienden a ser fijos o apoyados en la percepción visual del individuo para su funcionamiento. Son pocos los que logran la combinación de diferentes factores como la respuesta háptica, la portabilidad, el reconocimiento sonoro inteligente y el enfoque para el uso en caso de pérdida grave o profunda y que no requiera un procedimiento quirúrgico invasivo.

Debido a las limitantes descritas de los diversos productos, diferentes investigaciones han sido desarrolladas a lo largo de los últimos años para enriquecer las soluciones existentes con tecnología más reciente y nuevas perspectivas como lo son las técnicas de ML y en específico el aprendizaje automático; que como ya se mencionó, ha sido posible gracias a un avance significativo en las herramientas de cómputo. Además de abordar temas de implementaciones tecnológicas, se han hecho análisis previos para saber qué tipo de soluciones buscan los usuarios (personas con esta discapacidad) y cuáles de ellas tienen más impacto en sus vidas.

En un estudio realizado por *(Findlater et al., 2019)* y publicado en la Conference on Human Factors in Computing Systems Proceedings (CHI 2019), se presentan los resultados de una encuesta en línea contestada por 201 personas (la mayoría de Estado Unidos), en un rango de edades de 20 a 94 años y una media de edad de 42, con problemas de sordera o dificultades severas para escuchar y usando actualmente un dispositivo de apoyo (aparato auditivo, implante coclear o ambos). El objetivo era conocer las preferencias de posibles soluciones en diversos dispositivos (celular inteligente, reloj inteligente y visores para uso en la cabeza) y tipos de respuesta (visual y háptica) como apoyo para la percepción de sonidos y la comunicación. Los resultados reflejaron que al menos el 74.1% estaba “muy interesado” en la percepción del sonido. Las características de sonido más buscadas eran volumen, fuente del sonido, dirección, duración y patrón de tono. De estos, los más concurrentes fueron fuente del sonido y dirección. La preferencia de sonido clasificado como “alerta urgente” fue una de las más altas. Así mismo, el tipo de respuesta preferido resultó ser visual para teléfonos inteligente, háptica para reloj de mano inteligente, o ambos en una combinación adecuada. También se reflejó el deseo de filtrado y selección personalizada de los sonidos a reconocer dependiendo del lugar donde se encuentran y poder deshabilitarlo por completo si así lo desean (modo avión).

Entre otros estudios sobre preferencias de personas con hipoacusia y sordera se encuentra el realizado por el Instituto de Tecnología de Rochester, Nueva York (*Yeung et al., 2020*), que reporta resultados similares a *(Findlater et al.,*

2019). De entre ellos destaca la observación de que las preferencias varían dependiendo de factores demográficos, la importancia de la locación, tipo de fuente de sonido y fiabilidad, además de denotar que los usuarios pierden la percepción de algunos sonidos aun usando un aparato auditivo y dependen de otros para poder identificarlos. El estudio también hace referencia a las locaciones importantes para los usuarios y sus sonidos característicos y la posibilidad de poder personalizarlos dependiendo de ello; la casa, el trabajo y principalmente mientras están móviles (caminando o manejando). Todos estos factores llevan a algunas características comunes deseables en un dispositivo como lo son cuál es la fuente sonora, de dónde viene, cuán urgente es y qué tan fiable es la clasificación (qué tan seguro está el dispositivo del sonido detectado).

Otros estudios relacionados como el de (*Manchaiah et al., 2017*), muestran el deseo de los consumidores de dispositivos de apoyo de verse más involucrados en sus decisiones de cuidado de salud, lo cual puede ser posible gracias al incremento de capacidades de la tecnología (como en los *smartphones*), llevando a crear dispositivos más acordes, accesibles, económicos y con mejores características.

Además de los requerimientos extraídos mediante entrevistas en los estudios previos y para lograr desarrollos tecnológicos más asertivos para los usuarios con esta discapacidad, varias investigaciones aplicaron estudios del tipo *wizard-of-oz,* los cuales tomaban de primera mano las reacciones y opiniones de esta población haciéndolos interactuar con interfaces que emulaban la funcionalidad de futuros sistemas de apoyo para identificar sonidos del entorno. Este tipo de estudios los podemos encontrar en (*Goodman et al., 2020*), (*Jain et al., 2019*). Sus resultados varios puntos significativos para ser tomados en cuenta en las aplicaciones de esta tecnología: vibraciones constantes para las notificaciones no son deseadas y un filtrado pertinente en los sonidos es necesario; la importancia de combinar retroalimentación tanto visual como táctil por parte del dispositivo, resaltando a la vibración ya que provoca una reacción más inmediata; la utilidad en los patrones vibratorios (*tactones*) y que su exploración podría ser una dirección prometedora para futuros trabajos, aunque considerando que la cantidad de información que

aporta al usuario sobre el sonido es mucho menor; refuerzan la idea del interés de los usuarios en sonidos de alarma y alertas; y el deseo de conocer el tipo de sonido y su locación proveniente.

Basado en lo anterior y ante la oportunidad de integrar las nuevas tendencias tecnológicas en soluciones a un problema común y constante como lo es la inclusión en la discapacidad, existen diversos desarrollos que han aportado positivamente a estos sistemas de apoyo para la identificación de sonidos del ambiente. Estos sistemas son aplicaciones dentro del área *Human-Computer Interaction* (HCI), rama que estudia la manera en la cual la tecnología influye en el trabajo y actividades humanas (*Blanton et al., 2009*). Las implementaciones encontradas giran alrededor de tres dispositivos: pantallas montadas en la cabeza como la investigación de (J*ain et al., 2015*); teléfonos y relojes inteligentes. Esta investigación se relaciona más con estos últimos, por lo que se describen algunos de estos desarrollos a continuación. El trabajo realizado por (*Dhruv Jain et al., 2020*), compara el uso de diferentes redes neuronales convolucionales (CNN: *MobileNet, Inception, ResNet-lite y VGG-lite*) ya existentes (con algunas adaptaciones) para la clasificación de 20 clases (diferenciando tres sonidos prioritarios de emergencia) de cuatro arquitecturas portables y con una distribución de tareas específicas (solo reloj inteligente, reloj inteligente-celular, reloj inteligente-celular-nube, reloj inteligente-nube). Sus resultados arrojan que la mejor combinación fue la CNN *VGG-lite*, que tuvo un rendimiento similar a las más novedosas NN de la actualidad para dispositivos no portables con una exactitud promedio de 81.2% (σ=5.8%) sobre las 20 clases y de 97.6% (σ=1.7%) sólo usando las tres clases más prioritarias, pero fue la que tuvo más latencia (3397ms, σ=42ms); y las arquitecturas que mejor concesión mostraron entre los parámetros de sistema: uso de CPU, vida de batería y memoria, fueron la reloj inteligente-celular-nube y reloj inteligente-celular. Esta investigación no abordó el desarrollo de los SE, sino que usaron productos existentes en el mercado.

ProtoSound es otra aplicación móvil desarrollada por (*Jain et al., 2022*) que propone una solución diferente para estos sistemas de apoyo, mediante una técnica que involucra el entrenamiento para nuevas clases usando pocas grabaciones del

sonido deseado (abordando el modelo mediante *meta learning,* una técnica dentro de *machine learning* donde se “aprende a aprender”, basada en conjuntos de datos pequeños y utilizando información previamente aprendida), cubriendo así uno de los requerimientos más deseados por los usuarios que es la personalización. Previo al desarrollo de *ProtoSound*, los investigadores realizaron una encuesta a gran escala con 472 participantes DHH (*Deaf and Hard of Hearing*), para obtener y reforzar los requerimientos para la aplicación. Una vez claros estos, realizaron tres experimentos: evaluaciones cuantitativas en dos conjuntos de sonidos del mundo real y un estudio de campo. La diferencia entre los conjuntos era que uno fue grabado por personas oyentes mientras que el otro fue grabado por usuarios DHH. La exactitud en la clasificación de *ProtoSound* usando la arquitectura de CNN *MobileNetV2* en ambos conjuntos, rondó valores muy cercanos al 90%, resultados significativos siendo que uno de los conjuntos fue grabado por personas con la discapacidad, actividad evidentemente compleja para ellos. Para el tercer experimento, desplegaron la aplicación móvil para realizar un entrenamiento por usuario en entornos y tiempo real, tarea llevada a cabo por participantes oyentes teniendo en mente la premisa de la confiabilidad (en que pudieran escuchar sonidos del mundo real y evaluar mejor la exactitud de clasificación por el dispositivo) antes de intentar el mismo procedimiento en participantes DHH. Los resultados de este despliegue fueron buenos, pudiendo entrenar el modelo en el equipo con poco esfuerzo y pocas muestras por parte del usuario oyente, logrando la correcta clasificación del mismo sonido en diferentes contextos (ej. casa, restaurantes, parques, calles). Sin embargo, también reportan fallas debidos a errores de grabación y etiquetado, lo que apunta a la necesidad de desarrollar mejores interfaces en el futuro.

Como se ha descrito, dos de las principales características que los usuarios DDH buscan en un dispositivo es conocer cuál es la fuente sonora del sonido (y su tasa de fiabilidad) y de dónde proviene. Relacionado con la última característica, diversas investigaciones han abordado también este reto. Entre ellas destaca (*Küçük & Panahi, 2019*), quienes proponen un método basado en redes neuronales profundas (DNN por sus siglas en inglés) para la detección de la dirección de llegada

del habla, desarrollando una aplicación para teléfono inteligente con SO Android (con dos micrófonos integrados) para ello. Las características consideradas para alimentar la DNN son la magnitud y la fase de la señal de la voz. Para evaluar el desempeño, usan diferentes tipos de ruido (muchas voces, maquinaria, tráfico) y con tres diferentes relaciones señal/ruido (0, 5, 10 dB). Los resultados muestran una exactitud en la detección del ángulo de llegada del habla (resolución de 20° en un rango de 0° a 180°) de 66.25%, una mejora a otros métodos propuestos, pero aún con muchas posibilidades de desarrollo por delante.

Métodos más recientes como el propuesto por (*Tokgoz et al., 2020*) usan una modificación de una técnica ampliamente utilizada para la detección de llegada de una fuente de habla: GCC (*generalized cross-correlation*), pero usando tres micrófonos integrados. Este método es implementado en celular inteligente con SO Android. Los resultados muestran un buen desempeño con una RMSE (*root-mean square error*) de 7.13°.

Todos los desarrollos descritos han sido realizados fuera de México y sólo se encontró uno dentro de Latinoamérica: (*Fanzeres et al., 2018*) que presenta la creación de una aplicación para celular para el reconocimiento de sonidos del ambiente mediante el uso de diferentes algoritmos de clasificación: vecinos cercanos basados en distancia euclidiana, *naive* de Bayes, redes de Bayes y árboles de decisión, con una tasa de exactitud en la clasificación del 92.7, 88.7%, 89.7% y 92.3% respectivamente, mostrando resultados prometedores . Dentro del país sólo se han encontrado propuestas sin que haya información de haber sido consolidadas. En este caso podemos encontrar la propuesta de (Iván et al., 2015) relacionada también con una aplicación para celular.

Presentado lo anterior, la propuesta de este trabajo aborda los siguientes requerimientos: la detección del tipo de sonido ambiente, su dirección de llegada estimada, la respuesta táctil y la portabilidad.

También se puede notar los pocos desarrollos alrededor de dispositivos embebidos de tarea específica (al considerar al celular como un dispositivo multitareas de mayores capacidades) por lo que plantear la aplicación de

metodologías de ML en estos presenta una aportación que puede ser relevante para otros desarrollos.

Los capítulos restantes del documento presentan la metodología, requerimientos y arquitectura (capítulo 2), implementación y resultados (capítulo 3), discusión y conclusiones (capítulo final).

1.2 Estudio de factibilidad

Dentro del estudio de factibilidad y habiendo buscado a través de internet, no se encontró ningún dispositivo en estado de comercialización que use la tecnología descrita en el estado del arte, además de ninguna investigación relacionada dentro de México.

Poniendo en contexto la situación del país, el censo del 2020 muestra que al menos 1.3 millones de personas tienen dificultades para escuchar aun utilizando un aparato auditivo *(INEGI, 2020)*. Por la demanda visualizada, el desarrollo de este proyecto puede resultar de gran ayuda para la integración de la población mexicana con esta discapacidad.

Análisis basado en el ABC (*Architecture Business Cycle*)

Comercial

Existiendo una población alta de personas con algún padecimiento de sordera crónica o hipoacusia en México, se puede que el sistema tenga un amplio uso en el mercado. Así mismo, se pueden buscar acuerdos con la Secretaría de Salud así como asociaciones civiles para que sean parte de algún programa de apoyo para esta población.

Se puede plantear una cierta cantidad de productos sin costo para su prueba y evaluación, y así, crear un antecedente de uso previo a la venta.

Tiempo de vida

El tiempo de vida del producto, considerando el uso constante, depende en gran medida del cuidado que la persona usuaria tenga. Con un uso adecuado, se considera al menos de 2 años con una vida de batería al 80%. Este periodo puede variar de acuerdo con el desempeño de la batería. Si se habilita su reemplazo y al ser un dispositivo completamente de estado sólido, se considera una expectativa de al menos 5 años. Un punto para considerar es el tiempo de vida del micrófono seleccionado, existiendo algunos en el mercado con un lapso mayor al indicado.

Segmentación de mercado

El mercado al que va dirigido el producto es muy específico para personas con hipoacusia que va de moderada a severa y sordera total. Puede ser usado por cualquier persona mayor de aproximadamente 6 años, esto basado en las etapas típicas del desarrollo de la infancia (final de la segunda infancia e inicio de la tercera de acuerdo con (*Papalia et al., 2009*)), donde se espera que para esa edad el infante tiene una capacidad suficiente en las etapas de lógica, lenguaje y memoria para el uso del dispositivo.

Desempeño

Se espera que el dispositivo pueda clasificar de manera correcta con una exactitud de al menos 75% (promedio general de las clases, tomando como referencia los resultados descritos en las investigaciones del estado del arte), y que la persona usuaria, después de un entrenamiento corto de aproximadamente 2 hrs, pueda usarlo sin restricciones y le apoye para identificar los primeros elementos del entorno. Se toma en cuenta este tiempo a partir de dos sesiones de 50 minutos, una de presentación y la otra de uso en campo por la persona.

Usabilidad

El dispositivo se planea portable con forma de audífonos con un tiempo de autonomía de trabajo de 8 horas continuas.

Seguridad a la salud

El dispositivo por sí solo no presenta ningún riesgo para el usuario, no obstante, se debe considerar la calidad de la batería y un correcto diseño en el sistema de carga para evitar cualquier sobre calentamiento. Un riesgo pudiera ser ocasionado por el mal funcionamiento en la detección o no detección de sonidos, siendo entonces la omisión a un sonido de emergencia importante lo que pudiera conllevar a un cierto riesgo.

Confiabilidad

Se espera para la primera etapa una confiabilidad del igual o superior al 75% como promedio general de clases y mayor al 85% en la detección binaria de hay o no un sonido de emergencia.

Evaluación del sistema

Para el objetivo específico que es la detección de sonidos de emergencia del entorno, se pueden hacer ensayos controlados aportando diferentes estímulos sonoros de diversas fuentes y contar las veces que asigna la respuesta correcta al estímulo indicado.

Una medida similar se puede hacer durante los casos de prueba reales realizados con la participación de 5 personas con capacidades auditivas típicas y 5 con la discapacidad. Se puede contabilizar el número de veces que el usuario logró interpretar de manera correcta el sonido reproducido y si, además este coincide con el detectado por el dispositivo.

Respecto a la evaluación relacionada a la practicidad de uso, se podría hacer un análisis de satisfacción del usuario pasados los periodos de pruebas.

Desde la industria

No hay contacto directo con la industria, pero se considera como un estándar a seguir en caso de comercialización, la Norma Oficial Mexicana para apoyo a discapacidades como:

- NOM-015-SSA3-2018, para la atención integral a personas con discapacidad
- NOM-008-SEGOB-2015, Personas con discapacidad. - Acciones de prevención y condiciones de seguridad en materia de protección civil en situación de emergencia o desastre

1.3 Planteamiento

Hay dos líneas significativas en el desarrollo de este proyecto. Por un lado, aportar positivamente en el problema de salud tanto global como nacional que conlleva el padecimiento de alguna discapacidad, en específico la sordera e hipoacusia. En el apartado pasado se presentaron algunas estadísticas que muestran el impacto que genera, lo cual hace que desarrollos de este tipo sean pertinentes y necesarios para lograr la inclusión de estas personas. Esto lleva a la presente propuesta que involucra la detección automática de sonidos mediante aprendizaje profundo dentro de un sistema embebido portable para compensar la pérdida auditiva de las personas con esta discapacidad.

Por el otro lado, la investigación alrededor del uso de redes neuronales artificiales en dispositivos embebidos no es un problema trivial *(Zhang & Li, 2023)*. Al ser de recursos limitados, el diseño y aplicación de algoritmos de aprendizaje profundo en sistemas embebidos presenta un reto por sí solo, y las soluciones con estas herramientas sumado a la implementación de los procesos periféricos necesarios para su uso (tratamientos de los datos, preprocesamientos, calibraciones, etc.), suma un aporte significativo a los esfuerzos mundiales enfocados al aprovechamiento de estas técnicas. Se ha visto la utilidad de las ANN

para la predicción y clasificación de datos en sistemas de computadoras personales, *clusters* y servicios en la nube; con la posibilidad de implementarlas en dispositivos de bajos recursos, se puede esperar buenos resultados comparado con otras técnicas usadas con anterioridad.

1.4 Objetivos

1.4.1 Objetivo general

Desarrollar el Sistema de Apoyo para la Identificación de Sonidos del Entorno para personas con discapacidad auditiva (SAISE), para que sea portable y discreto y que, mediante inteligencia artificial, ayude a dichas personas a percibir la existencia de sonidos identificados como “emergencia”. Una vez reconocido alguno de estos sonidos, el sistema debe ofrecer una respuesta háptica identificativa para que el usuario pueda reaccionar al estímulo, así como aportar información de la dirección de llegada del sonido.

1.4.2 Objetivos específicos

- Poder reconocer al menos 4 patrones sonoros de “emergencia”: sirena, grito, llanto de bebé y claxon con una exactitud promedio de al menos 75%.
- Ofrecer un tiempo de latencia total (desde captura de sonido hasta notificación al usuario) de máximo 2 segundos. Esto tomando como punto de comparación el trabajo de (*Dhruv Jain et al., 2020*) cuya latencia máxima es de 5 segundos en un dispositivo de capacidades similares.
- Creación del conjunto de datos necesario para el entrenamiento de la CNN.
- Diseñar y validar diferentes arquitecturas de CNN.
- Diseñar una PCB satisfactoria para lograr un sistema embebido portable.
- Ofrecer información de la dirección de llegada del sonido.

- Realizar pruebas de campo con personas con y sin discapacidad auditiva para obtener puntos de mejora.

1.5 Metas

- Un prototipo funcional portable previo a la etapa de comercialización para realizar pruebas en campo y de satisfacción de usuario con una tasa promedio de fiabilidad en la clasificación de al menos 75%.
- La creación de un artículo de investigación con la intención de presentarlo en algún congreso de salud, tecnología o áreas del conocimiento relacionadas a la interacción humano-computadora.
- La aportación de un conjunto de datos abierto a la comunidad de investigación para que pueda ser un punto de comparación para futuros desarrollos.
- La aportación de algoritmos de preprocesamiento de datos digitales de audio para su uso en CNN.
- La aportación de evaluaciones de diferentes arquitecturas de CNN así como los diferentes parámetros de entrenamiento.
- Obtención de grado de Maestro en Sistemas Embebidos del sustentante en los tiempos pertinentes.

Capítulo 2
Metodología

Capítulo 2. Metodología

El presente capítulo describe la forma en la que se abordará el proyecto y las necesidades técnicas que lo rodean. Se seguirá un modelo iterativo en espiral del ciclo de desarrollo de un sistema embebido combinado con algunos principios del PMBOK Guide (Project Management Institute, 2017), como lo son la gestión de la integración del proyecto (consideraciones y definiciones al momento del arranque), la colección de requerimientos y definición de alcances, la estructura de desglose del trabajo (WBS por sus siglas en inglés) y la metodología para la gestión del calendario.

Se hablará también de los requerimientos, diseño arquitectónico, casos de uso y casos de prueba. También se indican los documentos relacionados donde se puede encontrar más información de estos apartados.

En el último título, se describe la técnica de diseño y prueba del modelo de aprendizaje profundo; cómo se obtendrán los datos de los diferentes conjuntos necesarios para el modelo (entrenamiento, validación y prueba), el preprocesamiento a los archivos de audio y el diseño de la red neuronal a usar. Se describen también las herramientas de cómputo que se usarán para tal objetivo.

2.1 Descripción de las iteraciones (WBS)

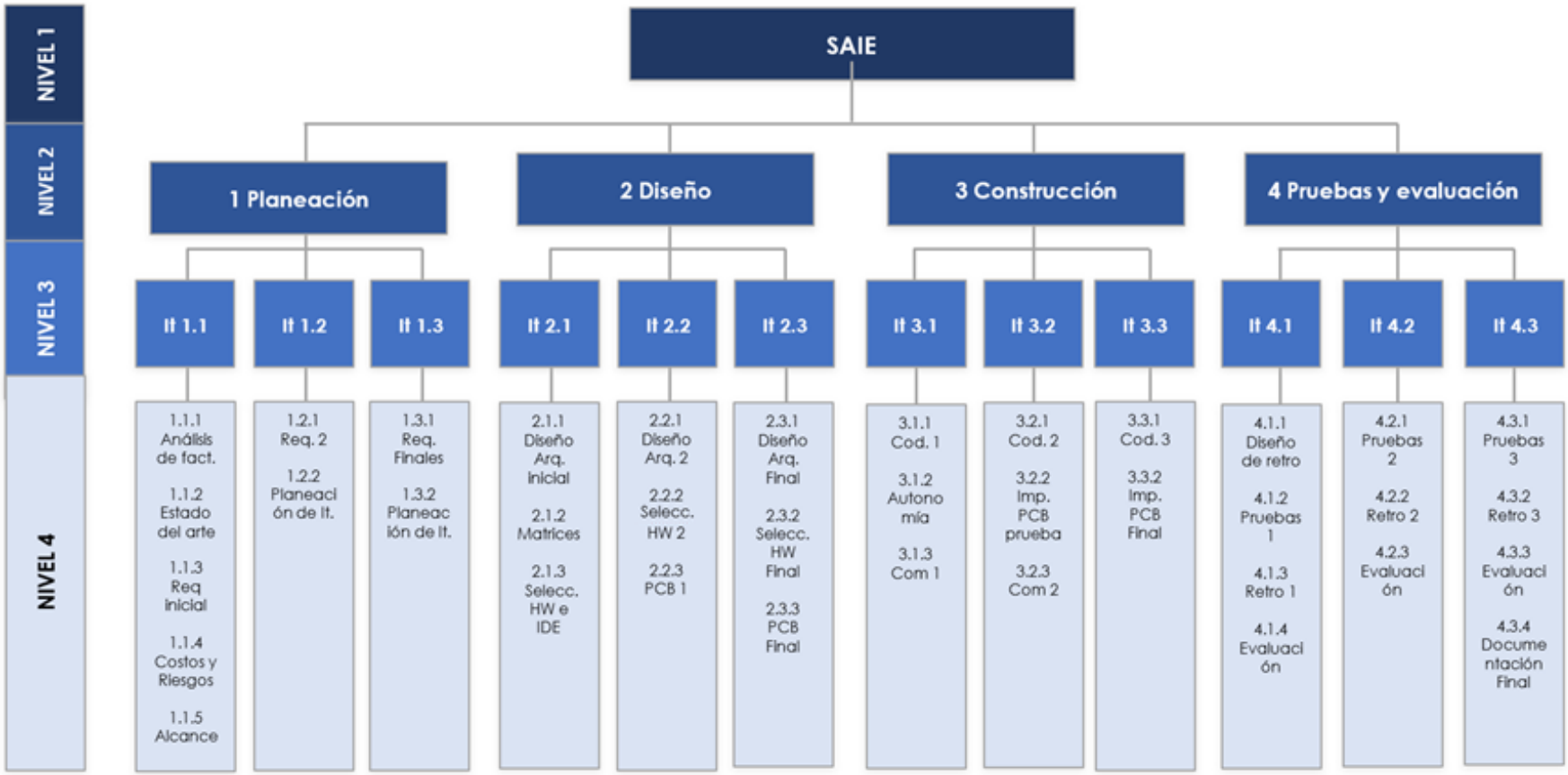


Ilustración 1.- WBS (Work Breakdown Structure). Las 4 actividades generales y sus iteraciones (elaboración propia).

ID	Nombre	Descripción	Complejidad
1	**Planeación**		
1.1	**It 1.1**	**Iteración 1**	
1.1.1	Análisis de fact.	Estudio de factibilidad. Encuestas a personas con el padecimiento y especialistas en audiología.	Media
1.1.2	Estado del arte	Revisión de los desarrollos actuales relacionados a la investigación.	Media
1.1.3	Req. inicial	Determinación de los requerimientos iniciales del sistema.	Alta
1.1.4	Costos y Riesgos	Análisis de costos y riesgos basados en los requerimientos, recursos y personal involucrado.	Alta
1.1.5	Alcance	Determinación de las metas y limitaciones.	Baja
1.2	**It 1.2**	**Iteración 2**	
1.2.1	Req. 2	Readaptación de requerimientos basados en la retroalimentación de la etapa anterior.	Media
1.2.2	Planeación de It.	Planeación de los tiempos de la iteración número 2.	Baja
1.3	**It 1.3**	**Iteración 3**	
1.3.1	Req. Finales	Readaptación de requerimientos y determinación final basado en la retroalimentación de la etapa anterior.	Baja
1.3.1	Planeación de It.	Planeación de los tiempos de la iteración número 3.	Baja
2	**Diseño**		
2.1	**It 1.1**	**Iteración 1**	
2.1.1	Diseño Arq. Inicial	Diseño arquitectónico basado en requerimientos.	Alta
2.1.2	Matrices	Matrices de componentes para análisis de HW y SW para la etapa de desarrollo.	Alta
2.1.3	Selecc. HW e IDE	Selección de HW y entorno de programación y desarrollo. Cotización y compra de materiales.	Alta
2.2	**It 1.2**	**Iteración 2**	
2.2.1	Diseño Arq. 2	Diseño arquitectónico basado en requerimientos actualizados.	Alta
2.2.2	Selecc. HW 2	Nueva selección de HW en caso de ser necesario por nuevos requerimientos.	Baja
2.2.3	PCB 1	Diseño inicial de PCB.	Media
2.3	**It 1.3**	**Iteración 3**	
2.3.1	Diseño Arq. Final	Diseño arquitectónico basado en requerimientos finales.	Baja
2.3.2	Selecc. HW Final	Selección final de HW en caso de ser necesario por nuevos requerimientos.	Baja
2.3.3	PCB Final	Diseño de PCB para prototipo final.	Media
3	**Construcción**		
3.1	**It 1.1**	**Iteración 1**	
3.1.1	Cod. 1	Creación de base de datos para análisis de Sonido 1: Llanto de bebé. Extracción de características, entrenamiento y codificación de embebido con capacidades de aprendizaje. Tarea realizada en tarjeta de desarrollo.	Muy alta
3.1.2	Autonomía	Integración de Sistema de batería y carga para el sistema basado en tarjeta de desarrollo. Análisis de consumo de energía.	Alta
3.1.3	Com 1	Implementación de la comunicación USB para interacción con SE.	Media

ID	Nombre	Descripción	Complejidad
3.2	**It 1.2**	**Iteración 2**	
3.2.1	Cod. 2	Creación de base de datos para análisis de Sonido 2 y 3: Ambulancia y Grito. Extracción de características, entrenamiento y codificación de embebido con capacidades de aprendizaje basado en funcionamientos previos. Tarea realizada en tarjeta de desarrollo.	Muy Alta
3.2.2	Imp. PCB prueba	Impresión y ensamblado de PCB de prueba para trabar con la codificación ya realizada en paralelo.	Alta
3.2.3	Com 2	Posibles correcciones en el sistema de comunicación PC-SE	Baja
3.3	**It 1.3**	**Iteración 3**	
3.3.1	Cod. 3	Creación de base de datos para análisis de Sonido 4: claxon. Extracción de características, entrenamiento y codificación de embebido con capacidades de aprendizaje basado en funcionamientos previos. Tarea realizada en tarjeta de desarrollo y SE de prueba	Alta
3.3.2	Imp. PCB Final	Impresión de PCB final para prototipo funcional. No se espera en etapa comercializable.	Alta
4	**Pruebas y evaluación**		
4.1	**It 1.1**	**Iteración 1**	
4.1.1	Diseño de retro	Diseño de instrumentos de retroalimentación: cuestionarios, entrevistas con usuarios y expertos.	Media
4.1.2	Pruebas 1	Implementación de pruebas en usuarios con hipoacusia, así como usuarios sin la discapacidad. Ambiente controlado.	Alta
4.1.3	Retro 1	Aplicación de los instrumentos de retroalimentación.	Baja
4.1.4	Evaluación	Análisis y evaluación de los resultados obtenidos, iteración 1.	Media
4.2	**It 1.2**	**Iteración 2**	
4.2.1	Pruebas 2	Implementación de pruebas en usuarios con hipoacusia, así como usuarios sin la discapacidad. Ambiente controlado y campo con tarjeta de desarrollo y SE.	Media
4.2.2	Retro 2	Aplicación de los instrumentos de retroalimentación.	Baja
4.2.3	Evaluación	Análisis y evaluación de los resultados obtenidos, iteración 2.	Media
4.3	**It 1.3**	**Iteración 3**	
4.3.1	Pruebas 3	Implementación de pruebas en usuarios con hipoacusia, así como usuarios sin la discapacidad. Ambiente controlado y campo con sólo el SE.	Media
4.3.2	Retro 3	Aplicación de los instrumentos de retroalimentación.	Baja
4.3.3	Evaluación	Análisis y evaluación de los resultados obtenidos, iteración 3.	Media
4.3.4	Documentación Final	Integración y ordenación de la documentación generada para los obtener los siguientes documentos: tesis de maestría, manual de usuario, manual técnico de funcionamiento.	Alta

Tabla 1.- Descripción de las tareas a realizarse en cada iteración y sección (elaboración propia).

2.2 Requerimientos del sistema

Los requerimientos del sistema embebido (SE) Sistema de Apoyo para la Identificación de Sonidos del Entorno para personas con discapacidad auditiva (SAISE) se registran siguiendo el formato IEEE para que estos sirvan de fundamento para las etapas posteriores de diseño y desarrollo. El alcance que abarca esta especificación es hasta la entrega de un prototipo embebido en un microcontrolador sin ser aún un producto comercializable, dejando a un lado requerimientos de diseño estético, empaquetado y los relacionados a ventas.

2.2.1 Funciones del sistema

1. El SE detectará la aparición de un sonido de “emergencia”. Estos sonidos son específicos como: sirena, grito, llanto de bebé y claxon.
2. Al reconocer la emergencia, el SE avisará al usuario con el problema auditivo mediante una vibración exclusiva y correspondiente al sonido en máximo 2 segundos.
3. El SE es un dispositivo portable. Como referencia puede compararse a unos audífonos supra aurales (TBD).
4. El SE es recargable con un tiempo de vida de batería de 6 h y avisará cuando esté por terminarse.
5. El SE podrá apagarse cuando el usuario así lo desee.

2.2.2 Requerimientos de Usuario

Descripción general de las funcionalidades con las que el usuario final tendrá interacción.

REQ-U-1: El usuario debe poder reconocer a través de vibración táctil en los oídos, ofrecida por el SE, la traducción de sonidos específicos relacionados a “emergencia”, principalmente sonidos de sirena, llanto de bebé, grito y claxon; y reaccionar a tiempo en caso de ser necesario.

REQ-U-2: El usuario debe poder reconocer todos los patrones hápticos ofrecidos por el SE en máximo dos sesiones de entrenamiento de 2 horas.

REQ-U-3: El usuario debe poder usar el sistema al menos 6hrs continuas.

REQ-U-4: El usuario debe poder apagar el dispositivo fácilmente mediante un botón.

REQ-U-5: El usuario debe poder usar el dispositivo como se usan regularmente unos audífonos inalámbricos (TBD).

2.2.3 Requerimientos Funcionales

Funcionalidad 1.- Identificación de patrones sonoros por el SE

REQ-1.1: El SE debe ser capaz de capturar sonidos del entorno mediante un micrófono MEMS analógico o digital (TBD) con patrón de captura omnidireccional con voltaje de trabajo CMOS 3.3V.

REQ-1.2: El SE deberá hacer un acondicionamiento de la señal por hardware, ya sea de impedancia, amplificación y/o filtrado en caso de uso de un micrófono MEMS analógico. (TBD)

REQ-1.3: El SE deberá poder hacer operaciones de acondicionamiento de la señal mediante algoritmos dedicados a ello en caso de ser necesario. (TBD)

REQ-1.4: La adquisición de la señal del SE deberá ser al menos a 16K muestras por segundo con una profundidad de 16 bits para tener una resolución suficiente para su posterior procesamiento.

REQ-1.5: El SE debe poder ofrecer la respuesta háptica como máximo 2 segundos después de que el sonido de emergencia con un volumen significativo sea captado por los micrófonos. El valor de menor latencia posible debe ser buscado a lo largo del diseño del proyecto.

REQ-1.6: El microcontrolador del SE debe trabajar con tecnología de alimentación CMOS de 3.3.V

Funcionalidad 2.- Respuesta háptica

REQ-2.1: El dispositivo de retroalimentación háptico al usuario deberá ser uno de tipo vibrador mecánico de dimensiones máximas de 10mm x 4mm x 4mm y ser capaz de trabajar a 3.3V.

REQ-2.2: El SE debe activar el motor vibrador mediante una señal binaria codificada con una respuesta única y reconocible (tactón) a cada uno de los sonidos en forma de estímulo táctil.

REQ-2.3: El SE no debe ofrecer respuesta vibratoria a sonidos no determinados previamente.

REQ-2.4: La intensidad de la vibración debe ser suficiente para ser detectada por el usuario sin que esto provoque incomodidad.

Funcionalidad 3.- Portabilidad

REQ-4.1: La batería de alimentación del SE deberá cargarse mediante cualquier cargador comercial de 220VAC/120VAC a 5VCD mediante conector USB-C o micro USB. (TBD)

REQ-4.2: El sistema de energía del SE debe poder ofrecer voltaje regulado de alimentación CMOS 3.3V para energizar el sistema.

REQ-4.3: El SE debe ser capaz de mantenerse en la cabeza de forma segura.

REQ-4.4: Las dimensiones del SE deben ser máximas de 10cm x 10cm x 3cm y no debe interferir con la movilidad normal del usuario, siendo cómodo y ergonómico.

2.2.4 Requerimientos No Funcionales

Requerimientos de Desempeño

- Un punto de desempeño importante aparece descrito en el REQ-1.5.

REQ-NFD1.-El SE debe tener una tasa de exactitud de al menos 75%.

Al ser un sistema de respuesta crítica, ya que está relacionado a sonidos de emergencia, esta tasa es la menos aceptable para poder continuar a una etapa posterior de perfeccionamiento para lograr un 90% de fiabilidad, aunque ya no corresponde a los alcances actuales de este desarrollo. Se tolera en alguna medida “falsos positivos” de detección siendo los “falsos negativos” no aceptables.

Requerimientos de Seguridad al Ser Humano

REQ-NFH1.- El SE no debe dañar en ninguna manera al portador por explosión, calentamiento o descarga eléctrica.

REQ-NFH2.- El SE no debe ser sumergido ni expuesto a ningún tipo de líquido de manera constante.

REQ-NFH3.- El SE no debe usarse si la temperatura del exterior excede los 40°C.

REQ-NFH4.- El SE debe considerar en la etapa de perfeccionamiento y previo a su comercialización, la satisfacción a las siguientes normativas:

- NOM-015-SSA3-2018, para la atención integral a personas con discapacidad.

- NOM-008-SEGOB-2015, Personas con discapacidad. - Acciones de prevención y condiciones de seguridad en materia de protección civil en situación de emergencia o desastre.

Requerimientos de Seguridad en los Datos Personales

No aplica. No se maneja información personal ni datos sensibles del usuario.

2.3 Modelo de Casos de Uso

Se describen los casos de uso más significativos del Sistema SAISE: la funcionalidad clave que consiste en la detección de ciertos patrones sonoros catalogados como “emergencia”, así como la salida de este principal proceso que es una secuencia vibratoria para ser detectada de manera háptica por el usuario final. También se describen otros procesos como el proceso de carga de batería.

2.3.1 Descripción de actores

Actor	Sonido del ambiente	**Identificador:** AS
Descripción	El sonido del ambiente iniciará el proceso de captura válida, extracción de características y finalmente reconocimiento. Dependerá si es reconocido o no, la activación del vibrador.	
Características	Tiene un volumen específico necesario para poder ser considerado por el reconocimiento. Sólo 4 sonidos llevarán a la activación del vibrador: llanto de bebé, grito, sirena y claxon El resto de los sonidos que no cumplan el volumen específico o no son parte de los cuatro previos, no tendrán respuesta alguna evidente para el usuario.	
Relación	El sonido del ambiente no es escuchado por el usuario que es el actor “persona con hipoacusia”, por lo que el sistema es el encargado de avisarle en caso de que cualquiera de los “sonidos detectables” son reconocidos por el sistema, realizando la acción de “activación del motor vibrador”.	
Referencias	Este actor inicia el caso de uso “Reconocimiento de sonido en el SE”.	

Tabla 2.- Descripción del actor “Sonido del ambiente” (elaboración propia).

Atributos	
Nombre	**Descripción**
Volumen	Presión sonora de la señal de sonido que ingresará al micrófono del sistema para su reconocimiento.
Fuente	Elemento que provoca el sonido

Tabla 3.- Atributos del actor "Sonido del ambiente" (elaboración propia).

Actor	Persona con hipoacusia	**Identificador:** AP
Descripción	La persona con hipoacusia es el usuario final del sistema, pero no inicia el caso de uso "Reconocimiento de sonido en el SE". Este actor interactúa con el sistema en el caso de uso carga de batería del SE.	
Características	La persona con hipoacusia es una persona con alguna discapacidad auditiva que le impide escuchar correctamente su entorno y es el usuario final del sistema. Se considera que tiene sensibilidad en sus oídos ya que es dónde el sistema ofrece sus respuestas hápticas. También se considera que el resto de sus aptitudes cognitivas no tienen ninguna afectación por lo que puede hacer uso de dispositivos de cómputo necesarios para los casos de uso donde participa.	
Relación	El actor "persona con hipoacusia" usa el sistema en su cabeza y este le ofrece una respuesta vibratoria cuando detecta un "sonido reconocible" para que pueda reaccionar.	
Referencias	Este actor interviene en los casos de uso "Carga de batería".	

Tabla 4.- Descripción del actor "Persona con hipoacusia" (elaboración propia).

Atributos	
Nombre	**Descripción**
Edad	Tiene una edad mayor a 6 años y está en sus facultades de interactuar con equipos de cómputo.
Sensibilidad	El actor tiene la capacidad de reconocer de manera sensitiva estímulos vibratorios que suceden en su cabeza.

Tabla 5.- Atributos del actor "Persona con hipoacusia" (elaboración propia).

2.3.2 Universo de Casos de Uso

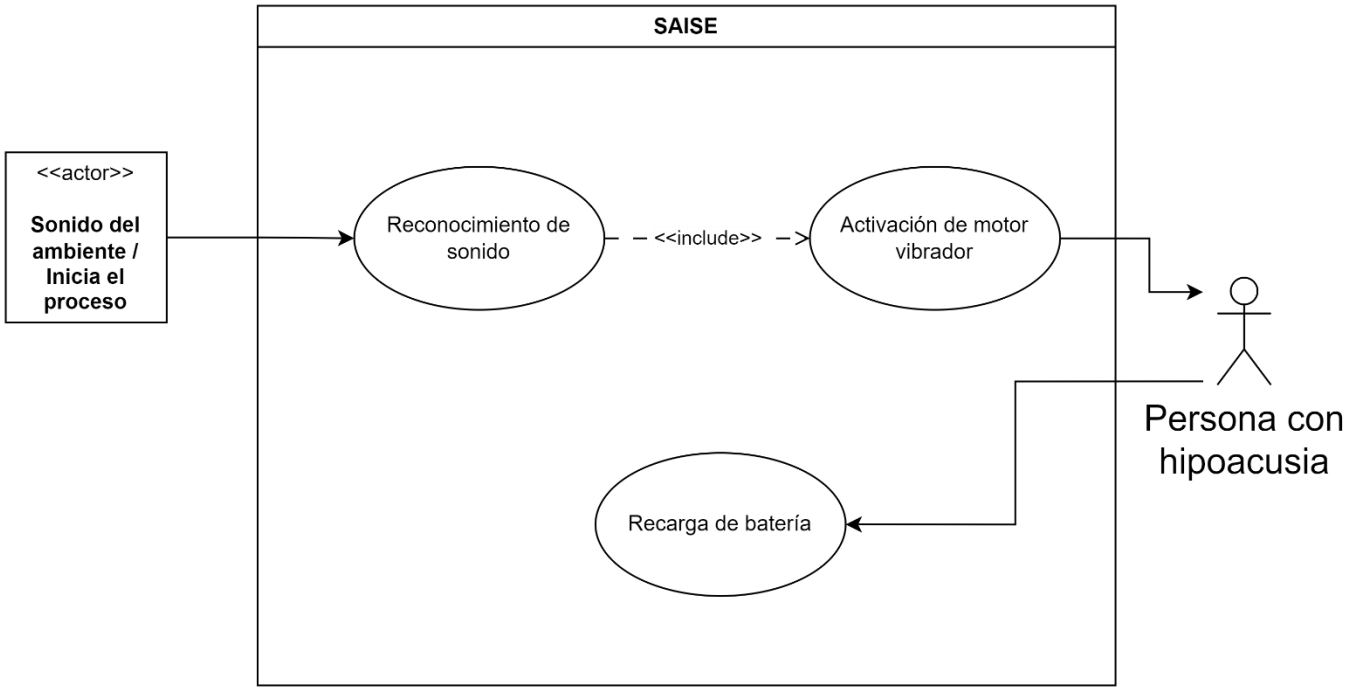


Ilustración 2.- Vista de todos los Casos de Uso considerados en el sistema SAISE (elaboración propia).

2.4 Modelado de Arquitectura

La arquitectura del sistema SAIE se presenta desde el modelo de vistas 4+1 de Kruchten. Esta organizado explícitamente para el entendimiento del sistema a partir de la vista Lógica, de Proceso, de Implementación y la vista Física. Esto para apoyar a la comunicación de la creación del proyecto para los diferentes interesados. Se utilizan diagramas de UML para describir estas vistas.

Estas vistas que surgen a partir de los requerimientos del sistema; su comportamiento en diferentes escenarios y el hardware donde se implementará la codificación, así como la relación con otros dispositivos. No se presentan soluciones de bajo nivel ni clases específicas de software, como tampoco etapas de diseño de hardware o descripción específica de protocolos de comunicación.

2.4.1 Plataformas

El SE del sistema se considera para un microcontrolador con arquitectura ARM M4. En principio se considera alguno de la marca *STMicroelectronics* que cumpla con estas características. Se utilizarán las herramientas de desarrollo de software (IDEs) ofrecidos por la misma marca, así como Python para las actividades de diseño y entrenamiento de la CNN.

2.4.2 Vista Lógica

Esta sección describe la parte conceptual del sistema, abordando sus componentes y relaciones generales.

En la ilustración 5 se muestra los componentes generales en su vista más alta y qué componentes tienen relación entre sí.

Cada componente se desglosa en detalle en el apartado de Vista de Desarrollo de apartado 2.5.5.

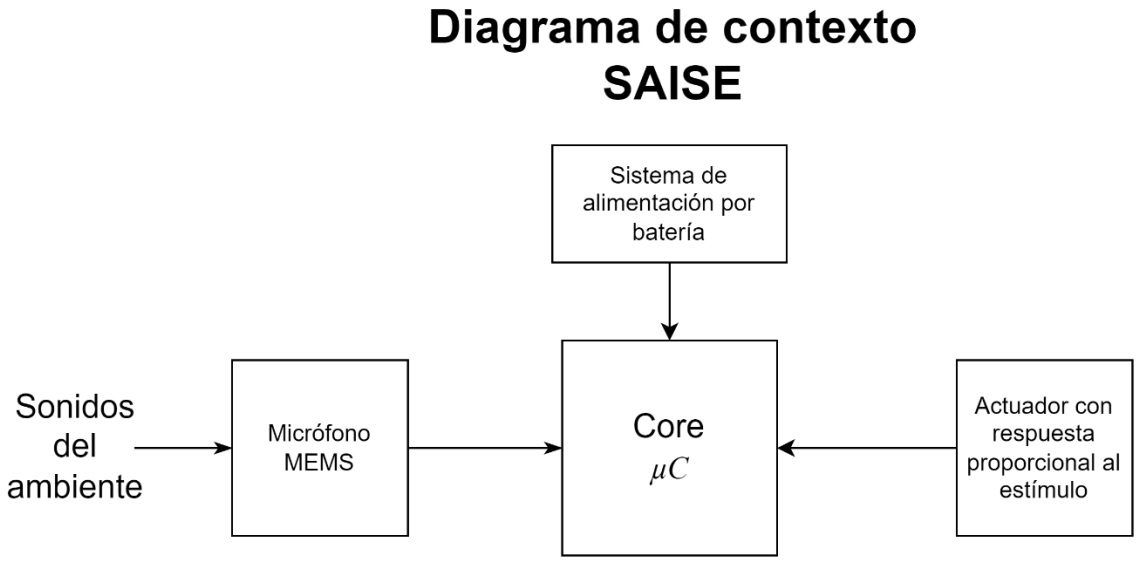


Ilustración 3.- Vista general del sistema (elaboración propia).

Los siguientes Diagramas de Estado, muestran la funcionalidad y desarrollo de los componentes previos.

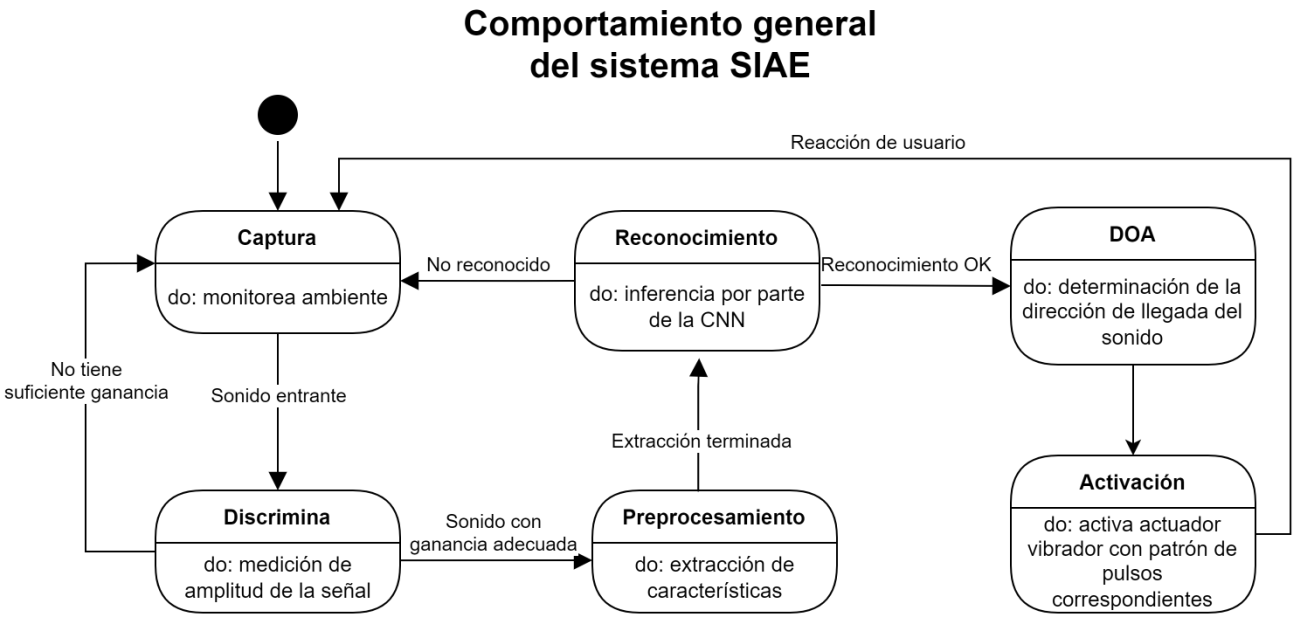


Ilustración 4.- Muestra las relaciones entre los componentes. El micrófono corresponde al estado de Captura; el microcontrolador a los estados Discrimina, Extracción y Reconocimiento, así como la de Activación que afecta al Actuador (elaboración propia).

2.4.3 Vista de Proceso

En esta vista se podrá observar cómo es el proceso del comportamiento del sistema y cómo se realiza la comunicación entre los diferentes componentes. Se cubren aspectos de desempeño y concurrencia.

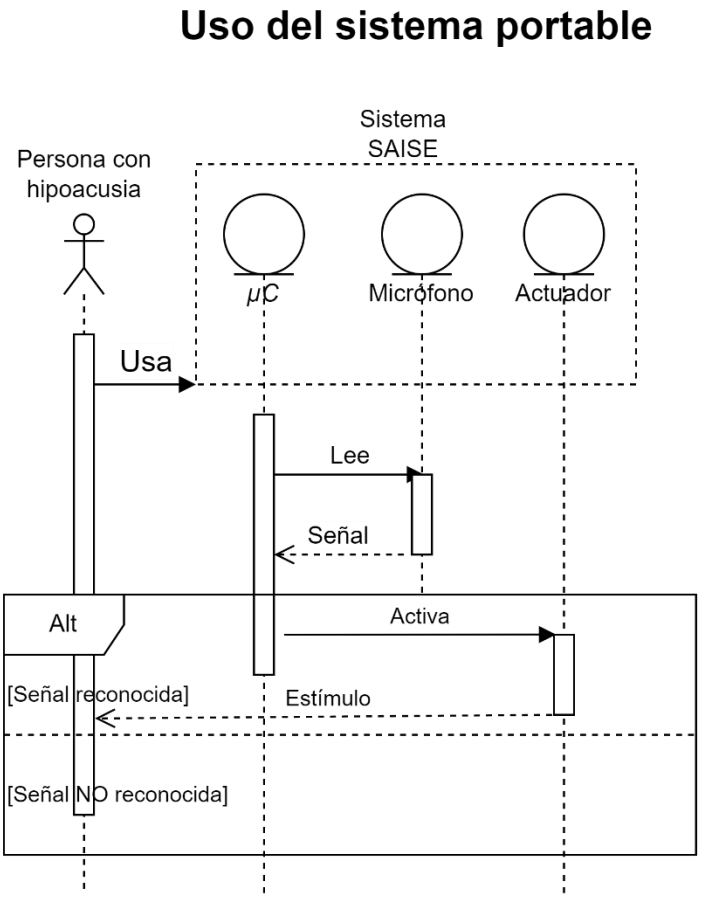


Ilustración 5.- Uso del sistema portable: diagrama de secuencia de la funcionalidad principal correspondiente al CU_reconocimiento. El tiempo de proceso de este hilo desde el inicio hasta el caso de que la señal sea reconocida no debe exceder los 2 segundos (elaboración propia).

2.4.4 Vista Física

Esta vista muestra conectividades entre los diferentes dispositivos en donde el software es desplegado y se ejecuta. Se utiliza un diagrama de despliegue para su representación.

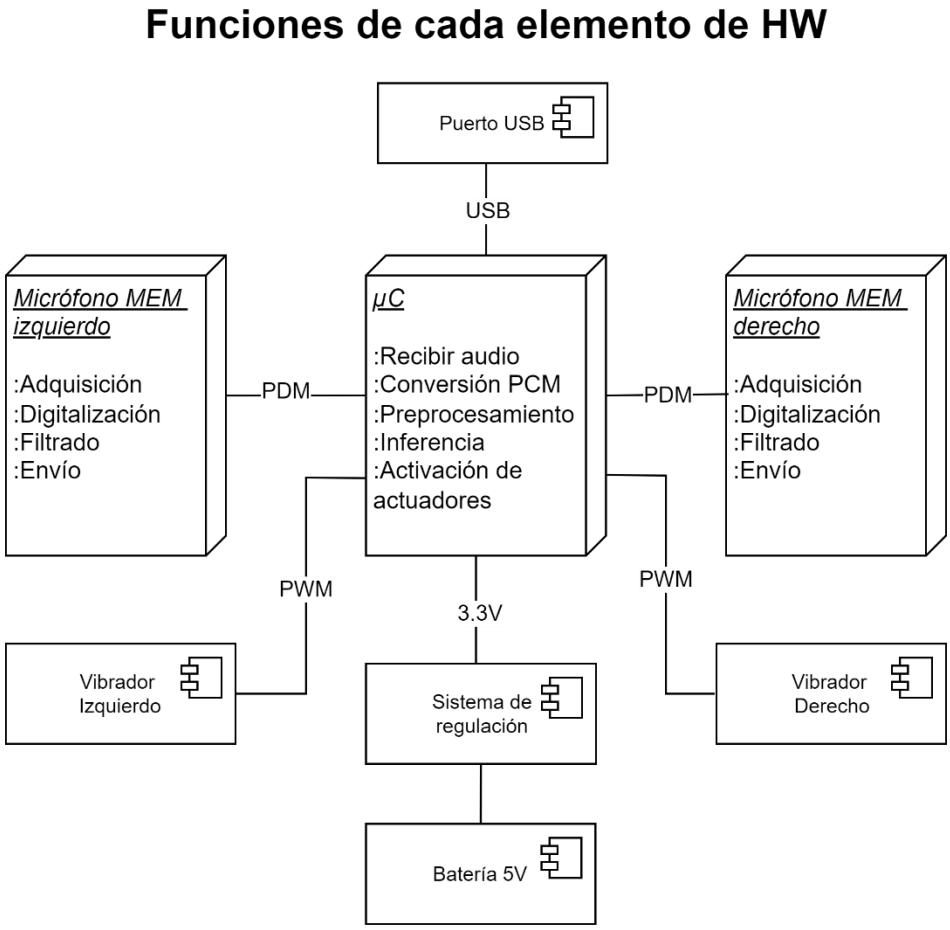


Ilustración 6.- Diagrama de despliegue de los diferentes elementos de hardware y como se comunican entre sí. También se muestran las funciones que se ejecutan en cada dispositivo (elaboración propia).

En esta vista se puede observar cómo interactúan los diversos dispositivos y las formas de comunicación o activación que tienen entre sí. Se duplican los micrófonos y vibradores ya que cada uno estará en un auricular para lo relacionado a la detección de la dirección de llegada.

2.4.5 Vista de Desarrollo

Esta sección describe la estructura general del SE y la descomposición de sus diferentes componentes que contiene, así como sus relaciones. Se utiliza una vista de componentes que, si bien está enfocada a componentes de software, puede funcionar como apoyo para la compresión del hardware involucrado.

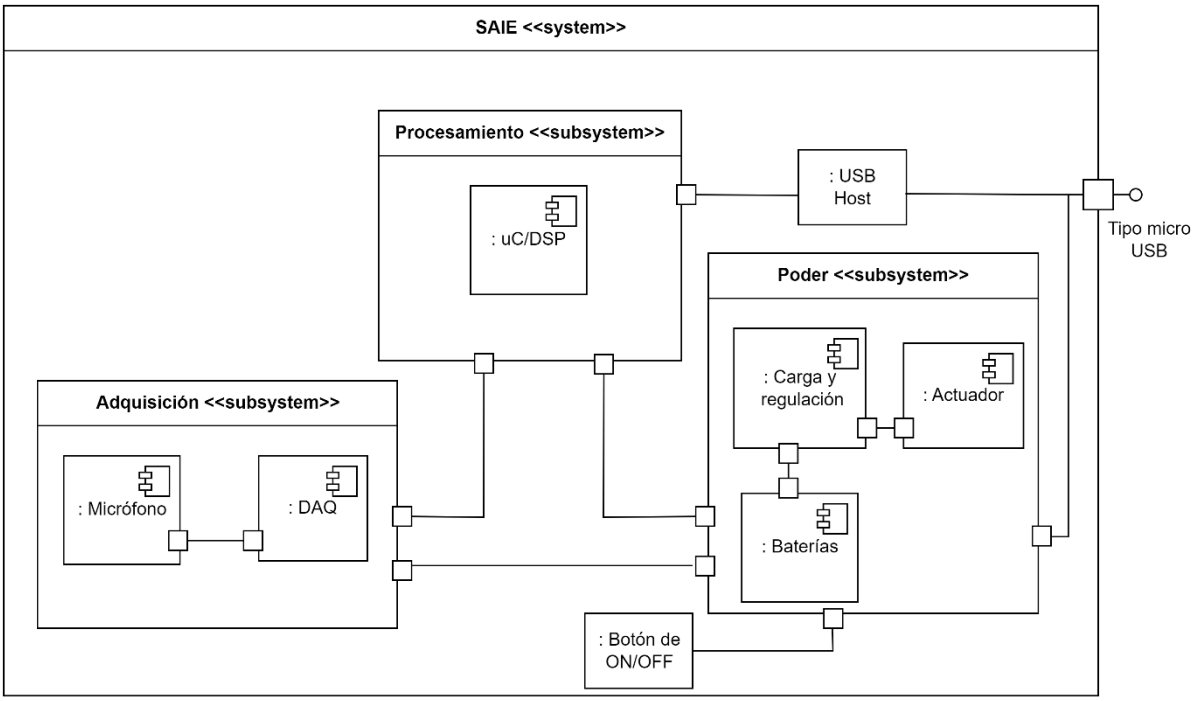


Ilustración 7.- Diagrama de componentes de hardware involucrados en el desarrollo del SE. Estos componentes forman parte del dispositivo portátil que el usuario final usará. Se pueden observar tres subsistemas principales y las relaciones entre ellos (elaboración propia).

El subsistema de adquisición es el encargado de la captura inicial de la señal y su primer tratamiento. Aquí tenemos dos componentes principales que es el micrófono; un dispositivo MEMS que puede o no realizar algún trabajo de filtrado sobre la señal. Posteriormente tenemos el componente de adquisición de datos (DAQ) que será quien acondicione y digitalice la señal para su procesamiento por el núcleo procesador.

El subsistema de procesamiento es donde se aplicarán los componentes de software para cubrir la tarea principal de reconocimiento de la señal. Contienen un componente principal de procesamiento digital de señales (DSP) que realizará operaciones pertinentes al tiempo discreto.

El subsistema de poder será el encargado de la gestión de energía del SE, proporcionando los voltajes necesarios para el funcionamiento de todos los dispositivos. Sus componentes permiten el proceso de recarga de baterías, así como la activación del actuador que dará la alerta háptica al usuario.

El resto del hardware es el puerto de recarga y comunicación por USB y un botón de apagado del sistema

2.4.6 Vista de Casos de Uso

Mismas que las especificadas en el apartado 2.4.2.

2.5 Plan de Pruebas

El plan de pruebas maestro se planea llevar a cabo al final de cada una de las tres iteraciones del proyecto. La finalidad es verificar los aspectos funcionales del sistema, así como su eficiencia. Se consideran pruebas de caja negra, caja blanca y regresión. También pruebas de rendimiento, estrés y usabilidad. El documento relacionado se puede enriquecer al final de cada iteración para considerar nuevas pruebas necesarias que se descubran a partir del análisis de los resultados. Las pruebas deben validar el cumplimiento de los requerimientos de usuario descritos en el documento IEEE Requerimientos SAIE disponible en la documentación del proyecto. Para algunas de las pruebas será necesaria la participación de personas con hipoacusia que son los usuarios finales del sistema.

2.5.1 Elementos de pruebas

1. Cualidades de captura del micrófono
2. Elementos del acondicionamiento de la señal (amplificación y filtrado)
3. Algoritmo de reconocimiento del sistema embebido portable
4. Patrones de activación del motor vibrador y su intensidad
5. Sistema de carga y batería

2.5.2 Funcionalidades a probar

Estas pruebas son resultado del análisis de escenarios de uso que se pueden encontrar en el documento Modelo de Casos de Uso SAIE V2.

A cada iteración del proyecto le corresponden un conjunto de pruebas a realizar.

Iteración 1, 2 y 3

1. Captura de los sonidos y su correcta clasificación y reconocimiento
    - Resultante de CU_Reconocimiento. Funcionalidad principal del sistema. Probar todos sus escenarios propuestos (flujo normal y alternos).
    - Prueba de caja negra y caja blanca en ambientes controlados. De una a cuatro fuentes sonoras diferentes.
    - Pruebas de estrés. Saturar el sistema con múltiples fuentes de sonidos con características diferentes y observar la capacidad de detección de los sonidos deseados.

2. Tiempo de respuesta del sistema desde la captura a la indicación de reconocido o no
    - Resultante de CU_Reconocimiento. Funcionalidad principal del sistema. Probar todos sus escenarios propuestos (flujo normal y alternos).
    - Prueba de rendimiento

3. Claridad para el usuario de los patrones de activación correspondientes a cada sonido
    - Resultante de CU_Reconocimiento que incluye Activación del motor vibrador. Funcionalidad principal del sistema. Probar todos sus escenarios propuestos (flujo normal y alternos).
    - Prueba de usabilidad. Aplicar rúbrica de conformidad del usuario.

Iteración 2

1. Tiempo de vida útil de la batería en correcto funcionamiento constante
    - Resultante de CU_Batería. Funcionalidad principal del sistema. Probar todos sus escenarios propuestos (flujo normal y alternos).
    - Prueba de caja blanca

2. Tiempo de carga de batería en estado desconectado
    - Resultante de CU_Batería. Funcionalidad no prioritaria. Probar todos sus escenarios propuestos (flujo normal y alternos).
    - Prueba de caja blanca

Iteración 3

1. Pruebas finales en usuarios
    - Resultante de CU_Reconocimiento. Validar el tiempo de uso necesario para que el usuario pueda entender el funcionamiento del dispositivo y valorar la correcta clasificación.
    - Prueba de caja negra

2.5.3 Pruebas de regresión

Después de cada iteración, se realizarán las siguientes pruebas de regresión en caso de ser necesario:

1. Verificación de captura correcta del sonido en caso de cambios en hardware como micrófono o acondicionamiento de señal.
2. Verificación del reconocimiento en caso de que no haya trabajado como esperado.
3. Verificación que los reconocimientos ya establecidos como correctos no hayan sido afectados por modificaciones al código debidas al punto anterior.

2.5.4 Enfoque de pruebas

Se realizarán pruebas de caja negra para evaluar las funcionalidades del sistema y que cumpla principalmente con los requerimientos de usuario descritos en el documento IEEE requerimientos SAIE.

Se realizarán pruebas de caja blanca para revisar aspectos tanto funcionales como no funcionales De los no funcionales, se abarcan principalmente el rendimiento y tiempo de respuesta del sistema, así como el almacenaje de información en la memoria del dispositivo, comunicaciones, consumo de energía y la respuesta en frecuencia de los micrófonos.

Se realizarán pruebas de usabilidad para determinar la facilidad de interacción del usuario con las interfaces tanto en PC como en el celular.

2.5.5 Criterios de aceptación, rechazo o suspensión

Criterios de aceptación o rechazo

Iteración 1, 2 y 3

1. **Prueba:** Captura de los sonidos y su correcta clasificación y reconocimiento
    - Al menos 75 de 100 ejecuciones con detección correcta.
    - Máximo 20 de 100 falsos positivos.
    - Máximo 5 de 100 falsos negativos.
2. **Prueba:** Tiempo de respuesta del sistema desde la captura a la indicación de reconocido o no
    - Máximo de 2 segundos

3. **Prueba:** Claridad para el usuario de los patrones de activación correspondientes a cada sonido
    - 9 de cada 10 patrones son interpretados correctamente por el usuario

Iteración 2

1. **Prueba:** Tiempo de vida útil de la batería en correcto funcionamiento constante
    - Al menos 6 horas de trabajo continuo

2. **Prueba:** Tiempo de carga de batería en estado desconectado
    - Hasta tres horas para lograr la carga completa.

Criterios de suspensión

De las siguientes pruebas mostradas, es indispensable su terminación aprobatoria para poner continuar con cualquiera de las demás del plan. En caso de no lograr los criterios de aprobación individuales al menos al 70% en iteración 1, se deberá suspender el avance hasta lograr un resultado mayor a esta referencia.

1. **Prueba:** Captura de los sonidos y su correcta clasificación y reconocimiento
2. **Prueba:** Tiempo de respuesta del sistema desde la captura a la indicación de reconocido o no

Criterios de reanudación

En caso de suspensión, las pruebas se reanudarán de acuerdo con lo planeado en cada iteración, cuando se sobrepase el criterio del 70% para las pruebas en iteración 1.

2.6 Diseño del modelo de aprendizaje profundo

Se le llaman modelos de aprendizaje profundo (*Deep Learning* por su definición en inglés) a la serie de algoritmos de aprendizaje supervisado bio-inspirados en las redes neuronales con el fin de la clasificación de nuevos datos en una clase específica. Así nace el concepto de redes neuronales artificiales (*DNN* por sus siglas en inglés).

Existen diferentes *frameworks* que facilitan la integración de este tipo de algoritmos para diferentes aplicaciones. Entre ellos podemos encontrar *PyTorch, Microsoft Cognitive Toolkit (CNTK), Caffe,* entre otros. Para esta investigación se plantea trabajar con *TensorFlow*, una biblioteca de *Machine Learning* desarrollada por Google, con una amplia documentación y utilidades, entre ellas la API *Keras,* que facilita el diseño y evaluación de los modelos DNN.

2.6.1 Descripción de los conjuntos de datos para el modelo de aprendizaje

Al ser de aprendizaje supervisado, es necesario tener datos de entrenamiento para el desarrollo y evaluación de la red neuronal. Esto aplica para cualquier tipo de clases a clasificar. En la actualidad, diferentes empresas e investigaciones han aportado sus conjuntos de datos, lo que ayuda enormemente al desarrollo ya que es una tarea de alto consumo de tiempo (recolección, adaptación y etiquetado), además que, entre más datos de entrenamiento, el resultado final del clasificador suele ser mejor. Para esta investigación se planea partir del conjunto *ESC-50 master (Piczak, 2015)*, un conjunto de audios creados para probar la clasificación de sonidos ambientales. El conjunto de audios consta de 2000 sonidos ambientales de 5 segundos promedio (formato *WAV*, 44.1 KHz, un solo canal) clasificados en 50 clases semánticas (40 audios por clase) que pertenecen a su vez a 5 super categorías (animales, naturaleza, humanos (no lenguaje), domésticos o de interior y exterior o urbanos). Como se planteó para este proyecto, el objetivo es la clasificación de sonidos catalogados como “emergencia” o “atención inmediata”, por lo que, de las 50 clases, se usan los audios correspondientes a “sirena”, “claxon”, “llanto de bebé”. La clase restante de “grito”

no se encuentra dentro de este conjunto, por lo que sus audios adquirirán a partir de diversas plataformas digitales y grabaciones propias.

Con esta selección, se tienen 160 audios (40 audios por 4 clases), que aún resultan insuficientes para poder tener un conjunto de entrenamiento adecuado, por lo que se realizará un procesamiento a estos audios inspirado en algunos tratamientos que se realizan a las imágenes usadas para el entrenamiento de modelos de DL (girar los objetos o desplazarlos en el espacio de captura), es decir, modificar las características de la señal en proporciones que pudieran resultar de sucesos reales. Esto se llevará a cabo en el entorno de programación de Python y usando la biblioteca *librosa* que está enfocada en el tratamiento de audio.

Se probarán modificaciones de altura (alteraciones de las frecuencias del audio) a más altas y más bajas; modificación de velocidad de los audios a más veloces y más lentos, así como una combinación de las anteriores. El objetivo es tener más audios con características que pudieran resultar de fuentes sonoras reales. La cantidad de modificaciones y valores se determinarán a partir de la apreciación humana de los audios donde se considerará si es factible que ese resultado pueda provenir de una fuente sonora real o circunstancia real.

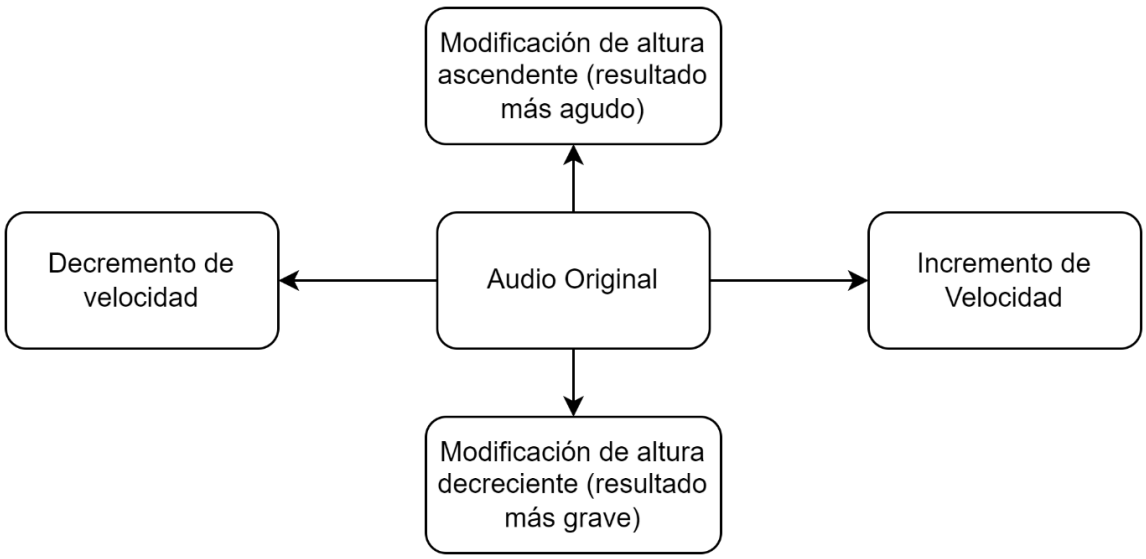


Ilustración 8.- Idea para el preprocesamiento de los audios para poder tener más información para el entrenamiento, validación y prueba (elaboración propia).

2.6.2 Tipos de redes neuronales consideradas para la implementación

Se planea realizar una comparación de dos modelos de redes neuronales implementadas en el embebido, Redes Neuronales Convolucionales (CNN por sus siglas en inglés) y en caso de que el tiempo sea suficiente, Redes Neuronales Recurrentes (RNN por sus siglas en inglés).

A continuación, se describen algunos parámetros relevantes que serán considerados al definir la arquitectura del modelo de aprendizaje profundo, así como su entrenamiento. La información es tomada de la documentación de *TensorFlow* referenciada como (*Abadi et al., 2015*) así como (*Nielsen, 2018*).

- Forma de los datos de la capa de entrada
    - Capa para usar como punto de entrada a la red neuronal, que es el vector de características. El tipo de capa es dependiente a la manera en que los datos están organizados (arreglos de 1 dimensión, 2 dimensiones o multidimensionales).
- Tipos de capas ocultas
    - Capa Perceptrón multicapa (*dense)*. - También llamada capa densamente conectada. Es una forma clásica donde cada neurona tiene una conexión con todas las neuronas de la siguiente capa. Estas uniones son llamados pesos que son valores entrenables.
    - Capa Convolucional. - Pueden ser de 1 Dimensión (1D), 2 Dimensiones (2D) o 3 Dimensiones (3D). Hace referencia a la realización de una operación de convolución sobre una señal. Para 1D habla de una convolución temporal, para 2D de una convolución espacial y para 3D de una convolución sobre volúmenes. Su principal parámetro entrenable es la ventana o *kernel* de *n* dimensión. Otro paramétro importante es si se aplica un relleno o "*padding"* para mantener las dimensiones de entrada iguales a la salida.
    - Capa de Normalización de Lote (*batch)*. - Es una capa que aplica una normalización de los datos antes de entrar a una capa de

entrenamiento. Esta capa normaliza los datos para que su media aritmética sea cercana a 0 y la desviación estándar cercana a 1.

- Capa de Agrupación por valor Máximo (*Max Pooling*). - Es acorde a la dimensión de los datos (1D, 2D o 3D) y reduce su resolución al tomar solo el valor máximo de una ventana de análisis (también acorde a la dimensión). Esta ventana recorre en su totalidad con un cierto número de avance (*stride)* los datos temporales que llegan a esta capa.
- Capa de Agrupación Global por Valor Promedio. - También acorde a la dimensión de los datos (1D, 2D, 3D) y de la misma forma reduce la dimensión de los datos en cada conjunto de características, pero mediante una operación de promedio de todo el conjunto de datos temporales que entran a esta capa.
- Capa de aplanado (*Flatten).* – Hace que cada elemento del conjunto de datos de *n* dimensiones quede en forma de un vector de 1D donde cada elemento es una neurona con posibilidad de ser parte de una capa perceptrón multicapa.

- Funciones de activación

  Las funciones de activación aportan no linealidad al resultado de la red, lo que le permite a esta desarrollar representaciones más complejas basada en los datos de entrada que no sería posible mediante, por ejemplo, un modelo simple de regresión lineal. Existen múltiples funciones de activación en las cuales podemos encontrar las siguientes:

  - Identidad. - *f(x)=x.*
  - Binaria. - *0 para todo x<0, 1 para todo x>=0*.
  - Sigmoide (o logística). - Como su nombre lo indica, basada en una función sigmoide con límites de salida entre 0 y 1.
  - Tangente hiperbólica. - Como su nombre lo indica, basada en una función del mismo nombre con límites de salida entre -1 y 1.
  - RELU. - Unidad lineal rectificada (*Rectified Linear Unit)* es una función cuyo resultado es *0 para toda x<0, x para toda x>=0*.

- Softmax. – Convierte un vector de valores en una distribución de probabilidad y se suele utilizar como función de activación de la capa de salida para que el resultado pueda ser interpretado como tal. La suma del vector de salida después de esta función es 1.

- Forma de la capa de salida
  - La forma de la capa de salida son *n* neuronas que dependerán de la cantidad de clases que se desean clasificar.
- Función de costo (o función de pérdida)

  Función para el cálculo del error entre lo predicho por el modelo y la clase real a la que pertenece una muestra del conjunto de entrenamiento. Este cálculo servirá para la corrección de los parámetros entrenables del modelo mediante la retro propagación durante el entrenamiento. Existen múltiples funciones para el cálculo del costo, como ejemplo podemos encontrar las siguientes.
  - Promedio del error cuadrático. - Basada en el cálculo del error como su nombre lo indica.
  - Promedio del error logarítmico cuadrático. - Basada en el cálculo del error como su nombre lo indica.
  - Entropía cruzada categórica. – También llamada pérdida logarítmica. Realiza el cálculo estadístico de la entropía cuyo valor de costo es más grande si la diferencia tiene a 1 (error grande de lo predicho contra la etiqueta real) y pequeña si tiende a 0. Esto debido a la operación logarítmica propia de la entropía. Esta función es usada cuando las etiquetas de las clases son un vector de probabilidades, donde la clase objetivo tiene el valor de 1.
- Hiperparámetros del entrenamiento

  Existen también múltiples propiedades que son configurables. Dentro de las más relevantes se tienen las siguientes.
  - Lote (*batch).* - Cantidad de *n* datos de entrada del conjunto de entrenamiento aleatoriamente seleccionados. Son los datos tomados

para realizar la actualización del gradiente (descrito en el siguiente punto).

- Época (*epoch).* – Número de iteraciones para entrenar el modelo. Es el número de veces que el optimizador realiza su función de actualización de los parámetros entrenables después de la retro propagación.
- Tasa de aprendizaje. – Es un parámetro que escala el resultado del gradiente calculado por el optimizador. Esta escalación determinará que tanta afectación tendrán los parámetros entrenables al realizar su actualización. El valor elegido también tendrá injerencia en el tiempo necesario para el entrenamiento, así como en algunos problemas comunes como lo son el sobre entrenamiento o sub-entrenamiento.

- Optimizador del entrenamiento

  Algoritmo que realiza el ajuste necesario de los parámetros entrenables durante el proceso de retro propagación, cuyo objetivo es la minimización de la función de costo en cada iteración. De igual manera existen múltiples algoritmos para realizar esta minimización. Algunos relevantes son los siguientes.

  - Gradiente Descendiente Estocástico (SGD por sus siglas en inglés). – Es el cálculo de las derivadas parciales que minimizan el error entre resultado de la función de funciones (donde las variables son los pesos, sesgos o valores del *kernel*) y la etiqueta esperada a ser predicha. Este proceso no es realizado para cada elemento del conjunto de datos de entrenamiento, en cambio, lo realiza sobre un subconjunto de datos de *n* elementos seleccionados aleatoriamente (de ahí su nombre estocástico). En su forma básica, el optimizador actualiza valor de los parámetros entrenables mediante $w=w- lr*g$, donde *w* es un parámetro como los pesos, sesgos o valores del kernel, *lr* es la tasa de aprendizaje y *g* es el resultado del gradiente.
  - Adam. – Es una variación del algoritmo SGD que está basado en una estimación adaptativa de los momentos del gradiente de primer y

segundo orden. En otras palabras, este optimizador adapta la tasa de aprendizaje de acuerdo con dichos momentos. Según se describe, es computacionalmente más eficiente, requiere menos memoria y presenta buenos resultados para problemas que son grandes en términos de parámetros entrenables y conjuntos de datos con una gran cantidad de elementos.

Se comenzará trabajando con las CNN al ser una técnica de referencia en el estado del arte, principalmente el realizado por *(Dhruv Jain et al., 2020)*, un trabajo ampliamente citado. Además de este, otros desarrollos han ahondado en el uso de CNN para la detección de audio como *(Jain et al., 2022)*, *(Abdullah Küçük et al., 2019)* o múltiples mencionados en el análisis de *(Dim et al., 2022)*. Aunque enfocadas principalmente a la clasificación de imágenes, los resultados del uso de CNN en los trabajos mencionados muestran que puede ser un camino prometedor. Además, con el uso de este tipo de redes neuronales artificiales, se busca poder tener comparaciones de resultados. Aunque el audio es una señal temporal, es posible su análisis como una imagen al realizar ciertas transformaciones a la información como el espectrograma, donde la intensidad del pixel representa la intensidad de una frecuencia determinada en un momento del tiempo, así, se obtiene una representación de la respuesta en frecuencia de varios fragmentos temporales de la señal. Aunque se podrían usar Redes Neuronales de Perceptrón básicas, estas no toman en cuenta las relaciones espaciales que mantienen los pixeles de la imagen, a diferencia de las CNN cuya arquitectura está pensada y adaptada para la clasificación de imágenes (*Nielsen, 2018*) gracias al uso de *kernels* en dos dimensiones, que son filtros espaciales que ayudan al resalte de las características de cada imagen.

Con este tratamiento del audio como imagen, se busca que el modelo pueda realizar el trabajo de extracción de las características y lograr inferencias certeras.

2.6.3 Tipo de preprocesamiento de los datos de entrada

Cabe mencionar, que hay muchas opciones de preprocesamiento (Sharma et al., 2020) para hacer la conversión de los datos temporales en imágenes o información de entrada para el modelo de Red Neuronal. La correcta selección de estas no es una tarea trivial, ya que influirá en gran medida al tipo de características que el modelo aprenderá, la cantidad de tiempo necesario para el entrenamiento e inferencia, así como para la cantidad de memoria necesaria para realizar los procesos. Además, cada opción presenta ventajas y desventajas de acuerdo con el problema a abordar, ya sea Procesamiento Natural del Lenguaje, Reconocimiento de Voz, Reconocimiento de Sonidos Ambientales, Procesamiento Musical, etc. Dentro de estas posibles opciones de preprocesamiento se encuentra la transformada rápida de Fourier (FFT), la conversión a coeficientes MEL, la envolvente de amplitud, los valores RMSE (*root- mean-square energy*), tasa de cruce por cero, tasa de energía de banda, flujo espectral, MFCC (*Mel-frecuency Cepstral Coefficients)* entre varias otras. Para este desarrollo plantean pruebas referentes a la combinación de valores frecuenciales con la FFT y la conversión a coeficientes MEL, preprocesamientos similares a los utilizados en (*Dhruv Jain et al., 2020*) con la intención de comparar resultados de desempeño.

Para la arquitectura de la CNN se partirá de dos propuestas hechas por (*TensorFlow, 2023*) y (*Valenti et al., 2016*). Con ellas se busca la obtención de los primeros resultados para su mejora o modificación total. El uso de CNN, además de ser referencia de buenos resultados en estos y otros trabajos como (*Jain et al., 2022*), permiten la clasificación a partir de características temporales-frecuenciales, lo cual las hace robustas comparados con otros métodos que solo aplican uno u otro tipo de característica.

De los preprocesamientos seleccionados, la FFT es un algoritmo que permite el cambio de datos discretos en el dominio de tiempo al dominio de la frecuencia, obteniendo un arreglo de números complejos de la mitad de las muestras usadas, donde la intensidad de las frecuencias será el valor absoluto del número complejo.

Para la implementación, el algoritmo necesita una cantidad de muestras que sea una potencia de 2 (ej. 128, 256, 512, 1024, etc.).

Una vez en el dominio de la frecuencia, con frecuencias en Hertz, se plantea la conversión a frecuencias en la escala MEL, que es una escala perceptual de alturas que son concebidas como equidistantes. El beneficio del cambio de escalas es la adaptación a aplicaciones que modelan la percepción auditiva y es adecuada para las aplicaciones de clasificación de audio (*Apple, 2023*). La transformación se logra multiplicando los valores del dominio de la frecuencia por un banco de filtros MEL.

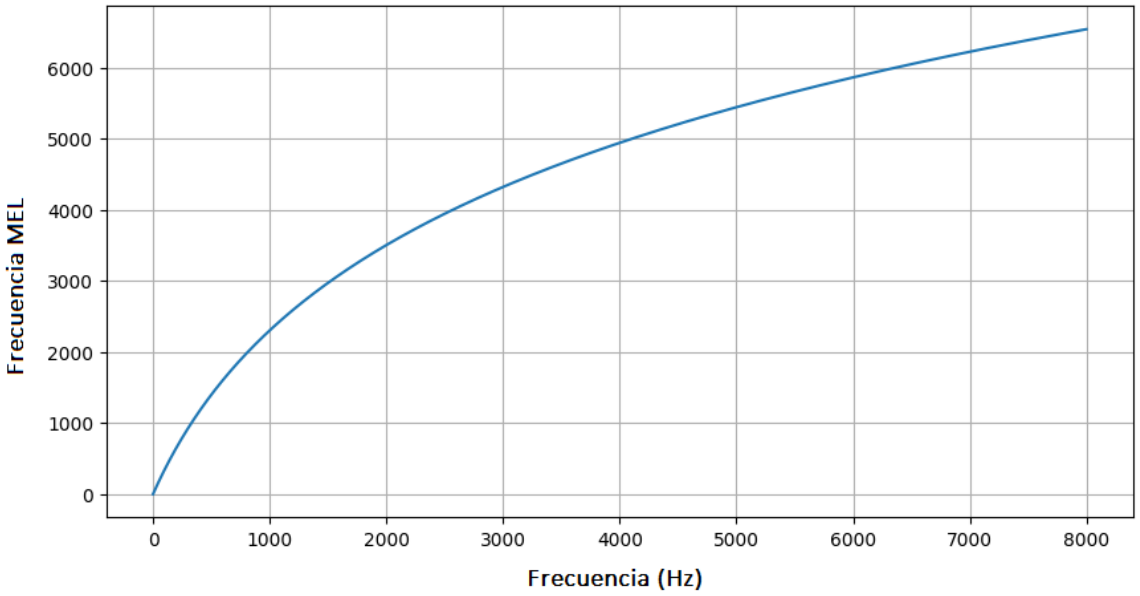


Ilustración 9.- Curva logarítmica de relación entre frecuencias Hertz -MEL (elaboración propia).

El banco de filtros MEL es una serie de filtros pasa bandas triangulares los cuales están empalmados de tal manera que la frontera inferior de un filtro está situada en la frecuencia central del previo y la frontera superior está situada en la frecuencia central del siguiente filtro. La máxima respuesta del filtro está localizada al centro del triángulo y está normalizada a la unidad. Hay una *M* cantidad de filtros dentro de un rango de frecuencia (frecuencia MEL más baja F_L *y* frecuencia MEL más alta del banco en F_H). Esta cantidad de triángulos *M*, será la que determine las frecuencias MEL finales obtenidas o *bins (Cheng et al., 2005)*. Un valor común de

bins representativos de un cierto espectro de frecuencias van de 40 a 80 bins, con lo que se logra una reducción de las variables.

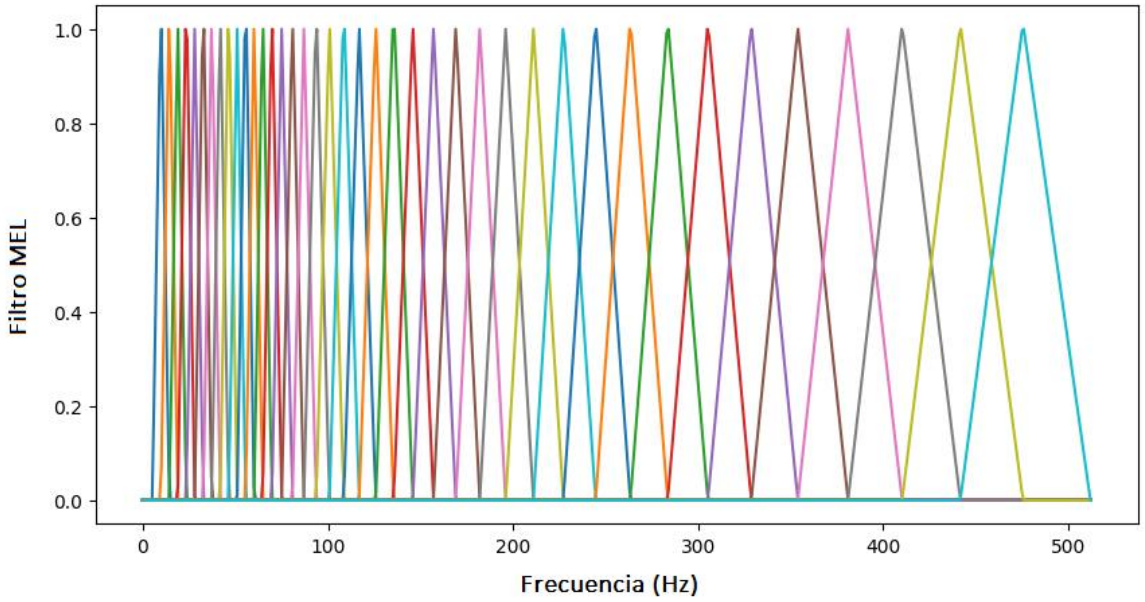


Ilustración 10.- Ejemplo de filtros del banco MEL (elaboración propia).

Una vez planteada la metodología, requerimientos y planeación del diseño, así como los fundamentos teóricos necesarios para el proyecto, se procede a describir los resultados.

Capítulo 3

Implementaciones y resultados

Capítulo 3. Implementaciones y resultados

El presente capítulo muestra los resultados obtenidos en la investigación a lo largo de sus tres iteraciones. Múltiples pruebas fueron realizadas y corregidas, por lo que se presentan los resultados más significativos. Se describe el diseño de la arquitectura del modelo de CNN y los resultados de múltiples entrenamientos realizados. Se abordan los pormenores de la captura de la señal de audio del entorno por el micrófono MEMS, tanto en un micrófono como en la captura estéreo, los diferentes algoritmos y etapas de preprocesamiento realizados, las características significativas de la señal y su posterior ingreso a la red neuronal encargada de la clasificación

El capítulo se divide en tres grandes apartados: en primer lugar, se describen los resultados del entrenamiento realizado fuera del dispositivo con los modelos descritos en la metodología. El apartado 3.2 describe la plataforma embebida comercial usada, su hardware, capacidades, y resultados de implementación. Finalmente, el apartado 3.3 aborda la integración del sistema en el diseño de tarjeta propio del proyecto, su comportamiento y resultados finales.

3.1 Diseño y entrenamiento del modelo de aprendizaje profundo

Se describirán las implementaciones descritas en la metodología respecto a las CNN, el preprocesamiento elegido, las diferentes variaciones hechas en la arquitectura, así como los resultados utilizando diferentes métodos de validación como por entrenamiento individual, así como validación cruzada de 10 etapas *(10-fold cross validation)*.

3.1.1 Proceso para acrecentar los datos de entrenamiento

Al ser elegido un modelo de aprendizaje supervisado, es necesario tener datos para el entrenamiento y evaluación de la red neuronal. El uso de bases de

datos existentes ayuda enormemente a la realización ya que es una tarea de alto consumo de tiempo (recolección, adaptación y etiquetado), además de poder comparar los resultados de clasificación con otros desarrollos que hayan usado dicha base de datos. Para esta investigación se usa la base de datos *ESC-50 master (Piczak, 2015)*, un conjunto de audios creados para probar la clasificación de sonidos ambientales. El conjunto de audios consta de 2000 sonidos ambientales de 5 segundos promedio (formato *WAV*, 44.1 KHz, un solo canal) clasificados en 50 clases semánticas (40 audios por clase) que pertenecen a su vez a 5 super categorías (animales, naturaleza, humanos (no lenguaje), domésticos o de interior y exterior o urbanos). El objetivo para este proyecto es la clasificación de sonidos catalogados como “emergencia” o “de atención inmediata”, por lo que, de las 50 clases, se usan los audios correspondientes a “sirena”, “claxon”, y “llanto de bebé”. La clase de “grito” no se encuentra dentro de este conjunto, por lo que estos audios se adquieren de la plataforma digital *freesound.org* y grabaciones propias. Una quinta clase es agregada para evitar los falsos positivos. Al ser únicamente 4 clases, el clasificador asignaba forzosamente a una clase cualquier sonido encontrado, aunque no fuera alguno de los cuatro deseados. Para mejorar esto, la quinta clase contiene otros sonidos ambientales generales (insectos, cuervos, perro, lluvia, viento, retrete, tormenta, estornudos, tosidos, aplausos, pasos, teclado, lavadora, aspiradora, motosierra, vidrio rompiéndose, motor, tren, campanas de iglesia y helicóptero), para que cualquier sonido diferente a los cuatro de emergencia, pudiera ser clasificado en esa clase a la que llamaremos “X”, que no provoca ninguna respuesta en el dispositivo.

Con esta selección, de las cuatro clases de emergencia se obtienen 160 audios (40 audios por clase) y 770 audios de la clase “X”. De estos audios originales, se realizan varios procesos. El primero consiste en modificar la frecuencia de muestreo f_s de 44.1 KHz a 16 KHz, que será la frecuencia de muestreo con la que trabajará el sistema embebido. Se escoge esta frecuencia (usada usualmente en audio comercial para voz y comunicaciones) para tener una compensación entre agudos del espectro audible (máximo 8KHz) y una demanda media para el microcontrolador.

Los sonidos de las clases de emergencia aún resultan insuficientes para poder tener un conjunto de entrenamiento adecuado. El segundo proceso consiste en un recorte manual de los audios. No se necesitan los 5 segundos de duración para la clasificación ya que se trabaja con una captura de 800mS. Por lo tanto, se recortan los audios en el software *Audacity* para obtener nuevos audios útiles con una duración de 1s a 1.5s. Esta tarea se repite hasta lograr 70 audios por clase.

Los setenta audios por clase de emergencia aún son desequilibrados comparado con los 770 audios de la clase “otros”. El tercer proceso que se realiza a estos audios está inspirado en los tratamientos aplicados a imágenes para enriquecer bases de datos (rotar los objetos o desplazarlos en el espacio de captura), es decir, modificar las características de la señal en proporciones que pudieran resultar de sucesos reales. Como equivalente de estas modificaciones en un audio, se eligen la manipulación de la altura y la velocidad. Esto se lleva a cabo en el entorno de programación Python usando la biblioteca *librosa,* enfocada en el tratamiento de audio.

Se probaron diversas modificaciones tanto en altura (alteraciones de las frecuencias del audio a más agudas o graves) como en velocidad (audios más veloces o lentos), así como una combinación de las anteriores. El objetivo es tener más audios con características que pudieran resultar de fuentes sonoras reales. La cantidad de modificaciones y proporciones se determinaron a partir de la apreciación humana, donde se considera si es factible que ese resultado pueda provenir de una fuente sonora o circunstancia real y que sea lo suficientemente diferente para ser considerado un nuevo archivo. Así, para la altura se escogen 1.5 y 3 semitonos tanto en incremento como en decremento. Para la velocidad, una tasa de 50% más rápido y 30% más lento. Además, de estos audios ya modificados, se realiza una combinación; los audios de ±3 semitonos se modifican también en velocidad (x1.5 y x0.7). En total, cada audio con Fs=16 KHz tiene 11 versiones (1 original, 4 transformaciones en altura, 2 en velocidad y 4 en altura-velocidad), dando como resultado 770 audios por cada clase de emergencia, igualando así la cantidad de audios de la clase “otros”. Un resumen de estas transformaciones se puede observar en el siguiente diagrama.

En total se tiene una base de datos de 3850 sonidos, los cuales fueron posteriormente preprocesados para extraer características relevantes para el modelo CNN.

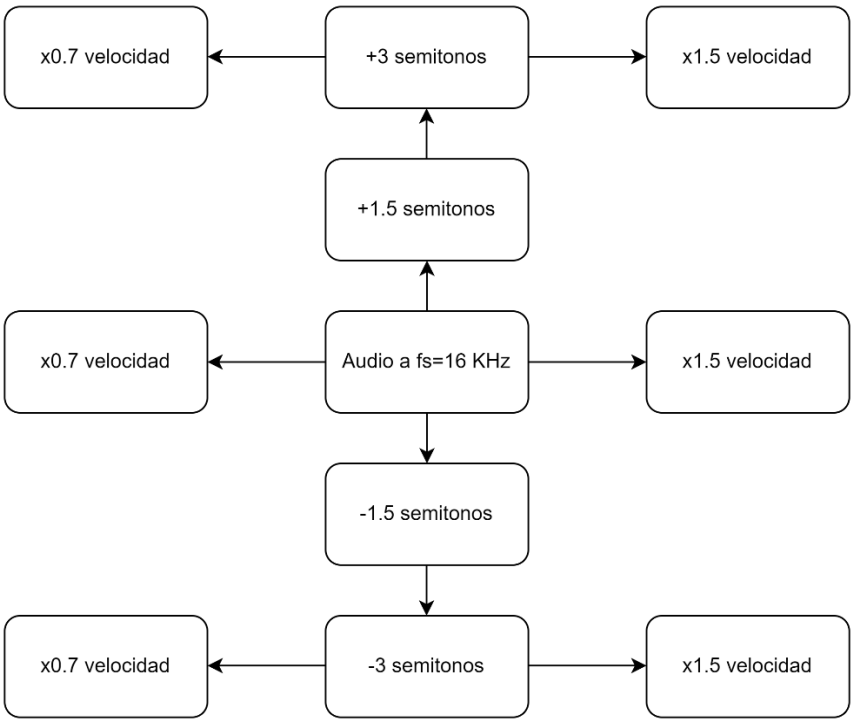


Ilustración 11.- Modificaciones realizadas a los audios del conjunto ESC-50 (elaboración propia).

3.1.2 Preprocesamiento de los datos para su entrada de la CNN

Como se mencionó, se utiliza un modelo de aprendizaje supervisado para reconocer las cuatro clases de emergencia, específicamente un algoritmo de Redes Neuronales Artificiales. Este desarrollo se realizó de igual forma en *Python,* haciendo uso del Framework de *TensorFlow y* su API Keras. Dicho proceso es descrito a continuación.

Los audios obtenidos son colocados en sus carpetas correspondientes, cuyo nombre corresponde a la clase (“Bebé”, “Claxon”, “Grito”, “Sirena” y “X”). Cada audio es cargado a Python en formato PCM 16 bits. Se seleccionan aproximadamente los primeros 0.8 segundos de audio y las muestras se organizan en una matriz de n

filas, con 1024 elementos cada una. Con una frecuencia de muestreo de 16 KHz, en 0.8 segundos de audio se obtiene un tensor de 12x1024. Si el audio dura más de 0.8 segundos, el restante se descarta, si es menor, las muestras faltantes se rellenan con 0 (caso poco usual). Cada audio se adjunta a dos listas, una lista de datos para entrenamiento y otra para validación-prueba. La proporción es de 80%-10%-10% respectivamente, dando como resultado 3080 tensores para entrenamiento, 385 tensores para validación y 385 tensores para el conjunto de prueba.

El modelo utilizado está basado en una red neuronal convolucional (CNN), modelo que ha demostrado su utilidad para el procesamiento de imágenes. El objetivo es crear una “imagen” representativa del audio para posteriormente ser la entrada a esta red. En este estudio se probaron dos preprocesamientos que son mostrados en la siguiente imagen.

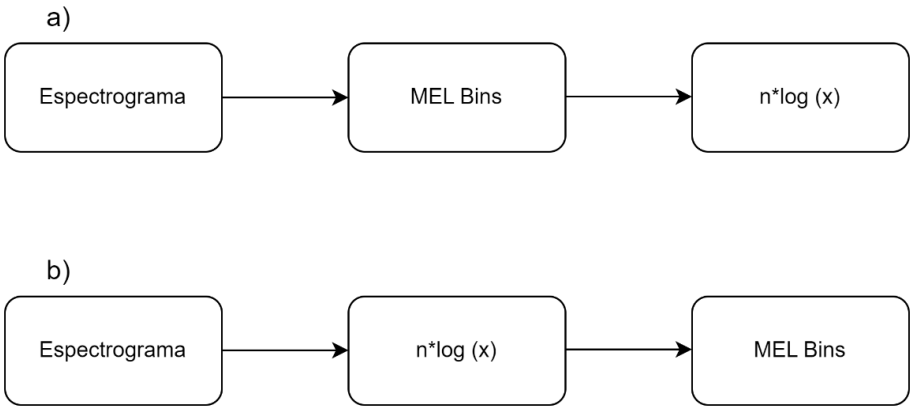


Ilustración 12.- Dos preprocesamientos para la extracción de características (elaboración propia).

La diferencia entre los dos preprocesamientos radica sólo en el cambio de orden de la operación.

En primer lugar, se obtiene la transformada rápida de Fourier para valores reales (RFFT por sus siglas en inglés) de cada vector que se define de 1024 muestras, lo cual nos da 512 valores complejos de los cuales se obtiene su magnitud. Esta transformación a frecuencia se probó con una operación de

ventaneo rectangular y con una ventana Hann, siendo esta última la transformación con mejores resultados en el entrenamiento. Se procede a la obtención de los coeficientes MEL, que como se mencionó, es una transformación de la magnitud de las diferentes frecuencias a una escala equiparable a la manera como los humanos percibimos el sonido, la cual no es lineal conforme la frecuencia aumenta. Esta transformación ha sido usada en diferentes estudios para la extracción de características de señales audibles además que permite una reducción de los datos, caso necesario para la implementación en el embebido ya que cada valor del tensor es de 32 bits, y existe un espacio muy limitado en el microcontrolador. Para lograr esta transformación, se crea un banco de filtros MEL y posteriormente se realiza el producto matricial con las magnitudes de las diferentes frecuencias. Para el diseño del banco MEL se hicieron pruebas con diferentes parámetros: una constante fue pasar de 512 valores frecuenciales lineales, a 80 bins en frecuencias MEL; pruebas con un límite inferior de frecuencia de 80 Hz a 160 Hz y un límite superior de frecuencia de 7.6 KHz a 7.84 KHz. Finalmente, existe una transformación a dB con la operación logarítmica.

Algunos valores estadísticos característicos de todo el conjunto de entrenamiento son:

Para el preprocesamiento de la ilustración 12 a) se obtiene:

Valor máximo:	149.0704
Valor mínimo:	-4.3675
Media:	89.3457
б:	22.8460

Para el preprocesamiento de la ilustración 12 b) se obtiene:

Valor máximo:	1575.0800
Valor mínimo:	-136.4173
Media:	415.8248
б:	263.4314

Después de varios entrenamientos usando una arquitectura de CNN inicial (*Valenti et al., 2016*), el preprocesamiento Espectrograma->Frecuencias MEL->dB

fue el que tuvo un mejor desempeño en el clasificador, logrando valores de exactitud con el conjunto de prueba mayores al 90%, mientras que la otra opción de preprocesamiento no superó este valor. Es por esto, que la primera decisión en las pruebas siguientes fue continuar trabajando con el orden de preprocesamiento a).

Una vez preprocesados, se obtienen tensores con dimensión de 12x80 valores. Esta será la forma de entrada de la red neuronal. Antes de describir las características de la arquitectura de la CNN, se mencionarán algunas otras pruebas relevantes de esta etapa de preprocesamiento, las cuales ayudaron para finalmente determinar la forma de entrada y tipos de valores finales. De estas pruebas se describe a continuación las relacionadas a tiempo de captura de audio, el uso de cuadros empalmados y diferentes rangos para el diseño del banco MEL.

El tiempo de captura es un parámetro relevante ya que está relacionado directamente con el tiempo de reacción del usuario hipoacúsico y a su vez la cantidad de información que la red neuronal tiene para poder realizar una inferencia fiable. Un tiempo de reacción largo no es funcional para un dispositivo enfocado a sonidos de atención inmediata aun si la clasificación es más exacta por parte del sistema embebido (más características temporales-frecuenciales). Un equilibrio entre ambos es necesario.

Se comenzó con un tiempo inicial de 3 segundos de captura que disminuyó hasta 0.5 segundos (de 48 a 8 cuadros de captura de 1024 muestras). A través del análisis de las curvas de entrenamiento y validación, así como la matriz de confusión resultante de la evaluación con el conjunto de prueba, se observó que por debajo de los 0.7 segundos de captura la exactitud disminuía respecto a tiempos mayores, y por debajo de 0.5 segundos, tenía una disminución más acentuada. Superior a 0.7 segundos, la exactitud del conjunto de prueba superaba el 91% hasta menor de 94%, sin notar grandes cambios conforme subía el tiempo (resultados fueron obtenidos tomando como punto de partida un diseño inicial de CNN propuesto por (*Valenti et al., 2016*). Esto llevó a la decisión de elegir un tiempo de captura de 0.8 segundos. Así, los datos tenían suficientes características temporales-frecuenciales para el modelo CNN usado y ofreciendo un tiempo de respuesta satisfactorio para el usuario.

También se realizó la aplicación de un filtro *pre-emphasis* en el domino del tiempo de características $P(z)=1-0.68z^{-1}$, sin que esto aportara grandes mejoras.

Otra prueba relevante fue el uso de cuadros empalmados por mitad de los 1024 que corresponde un cuadro (en la muestra 512) durante la captura. Con esto se buscaba suavizar las transiciones entre los instantes de tiempo, al aumentar la "resolución" de la "imagen" resultante. Es decir, en el proceso anterior quedaban 12 cuadros consecutivos para la captura de 0.8s. Para este caso, al reorganizar las capturas provocando empalmes intermedios de la misma longitud, se obtiene un dato de entrada de 23 cuadros. En la siguiente figura se muestra esta organización.

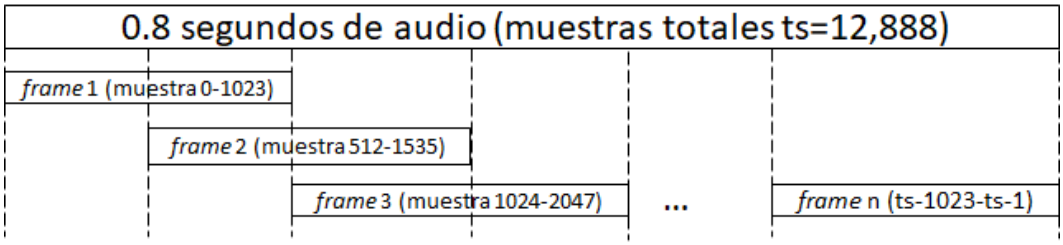


Ilustración 13.- Forma de captura con frames empalmados (elaboración propia).

El resto del preprocesamiento es el mismo que el indicado previamente (Espectrograma->Frecuencias MEL->dB). Así, la matriz de entrada para la red usando cuadros empalmados es de 23x80 datos. La exactitud de la clasificación con el conjunto de prueba ronda alrededor del 93.8%. El resultado no aporta gran ventaja al previo con cuadros consecutivos y por el contrario requiere más memoria para los datos de entrada y un algoritmo más complejo para la organización de los empalmes.

Por último, otras pruebas realizadas fueron las relacionadas a los parámetros del banco MEL. Como se mencionó, se necesita determinar la mínima y máxima frecuencia a convertir, así como el número de frecuencias MEL que harán la equivalencia a las 512 magnitudes de la FFT. La definición de 80 frecuencias MEL para el banco no cambió, lo que se modificó fue la frecuencia mínima y máxima que servirán de límites para la transformación. Así, la elección de estos valores funciona

como un filtro pasa bandas, discriminando frecuencias por debajo de la frecuencia mínima del banco y frecuencias por arriba de la frecuencia máxima. Diferentes parámetros fueron probados para determinar el mejor rango, incluso la puesta a 0 manual de las frecuencias por debajo de cierto valor. Es importante mencionar que las frecuencias útiles de los sonidos a clasificar (claxon, grito, sirena y llanto de bebé) tienen frecuencias útiles a partir de los 350 Hz aproximadamente. En la ilustración 14, se muestran algunos ejemplos de cómo cambia la matriz de entrada con la modificación de este rango.

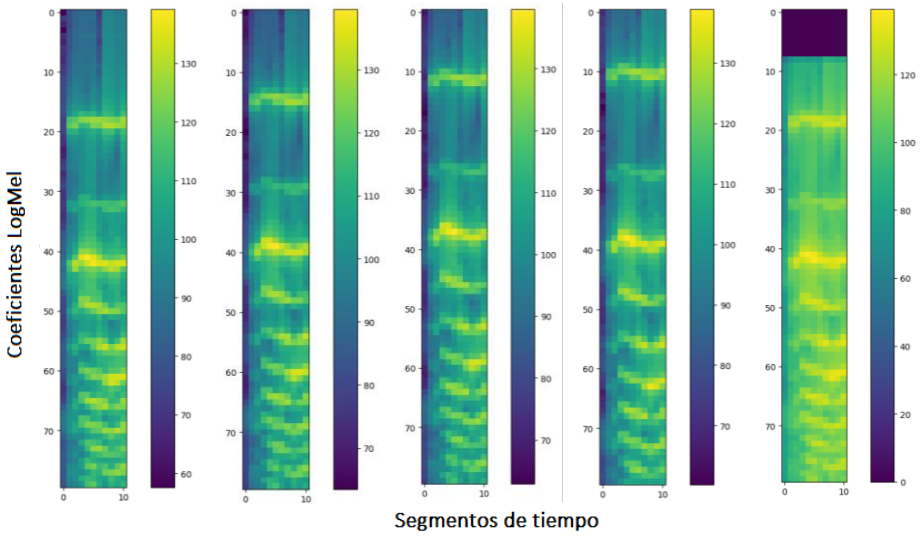


Ilustración 14.- De izquierda a derecha: a) Fmin=80Hz, Fmax=7800Hz, b) Fmin=200Hz, Fmax=7800Hz, c) Fmin=300Hz, Fmax=7800Hz, d) Fmin=300Hz, Fmax=6600Hz e) Fmin=80Hz, Fmax= 7800Hz con una asignación *manual de 0 intensidad por debajo de los 328.125 Hz (elaboración propia).*

Se puede observar cómo se desplazan las curvas características de las frecuencias dependiendo de los rangos del banco MEL. El pasa altas creado al enviar por completo a 0 la intensidad de las frecuencias por debajo de cierto valor no aportó gran mejora. Después de varias iteraciones de pruebas y basados en los resultados del entrenamiento, el rango de diseño del banco MEL quedó con una frecuencia mínima de 160 Hz y una frecuencia máxima de 7840 Hz.

Haciendo un resumen de la etapa de preprocesamiento de los datos, la forma de entrada para la red neuronal es la siguiente. Con una Fs =16 KHz, se capturan

cuadros consecutivos de 1024 muestras durante aproximadamente 0.8 s, dando 12 cuadros en total. A cada cuadro se le aplica la FFT aplicando una ventana de *Hanning* y el cálculo de su magnitud para obtener 512 valores frecuenciales, posteriormente se hace la transformación a frecuencias MEL con un diseño de banco de 80 bins, frecuencia mínima de 160 Hz y frecuencia máxima de 7840 Hz. Una vez hecha la transformación, se aplica una operación *10*log(F MEL)* para reducir la dispersión de los datos (valores en dB). Así, la forma de entrada de la red es una matriz de 12x80 valores.

3.1.3 Diseño y pruebas para la obtención de la arquitectura de CNN

Se procederá a hablar del diseño de la CNN, las pruebas hechas y sus resultados, para finalmente mostrar la arquitectura de red utilizada. Como ya se mencionó, para el entrenamiento se utilizó el *framework* de *TensorFlow* con su API *Keras* y se tomó como base la CNN utilizada en (*Valenti et al., 2016*). Una vez entrenado el modelo y si los resultados de exactitud y matriz de confusión eran satisfactorios, se exporta y se carga en el dispositivo embebido. De la arquitectura de la CNN, es importante considerar la cantidad de parámetros resultantes, ya que esto tiene relevancia en el uso de memoria del microcontrolador. La modificación de arquitectura propuesta se puede observar en la ilustración 15.

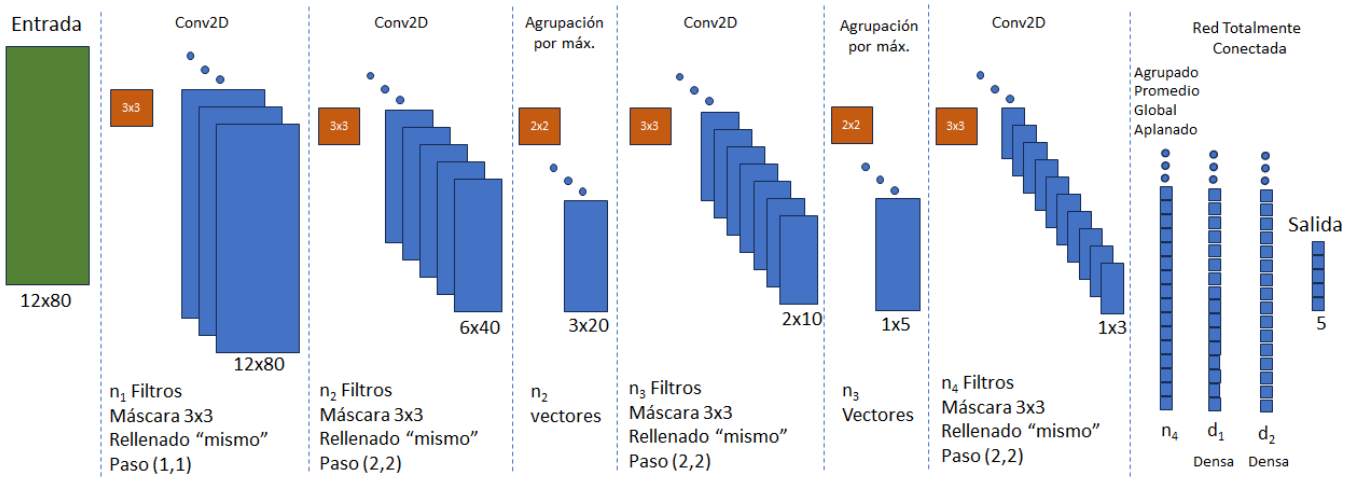


Ilustración 15.- Organización de las capas de la arquitectura de CNN usada (elaboración propia).

Se pueden observar 11 capas desde la entrada a la salida. Con esta base de arquitectura, se hicieron múltiples variaciones para encontrar la red final a utilizar, es decir, consistió en un proceso iterativo de *entrenamiento-prueba en el embebido-reajuste de los parámetros de la red*. Los parámetros modificados durante las pruebas del entrenamiento fueron *learning rate, epochs, batch size, optimizer;* número de filtros en las capas convolucionales n_1, n_2, n_3, n_4; número de neuronas en las capas densas d_1, d_2; tamaño del *stride* de las capas convolucionales y diferentes funciones de activación. En algunas pruebas esta arquitectura base fue modificada al agregar una capa convolucional más u otra capa densa, sin embargo, no se ahonda en ello ya que resultaba en una red de mayores parámetros (no funcional para el microcontrolador elegido) o el resultado no aportaba ninguna mejora significativa. Con el fin de simplificar la red y el consumo de memoria, se probó una arquitectura más simple con menos capas, la cual dio resultados más bajos de la red mostrada en la figura anterior. Los siguientes resultados son considerando el preprocesamiento a) descrito en la figura 13 y un tiempo de captura de 0.8 s. Se realizaron ejecuciones del entrenamiento individuales en laptops comerciales (Intel Core I5, 12 GB de RAM, no GPU) así como procesos iterativos en un servidor para encontrar las mejores configuraciones. Se entrenó y evaluó la red organizando la información en los conjuntos de *entrenamiento-validación-prueba (80%-10%-10%)* para iteraciones individuales en la PC y una segunda etapa de reforzamiento de resultados mediante iteraciones de validación cruzada de 10 etapas (*10-fold cross validation*). Estas últimas pruebas fueron posibles gracias a que se contaba con el servido disponible.

La mayoría de los resultados de las diferentes pruebas entrenando con la arquitectura base fueron superiores al 80% de exactitud con el conjunto de prueba. Pocos de los resultados estuvieron debajo de este valor. El primer resultado con un modelo inicial $n_1=3$, $n_2=16$, $n_3=32$, $n_4=48$, $d_1=8$, $d_2=4$, un optimizador "*Adam",* una lr=0.0001, epochs=60, batch=44, función de pérdida "*categorical crossentropy*" y funciones de activación *"relu",* tuvo una exactitud del 91%. El peor resultado individual obtenido en el proceso iterativo fue del 69.6% de exactitud. Los resultados usuales rondaron entre el 88% y el 94%, para finalmente encontrar los mejores

resultados alrededor del 97.14% para una iteración individual y 95.75% de promedio con *10-fold cross validation*. Cabe mencionar que un parámetro determinante de cambio fue el del optimizador, que pasó de "*Adam*" a "*SGD*", para lograr una mejor generalización de las 5 clases, lo que significó un incremento en las *epochs* del entrenamiento hasta valores entre 400 y 700. Algunos de los resultados obtenidos de las múltiples iteraciones son mostradas en el siguiente apartado.

3.1.4 Resultados del entrenamiento

En la siguiente tabla se encuentran algunos de los resultados de las iteraciones de *10-fold cross validation* con los filtros convolucionales de n_1=3, n_2=16, n_3=32 y n_4=48.

Forma d1-d2	Exactitud (%)	σ (%)	Clase más baja	Clase más alta	Con Ventana Hann	Con Pre-emphasis
2-40	78.00%	4.06	Be=0.63	Cl=0.85	No	No
8-0	92.50%	0.86	Be=0.91	Cl=0.9344	No	No
40-16	92.60%	1.15	X=0.88	Gr=0.94	No	No
44-64	89.90%	3.65	Be=0.86	Cl=0.9324	No	No
48-16	93.50%	0.34	X=0.89	Si=0.9493	No	No
48-96	90.00%	3.16	X=0.86	Cl=0.93	No	No
52-28	92.10%	1.09	Be=0.88	Si=0.946	No	No
52-72	88.50%	3.17	Gr=0.8707	Cl=0.901	No	No
60-8	92.09%	0.57	X=0.88	Si=0.936	No	No
64-8	92.46%	0.65	X=0.90	Cl=0.95	No	No
78-8	91.00%	0.95	X=0.89	Si=0.9259	No	No
80-8	90.00%	2.3	Be=0.847	Cl=0.94	No	No
80-32	91.00%	3.06	X=Gr=0.89	Cl=0.935	No	No
80-72	92.00%	2.07	Be=0.8922	Cl=0.9376	No	No
88-8	93.72%	0.35	X=0.9201	Cl=0.9610	No	No
88-32	93.23%	0.19	Si=0.9233	Cl=0.9461	No	No
88-88	94.10%	0.18	X=0.9194	Cl=0.9642	No	No
88-88-b	82.67%	0.95	Be=0.711	X=0.8967	No	No
96-16	93.27%	0.15	Be=0.910	Cl=0.962	No	No
96-20	90.80%	1.39	X=0.878	Si=0.9246	No	No
96-56	90.53%	1.93	Be=0.884	Cl=0.933	No	No
48-0	94.49%	0.12	X=0.906	Si=0.9681	Si	No

46-16	95.36%	0.15	Be=0.9324	Gr=0.9727	Si	No
48_16b	93.98%	0.7	X=0.9194	Cl=0.9701	Si	No
88-0	94.31%	0.4	Be=0.9227	Si=Cl=0.957	Si	No
88-0b	95.36%	0.1	X=0.9311	Cl=0.9642	Si	No
88-8	94.14%	0.64	X=0.9240	Cl=0.9577	Si	No
88-8b	92.42%	1.59	Be=0.9064	Gr=0.9454	Si	No
88-88	95.03%	0.58	X=0.9233	Cl=0.9668	Si	No
88-88b	95.75%	0.1	X=0.9350	Cl=0.9772	Si	No
48-16	95.44%	0.1	Be=0.9396	Gr=0.9720	Si	Si
88-8	94.98%	0.07	X=0.9155	Gr=0.9707	Si	Si
88-88	94.66%	0.2	X=0.9188	Si=0.9668	Si	Si

*Tabla 6.- Resultados de 10-fold cross validatio*n. *Mejor resultado marcado (elaboración propia).*

En la tabla podemos observar si en el preprocesamiento se tuvo considerado o no aplicar a los audios de entrenamiento un ventaneo *Hann* para la STFT y si hubo o no filtro de preénfasis aplicado. Vemos también los promedios de las 10 etapas de validación de exactitud, desviación estándar (σ), la clase más alta y la clase más baja. Basado en los resultados de exactitud y σ, se eligió como modelo final para cerrar este proyecto la que tienen las capas densas finales d_1=88, d_2=88, versión b, con aplicación de ventana y sin filtro preénfasis, que tuvo un promedio de exactitud de 95.75%, una σ=0.1. Cabe mencionar que no fue el único modelo probado en el SE, ya que se probaron otras a lo largo de las iteraciones y en diferentes condiciones, y se iban buscando mejoras en el preprocesamiento y arquitecturas.

La ilustración 16 muestra el desempeño del entrenamiento para la evaluación individual con mayor exactitud (que está dentro del *10-fold cross validation* seleccionado) y que tiene una exactitud del 97.01%.

Se puede observar como el comportamiento de las métricas entre los datos de entrenamiento (línea azul) y los datos de validación (línea naranja) tienden a converger y presentan pocas oscilaciones, lo cual es deseable.

La ilustración 17 muestra la matriz de confusión resultante del promedio de exactitudes normalizado de la *10-fold cross validation* seleccionada, con el mejor resultado en la clase Claxon y la exactitud más baja en la clase X.

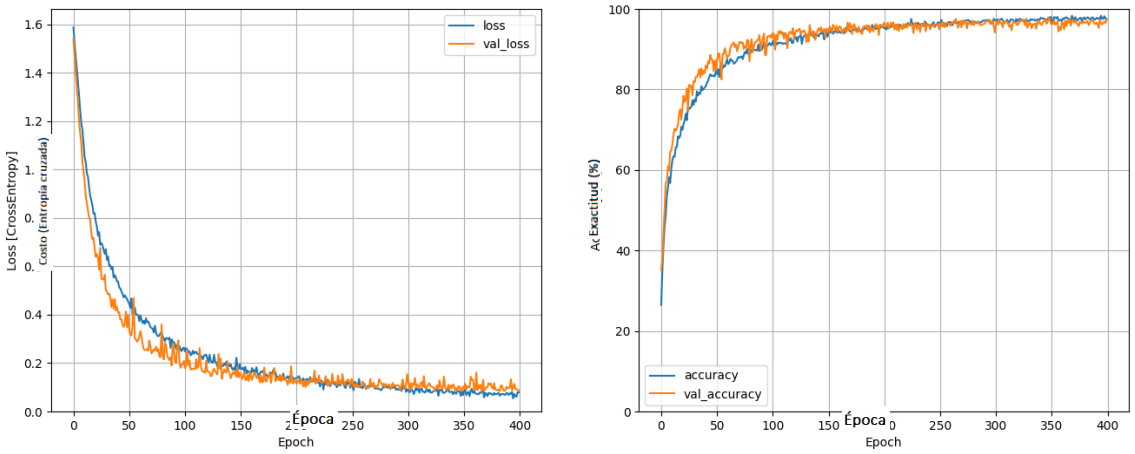


Ilustración 16.- Comportamiento de la función de costo y exactitud del mejor resultado individual obtenido y seleccionado como modelo final en el SE (elaboración propia).

Ilustración 17.- Matriz de confusión resultante de la evaluación del modelo con el conjunto de prueba (elaboración propia).

Con esto se finaliza el diseño, validación y resultados del modelo de CNN que será el encargado de realizar la inferencia en el embebido. En el siguiente apartado, se describirá cómo se implementó dentro del microcontrolador; las primeras pruebas usando una tarjeta de desarrollo comercial para posteriormente ser desplegado en una tarjeta de diseño personalizado para el prototipo del sistema.

3.2 Implementaciones en el microcontrolador

En este apartado, se distinguen dos etapas realizadas durante las iteraciones. La primera consiste en la implementación y pruebas del modelo de CNN en una tarjeta comercial y la segunda etapa es el diseño, implementación y pruebas del SE propio del prototipo. Se describe detalladamente la evolución del proceso de captura en tiempo real del audio a través de los micrófonos MEMS, así como diferencias numéricas entre los tipos de datos a lo largo de los cálculos necesarios para implementar la inferencia de la red.

Se decide trabajar con los microcontroladores y dispositivos de la marca *STMicroelectronics* y sus entornos de desarrollo, después de una comparativa (tabla 7) con diferentes tipos de circuitos integrados en el mercado. Se realiza la programación en el entorno *STM32CubeIDE,* que cuenta con la herramienta de configuración de proyecto *STM32CubeMX*, donde se pueden integrar diferentes bibliotecas y *firmware* para el desarrollo. En específico, para la carga y uso del modelo de red neuronal creado, se utiliza el paquete de expansión para inteligencia artificial *X-CUBE-AI*.

	IDE	Licencia	Resolución ADC	MSPS	CMOS	Encapsulado (mm)	Conectividad	MIPS	Reloj (MHz)	Ancho de palabra
TI TMS320C5517	Code Composer	Freeware Complementos privados	DSP 16 bits Punto fijo	75-250	Si	10x10	USB, McSPI, UART, I2C	Hasta 200	75-250	16 bits
TI C2000 Family TMS320F2802x	Code Composer	Freeware	12 bits	Hasta 4	Si	12x8	UART, SCI, SPI, I2C	60 (15-600)	Hasta 60	32 bits
MICROCHIP dsPIC33CK64MC105	MPLAB	Freeware Complementos privados	12 bits	hasta 3.5	Si	10x7	UART, SPI, I2C	70	Hasta 32	16 bits
STM32L4	STM32CubeIDE	Freeware	12 bits	5	Si	10x10	UART, SPI, I2C, DFSDM	80-120	80	32 bits

Tabla 7.- Comparativa de HW disponible (elaboración propia).

Para las primeras pruebas, se utiliza la tarjeta de evaluación B-L475E-IOT01A2, que contiene un microcontrolador de la familia STM32L4. Más información se puede revisar en la referencia (*STMicroelectronics, 2017*).

A continuación, se detalla la carga del modelo en el microcontrolador, la adquisición de la señal sonora, su tratamiento, preprocesamiento e inferencia.

3.2.1 Carga y validación de la CNN en el embebido

El paquete de expansión *X-CUBE-AI,* permite la carga de modelos de diferentes *frameworks* de inteligencia artificial. Para este caso, el modelo seleccionado y creado con *Keras* puede ser validado dentro del IDE y así revisar si es compatible con las funciones del paquete. Este análisis y validación devuelve métricas útiles como lo es la memoria requerida, complejidad (en MACC, *Multiply-and-accumulate complexity*), direcciones de memoria asignada para su ejecución, descripción de las capas de la red y su demanda en el hardware, entre otros valores. En las siguientes tablas e imágenes se pueden observar estos análisis.

Complexity report (model)

m_id	name	c_macc		c_rom		c_id
0	input_0	|||||	9.5%	|	0.1%	[0]
1	batch_normalization	|	1.9%	|	0.0%	[1]
2	conv2d_1_conv2d	|||||||||||||||	35.5%	|	1.4%	[2]
3	batch_normalization_1	||	2.5%	|	0.1%	[3]
4	max_pooling2d	|	1.3%	|	0.0%	[4]
5	conv2d_2_conv2d	|||||||||||||	30.6%	||||||	14.6%	[5]
6	batch_normalization_2	|	0.4%	|	0.2%	[6]
7	max_pooling2d_1	|	0.2%	|	0.0%	[7]
8	conv2d_3_conv2d	||||||	13.7%	|||||||||||||||	43.7%	[8]
9	batch_normalization_3	|	0.1%	|	0.3%	[9]
10	global_average_pooling2d	|	0.0%	|	0.0%	[10]
12	dense_dense	|	1.5%	|||||	13.6%	[11, 12]
13	dense_1_dense	||	2.6%	|||||||||	24.6%	[13, 14]
14	dense_2_dense	|	0.2%	|	1.4%	[15, 16]

macc=303,307 weights=127,108 act=19,584 ram_io=20

Tabla 8.- Reporte devuelto por X-CUBE-AI de la demanda de recursos del microcontrolador necesaria para el funcionamiento de la red (STMicroelectronic Cube AI).

Getting Flash and Ram size used by the library

Model file:	LogMelModel_800ms_SGD_88_88_5v_1_Hann.h5
Total Flash:	150332 B (146.81 KiB)
Weights:	127108 B (124.13 KiB)
Library:	23224 B (22.68 KiB)
Total Ram:	25252 B (24.66 KiB)
Activations:	19584 B (19.12 KiB)
Library:	5648 B (5.52 KiB)
Input:	3840 B (3.75 KiB included in Activations)
Output:	20 B

Tabla 9.- Reporte devuelto por X-CUBE-AI de la memoria requerida del microcontrolador (STMicroelectronics Cube AI).

Se puede observar en las tablas previas cuáles capas son las que más demandan proceso y memoria en el microcontrolador (c_macc y c_room respectivamente). Por ejemplo, en la tabla 8, el proceso con m_id 2, que es una capa convolucional, es la capa que necesita más operaciones (MACC), lo cual es el 35.5% de las operaciones totales. Por otro lado, m_id 8, que es otra capa convolucional, es la que requiere más memoria no volátil para sus parámetros de kernels. En la tabla 9 podemos observar los requerimientos totales en memoria FLASH y memoria RAM y qué cantidad es usada por sus variables en sus diversos procesos. Para este caso, la memoria FLASH usada es de aproximadamente 146KB, lo cual nos da pauta de pensar en un microcontrolador con al menos 512MB de memoria. Por otro lado, el uso de RAM es de 26.66KB. Estos valores son tomados en cuenta para la selección del microcontrolador en el SE. Para el caso de la tarjeta comercial, las capacidades del microcontrolador con el que se trabaja son suficientes.

Otra información relevante es el tamaño y las fronteras de las direcciones de memoria asignada para que la red trabaje. Esta información es necesaria para cuando se realice el ingreso de datos a la capa de entrada y la inferencia.

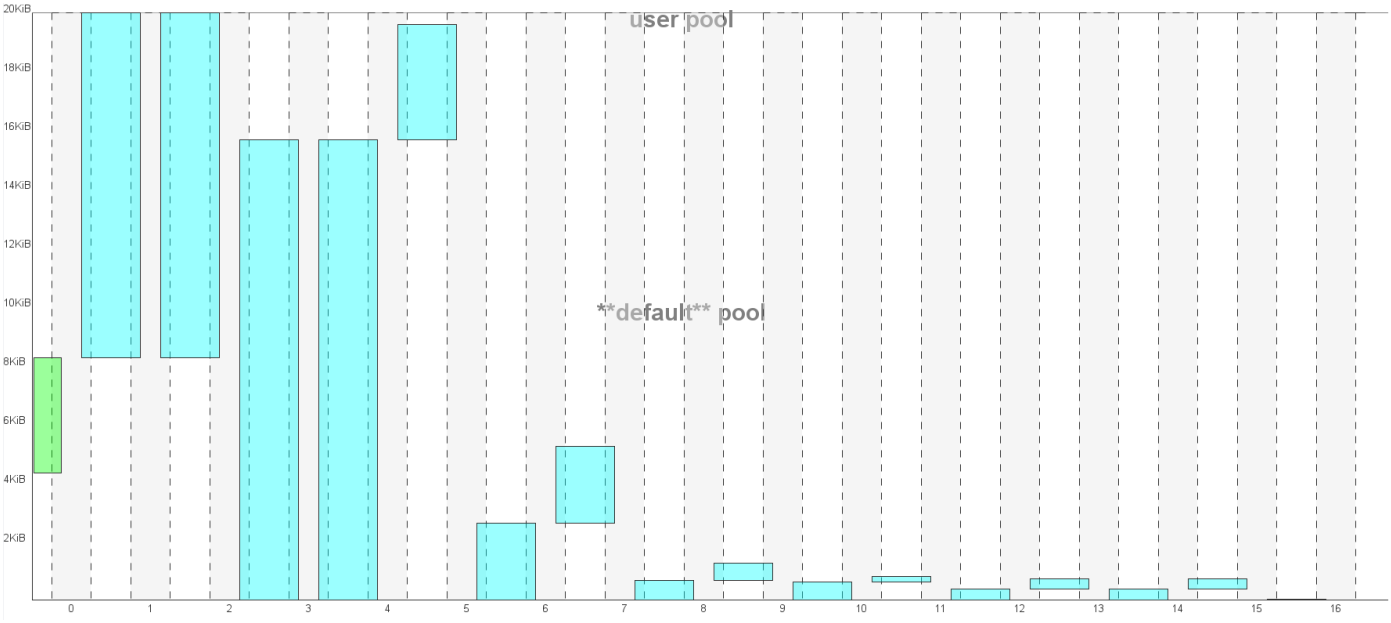


Ilustración 18.- Memoria asignada para guardar los valores dinámicos resultantes de las funciones de activación (Pool) (STMicroelectronics Cube AI).

Se puede observar que los datos de entrada son almacenados en el mismo espacio de la *pool* para el proceso de las funciones de activación (bloque verde, ilustración 18). Con esto se eficientiza el uso de memoria al no requerir más RAM para los datos de la capa de entrada. También se ve la dirección de inicio y fin asignada para tal fin. Los últimos bloques son casi imperceptibles ya que el espacio de memoria necesario para su proceso es muy poco comparado con las capas anteriores.

Una vez analizada y validada la red, se puede generar el código de apoyo necesario para ejecutar las inferencias de la clasificación en el embebido, con esto, los pesos y activaciones estarán listos en el proyecto para su uso.

3.2.2 Características de la adquisición de sonido ambiental

El micrófono digital MEMS MP34DT de STM tiene una salida PDM (*Pulse-Density Modulation*) y además necesita un reloj en un rango de uno a 3 MHZ para su funcionamiento. Para lograr la comunicación entre el micrófono y el microcontrolador y, además, realizar la adecuación de la señal PDM a una interpretativa en amplitud PCM (*Pulse-Code Modulation*), se utiliza el periférico DFSDM (*Digital Filter for Sigma-Delta Modulator*), donde se puede aplicar un filtrado y un *decimador* (ilustración 19). Este periférico se configura para ofrecer al micrófono una señal de reloj de 2 MHz y leer los datos con una frecuencia de adquisición de 16 KHz (frecuencia de muestreo f_S) para su posterior conversión a PCM de 16 bits mediante el filtro y decimador mencionado. Así, se hace coincidir la f_S de los audios de entrenamiento con la frecuencia de adquisición. Además, para liberar la carga en el CPU, se hace uso del almacenamiento DMA (Direct Memory Access). Para esto, se escoge un buffer de 2048 muestras y se realiza el análisis de cada captura a la mitad y al final de la conversión (cada 1024 muestras), para así lograr un trabajo simultáneo; mientras la primera mitad está siendo procesada, la segunda mitad está siendo adquirida para su posterior proceso. Con una f_S*=16 KHz,* cada bloque de 1024 se captura en 64 ms, tiempo de referencia para realizar las operaciones de extracción de características.

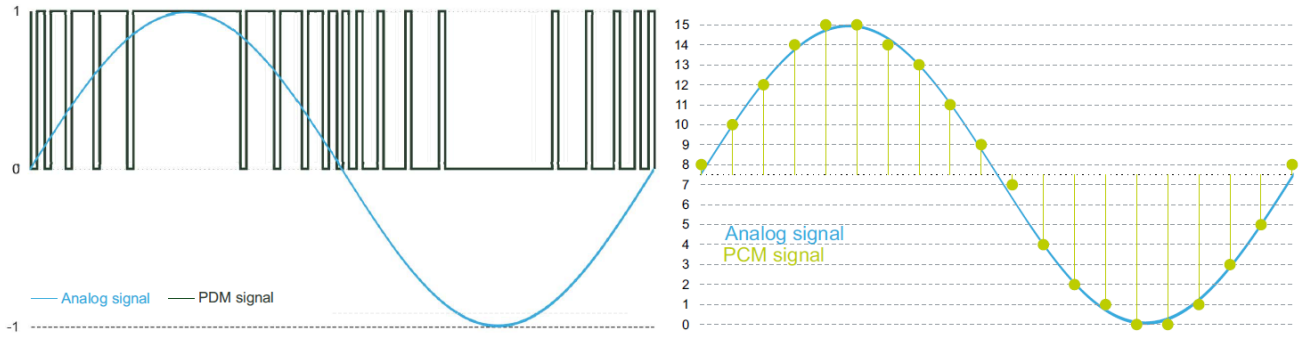


Ilustración 19.- Conversión PDM a PCM (STMicroelectronics).

Cabe mencionar que cada micrófono tiene ligeras diferencias en su respuesta. Instrumentos de medición acústicos de buena calidad (sonómetros) son necesarios para poder calibrar esta etapa de la mejor manera (selección de filtro y primera corrección de *offset*). Para este proyecto, fueron usadas herramientas no tan especializadas (aplicaciones a través de celular) para poder hacer la selección de la configuración de los valores DFSDM para la adquisición estéreo, un punto de mejora que se detallará en la conclusión de este documento.

3.2.3 Preprocesamiento del audio capturado

Corrección de *offset*

Una vez definido el proceso de adquisición, las muestras deben ser tratadas para obtener datos válidos. La muestra adquirida es almacenada en 32 bits, pero sólo los 24 bits más significativos contienen información real del sonido por lo que se realiza un desplazamiento de 8 bits a la derecha. Además, se necesita hacer una corrección de *offset* ya que no constante a lo largo de la adquisición. La estabilidad del *offset* es necesaria para lograr el disparo del preprocesamiento e inferencia sólo en ciertas condiciones de intensidad de sonido y así evitar una constante activación del actuador que retroalimentará al usuario (sensibilidad del dispositivo).

Se probaron dos algoritmos de compensación de *offset*, uno a través del valor máximo y mínimo de 1024 muestras, y otro mediante el promedio de *n* muestras. En siguiente figura se puede observar la señal como es directamente adquirida.

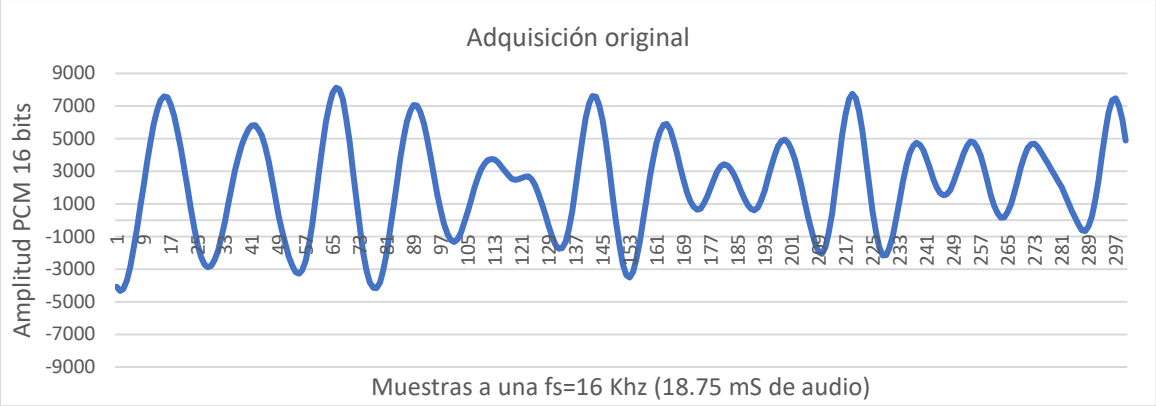


Ilustración 20.- Señal adquirida con offset diferente de 0 (elaboración propia).

Para la corrección por valor máximo y mínimo, se obtienen estos valores de 100 muestras crudas, se suman y se dividen entre 2. Este valor será restado en el siguiente paquete de 1024 muestras, donde se hará nuevamente la actualización del valor de *offset* y así sucesivamente. Es decir, el *frame* actual sirve para corregir el siguiente. Esto se hizo para tratar de salvar tiempo de procesamiento, al no tener que recorrer nuevamente las muestras para su corrección, sino haciéndolo sobre la marcha durante la operación de desplazamiento. Los resultados eran buenos, aunque en ocasiones podían no ser estables debido a valores atípicos de alguna muestra (un valor máximo o mínimo excesivo).

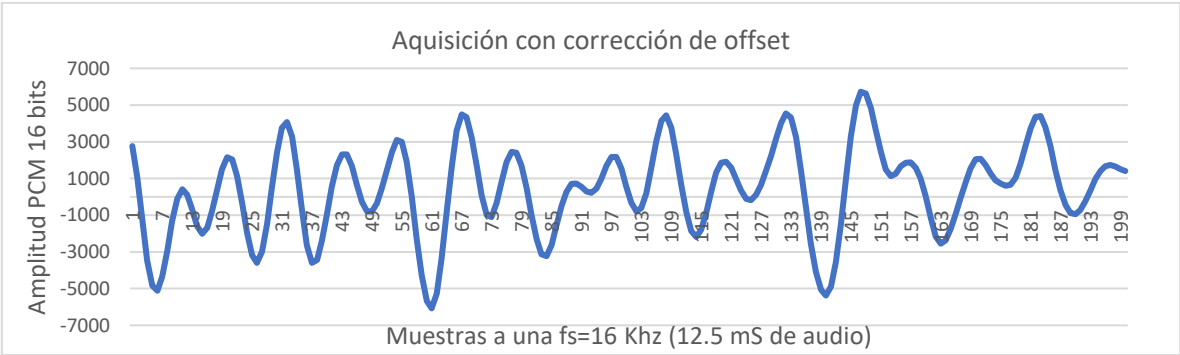


Ilustración 21.- Señal aplicando la corrección de offset (elaboración propia).

Para la corrección por promedio, se realizaba su cálculo y corrección al momento en *n* cantidad de muestras divisoras de 1024 (64 o 128). Es decir, aquí sí se recorrían nuevamente las muestras para aplicar la corrección a las mismas con

que el promedio fue calculado. Esto permitía mayor estabilidad y fue la manera final de corrección que se implementó. La señal con offset corregido por promedio se muestra en la ilustración 21.

Criterio de intensidad de sonido válido

Una vez estable y con un offset cercano a 0, se estableció el criterio de umbral para determinar que un sonido tiene la intensidad necesaria para entrar al preprocesamiento e inferencia. Nuevamente, aquí se probaron dos opciones: una por comparación de *n* valores superiores a un umbral de amplitud y una por cálculo escalado de RMSE (*root-mean-square-energy)*.

Para el criterio por umbral, se establecieron dos valores que se deben de cumplir: superar una línea imaginaria de amplitud (*threshold)* y que al menos *n* cantidad de muestras (*sOver*) de las 1024 superen tal valor. Para verificar la funcionalidad del criterio se hace uso de una aplicación de sonómetro para celular (*Sound Analyzer App*) y una bocina conectada a un generador de funciones que reproduce una señal senoidal de 400 Hz. Se realiza la lectura en decibeles LAF (una lectura rápida de frecuencias ponderadas). La siguiente tabla muestra algunos valores de configuración y los decibeles que resultan en el disparo del resto del proceso.

Threshold	Mínimo sOver	Dispara en
5000	250	75 dBA
3000	150	69-70 dBA

Tabla 10.- Valor de disparo del proceso de umbral con diferentes valores de configuración (elaboración propia).

Para el criterio de intensidad por RMSE, fue necesario hacer una sumatoria de cada valor de la muestra, acorde a la ecuación $\sqrt{\frac{1}{N}\sum_{n}|x(n)|^2}$. Sin embargo, la sumatoria de 1024 muestras de 32 bits elevadas al cuadrado podría desbordar la variable. Es por esto por lo que se escaló cada muestra atenuándola 5 veces, se elevaba al cuadrado, realizaba la sumatoria, la división por 1024 y finalmente la

obtención de su raíz cuadrada. Así, un valor de umbral de RMSE de 6 fue el determinado para hacer disparar la señal, donde una habitación en silencio devuelve valores menores a 1 y con un ruido ligero es menor a 3. Además de esto, fue necesario aplicar un filtro pasa bajas IIR sencillo, para amortiguar oscilaciones drásticas del valor RMSE. El filtro quedó determinado de la forma $P(z)=0.33-0.77z^{-1}$. Ambos criterios de volumen tuvieron buenos resultados. El disparo a un volumen configurable fue estable (mediante mediciones de presión sonora LAZ) y muy importante para poder tener pruebas controladas. Se decidió por el criterio RMSE al ser datos relevantes con los que se logró el algoritmo de detección de la dirección de llegada del sonido, DOA (*direction of arrival*). Estos valores de RMSE se detallarán de mejor manera en dicho apartado.

Extracción de características y almacenamiento

Una vez validado que la señal de audio tiene la intensidad suficiente para ser clasificada, se procede con la extracción de las características representativas que serán los datos de la primera capa de entrada de la CNN, como se ha descrito en el capítulo 3.1.2 (FFT con ventaneo, transformación a frecuencias MEL, operación logarítmica).

En primera instancia, se realiza la FFT para valores reales. Este algoritmo es parte de la biblioteca CMSIS_DSP de ARM, y es posible de utilizar con los microcontroladores que tengan esta arquitectura, en específico los que cuentan con unidad FPU (*floating point unit),* que son los *cortex* M4 y M7. Se debe de configurar el proyecto en el entorno de desarrollo para poder usarla. La instrucción *arm_rfft_fast_f32* usa dos vectores uno de entrada y uno de salida para poder obtener las componentes en frecuencia de la serie numérica, en este caso de 1024 muestras. El proceso se realiza mediante apuntadores a memoria para ser más eficiente. Una vez calculada, se obtiene un nuevo arreglo de 1024, pero esta vez de números complejos, donde los índices pares guardan la componente real, y los nones la imaginaria. Se obtiene la magnitud de estas componentes y se tiene como resultado 512 valores que es lo esperado de este algoritmo.

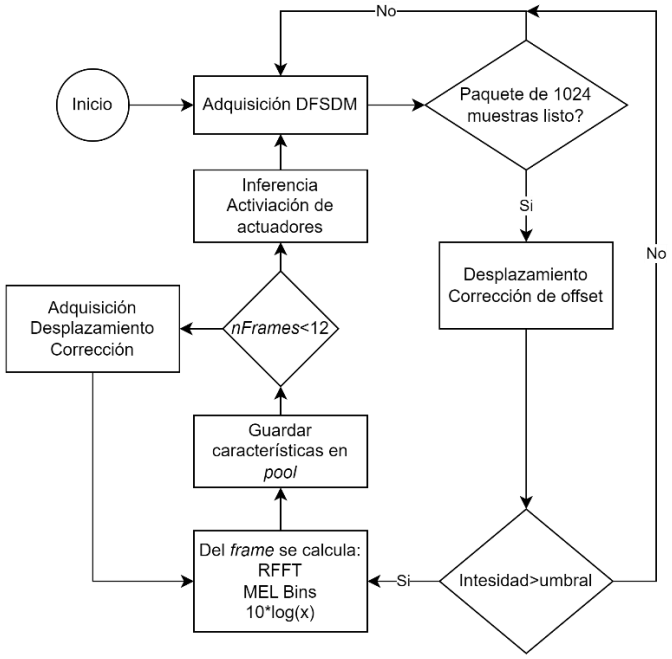
Posterior de la transformación del dominio del tiempo a la frecuencia, se realiza el cálculo de los *bins* correspondientes a la equivalencia en las frecuencias MEL. Para ello, es necesario obtener la matriz de coeficientes del banco MEL y realizar el producto punto con el vector de magnitudes en frecuencia. Al determinar un número de 80 *bins* (misma configuración que en la extracción de características para los audios de entrenamiento del modelo), la matriz de coeficientes del banco MEL tiene una dimensión de 512x80 elementos. Para obtenerla, se creó un algoritmo en Python para exportar los valores generados por la instrucción de *TensorFlow* (ya usada en el entrenamiento), y poder sumarla al proyecto de programación del microcontrolador. La matriz de valores flotantes de 32 bits es cargada dentro de un *.h* y asignada a memoria de programa ya que son valores constantes.

Ya con los coeficientes del banco MEL listos, los 512 valores frecuenciales lineales son multiplicados en la forma correspondiente de producto punto con la matriz MEL (multiplicación de elementos de índice *i* y suma acumulativa) reduciendo a 80 MEL bins. Una vez obtenido el valor, cada bin es transformado a decibeles mediante la operación *10*log10f(Mel_bin)* y guardado en la *pool* de memoria, en la capa de entrada de la CNN (ver capítulo 3.2.1).

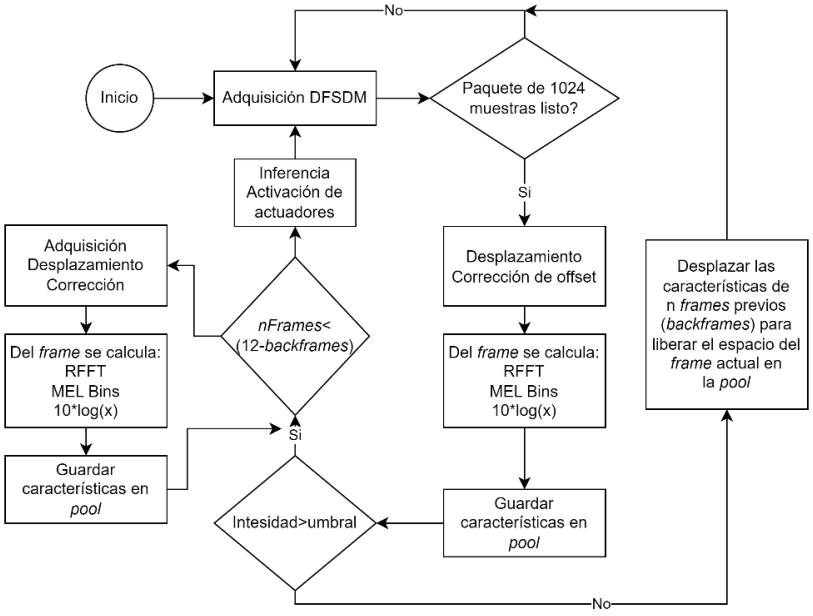
Lo anterior descrito es repetido *n* veces, de acuerdo con el número de segmentos de muestras determinado para la extracción de características. En este caso son 12 segmentos, que en conjunto dan 0.8 segundos de audio capturado a una frecuencia de muestreo de 16 KHz (12 *frames* con 80 *bins* cada uno).

Para determinar cuándo iniciar el proceso de cálculo de las características descritas, varias pruebas fueron realizadas y múltiples combinaciones de los diferentes procedimientos involucrados. Se detallan dos formas de abordar el proceso de extracción de características y sus diferencias. En la primera (ilustración 22, a)), solo hasta que el volumen de la señal de entrada es suficiente, es cuando se inicia con las extracciones de características, su acomodo en la *pool* de la capa de entrada de la CNN y la posterior inferencia. En la segunda (ilustración 22, b)), la extracción de características se realiza constantemente y la *pool* de memoria es

desplazada y sustituida con cada nuevo *frame*. Esto crea un respaldo de *frames* previos hasta que uno de ellos sea suficiente para continuar con la inferencia.



a)



b)

Ilustración 22.- Dos algoritmos para el cálculo de características y almacenamiento (elaboración propia).

Estos dos algoritmos fueron creados debido a la etapa de validación de inferencias (tratada a detalle en el apartado siguiente). Se creó un procedimiento de transmisión serial para poder graficar en *Python* el espectrograma MEL resultante y poder hacer comparaciones con el gráfico de un audio conocido.

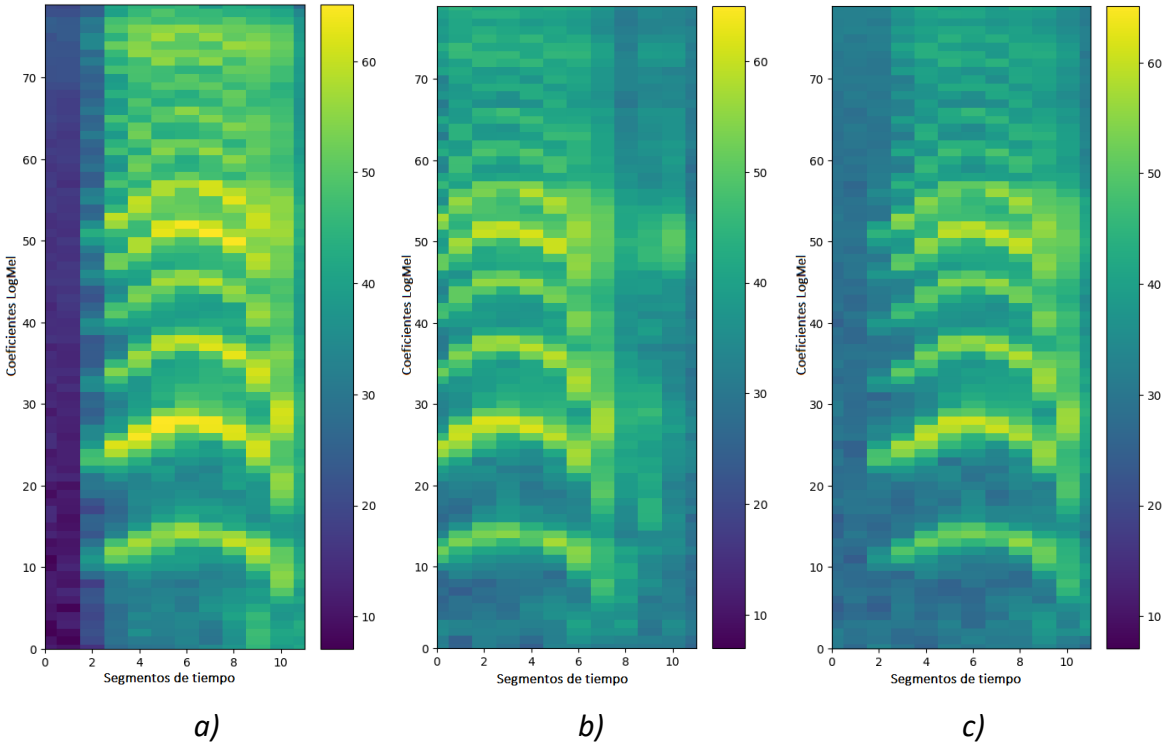


Ilustración 23.- Comparaciones de espectrogramas MEL. a) Datos desde archivo de audio. b) Captura por embebido con algoritmo de almacenamiento hasta superar volumen. c) Captura por embebido con algoritmo de frames de respaldados (elaboración propia).

Se tomo como base un audio de llanto de bebé en condiciones usuales. La ilustración 23 muestra 3 espectrogramas de frecuencias MEL, es decir la matriz de características de 12x80 (12 *frames,* 80 frecuencias MEL). Se puede observar en a) que las características extraídas desde el archivo de audio tienen los primeros *frames* con valores muy bajos (casi 0). Esto debido a que hay casi silencio antes de iniciar el llanto. Por su parte, b) y c) son capturas desde el embebido y no tienen esa respuesta cercana a 0 en condiciones de poco ruido. Vemos también en b) (el espectrograma que corresponde al criterio de extracción y almacenamiento hasta que supera el umbral de volumen), que hay una pérdida de *frames* iniciales, ya que

estos no tenían el suficiente volumen para activar el proceso de extracción y almacenado. Esto afecta la inferencia en algunas clases y es por lo que se llegó al segundo algoritmo de *frames* de respaldo previos. Esto se puede observar en c), donde también es la captura desde el microcontrolador, pero utilizando el segundo algoritmo. Aquí se determinaron 3 *frames* de respaldo que son desplazados y sustituidos constantemente hasta que el *frame* capturado supere el volumen umbral. Se puede observar mayor similitud con a), lo cual provoca una mejor inferencia de la red. Hay una compensación que tomar en cuenta entre volumen umbral (sensibilidad al sonido) y la cantidad de *frames* de respaldo, ya que están directamente relacionados e influyen en buena manera en la inferencia de la clase.

Listo los datos en la memoria, se detiene el proceso de captura y conversión (DFSDM detenido) y se realiza la inferencia para la clasificación del sonido. Los resultados de esto se describen en el apartado 3.4.

Compensación de volumen del sonido

Como última parte del preprocesamiento, se describe otro pormenor que influye en el proceso de inferencia.

Se observó que la intensidad del sonido afecta directamente a las inferencias. A una mayor intensidad, la probabilidad de una correcta clasificación era mayor. Aunque se supere el volumen de sensibilidad, el cual es configurable, no garantiza que las muestras de audio capturadas tengan el valor suficiente para una correcta inferencia, y colocar el umbral en un valor muy alto haría que el sistema dejara pasar sonidos de emergencia que sean necesarios detectar. Además, la posibilidad del sistema de captura en estéreo hace que si el sonido está muy orientado a un solo lado (no de frente al usuario) el otro tenga bajos valores por la barrera acústica de la cabeza. Realizar la extracción de las características y su almacenamiento desde un solo micrófono no es factible, además que las diferencias de amplitudes influyen en el algoritmo de DOA. Por esto, se procedió a compensar la señal en el tiempo, de tal manera que cada muestra es escalada a partir de una razón calculada con un valor de referencia RMSE.

Una vez obtenido el valor RMSE del *frame* de 1024 muestras, se obtiene la razón a partir de un valor de referencia de RMSE de entre 150 y 200. Este valor fue determinado al hacer diferentes pruebas de adquisición y escalamiento. Por ejemplo, una señal de 80dB de presión sonora (medida a la misma altura que el micrófono) tiene un RMSE de aproximadamente 85 y un valor pico PCM cercano a los ±10000. Al hacer la proporción *RMSE_frame/RMSE_Referencia*, se tendría un valor de 2, lo que haría que la señal ahora tuviera un valor pico de alrededor de los ±20000. Esta proporción para compensar el volumen es aplicada a los datos que se usarán para la extracción de características y la inferencia (a los dos micrófonos de la captura estéreo), para así tener una mayor uniformidad independiente de la dirección por la que el sonido esté llegando al sistema. Los resultados de evaluaciones con y sin la compensación son mostrados en el apartado 3.4. Cabe mencionar que para el algoritmo de DOA, esta compensación no es usada ya que este depende de la diferencia de amplitudes entre el micrófono izquierdo L y el micrófono derecho R.

3.2.4 Dirección de llegada del sonido (DOA)

Como ya se mencionó en las funciones del sistema, el dispositivo debe tener la capacidad de determinar la orientación aproximada de dónde proviene el sonido, y reflejarlo de alguna manera al usuario. Para este caso, diferentes intensidades de motores vibradores serán los que indiquen a la persona con la discapacidad auditiva la probable dirección de llegada del sonido.

Para ello se intentaron varias opciones. El primer algoritmo probado fue uno de correlación, basado en el principio de retraso de la llegada de la señal debido a la propagación de la onda sonora por un medio y la distancia existente entre los micrófonos. Es un algoritmo menos costos que otras opciones como correlación cruzada por transformación de fase (GCC-PHAT). La principal desventaja en un inicio es que fue probado en la tarjeta comercial (la cual tiene dos micrófonos MEMS), pero la distancia entre ellos era muy corta para ver diferencias de retraso

con la frecuencia de muestreo elegida. Fue hasta que se ensambló el prototipo que se pudo probar más a detalle este algoritmo de correlación, notando entonces claramente el retraso. Sin embargo, el bloqueo acústico provocado por la cabeza deformaba la señal capturada por el micrófono más alejado de la fuente sonora provocando falsas determinaciones de la cantidad de muestras de retraso. El algoritmo funcionaba correctamente mientras los micrófonos estuvieran alejados y tuvieran un campo libre entre ellos. Por ello, se buscó otra alternativa que recayó en los valores ya calculados de RMSE.

Se obtuvieron varias lecturas de RMSE a diferentes presiones sonoras. Las condiciones fue un cuarto regular en silencio, reproduciendo ruido blanco con una bocina a volumen constante en cada etapa, con una medición de presión sonora de tipo LZeq a través del sonómetro de la aplicación celular. Los resultados obtenidos se pueden observar en la siguiente tabla.

	Der		2/3 Der		1/3 Der		Frente		1/3 Izq		2/3 Izq		Izq	
dB Frente	RMSE L	RMSE R	RMSE L	RMSE R	RMSE L	RMSE R	RMSE L	RMSE R	RMSE L	RMSE R	RMSE L	RMSE R	RMSE L	RMSE R
60	6	10	6	8	7	8	6	6	8	5	9	6	10	5
65	7	17	8	11	9	11	11	11	10	8	15	6	16	6
70	10	25	10	20	13	19	13	15	17	11	22	10	28	11
75	16	38	15	30	20	30	21	25	26	19	33	17	44	16
80	29	80	25	55	35	55	38	46	48	34	63	30	79	29

Tabla 11.- Valores de RMSE medidos a diferentes SPL y diferentes orientaciones (elaboración propia).

Analizando los datos, se buscó una relación en su comportamiento. Después de varias pruebas para poder determinar alguna, se llegó a la siguiente ecuación: $(RMSE\_L - RMSE\_R)^2 / (RMSE\_L^2 - RMSE\_R^2)$, que aporta el signo necesario para saber el lado predominante de llegada y en una proporción casi constante para todos los valores de SPL. La ecuación se simplifica en $(RMSE\_L - RMSE\_R) / (RMSE\_L + RMSE\_R)$, la cual se proporciona para obtener una aproximación del ángulo. Completamente a un lado, el valor del cálculo es cercano a ±0.42, a 2/3 ±0.33, a 1/3 ±0.19 y de frente ±0.08. De aquí, determinando el valor más extremo

de -0.5 a -90° y 0.5 a 90°, se puede aproximar la detección de llegada del sonido con cierta confiabilidad, pero sin una resolución muy precisa en el ángulo, lo cual no es tan relevante para este trabajo. A partir de esto, se determinaron 5 regiones, que aportarán diferentes intensidades sonoras a los motores vibradores: solo un motor vibra (aprox. ±90°), un motor vibra a media intensidad que el otro (aprox. ±45°) y los motores vibran con la misma intensidad (aprox. 0°). Un gráfico de estas regiones es mostrado a continuación.

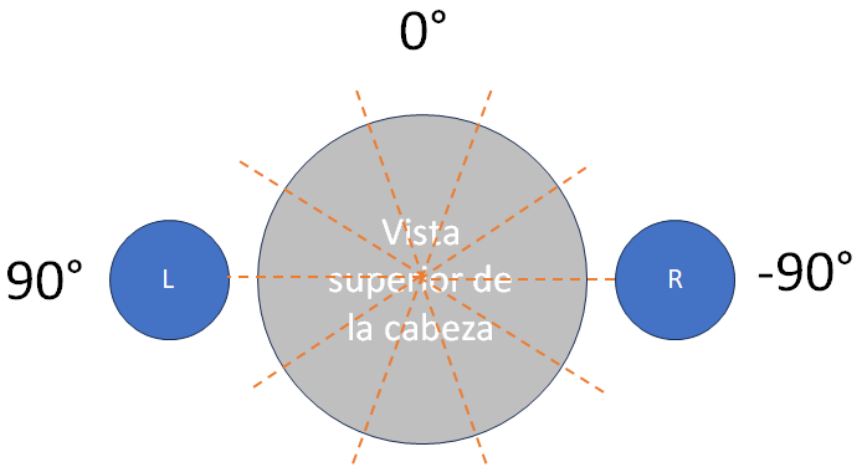


Ilustración 24.- Regiones de DOA (elaboración propia).

Esto ayuda a dar al usuario una aproximación de dónde viene el sonido, sin importar si está frente a él o no, ya que la proporción se mantiene aun cuando la fuente sonora está detrás de la persona.

Para finalizar este apartado cabe que mencionar que todas estas implementaciones (exceptuando DOA) sucedieron tanto en la tarjeta comercial como en el diseño del PCB propio del prototipo. Algunos resultados de cada uno se muestran en 3.4 y muy ligeras diferencias fueron encontradas, ya que los microcontroladores en ambas tarjetas fueron de la familia STM32L4, teniendo como principal diferencia sus capacidades de memoria FLASH y RAM (el microcontrolador de la tarjeta comercial tiene mayores valores en un empaquetado más grande), sin que esto fuera una dificultad ya que se consideró al momento del diseño de PCB. Se dedica el siguiente punto al detalle de este diseño.

3.3 Diseño de PCB para prototipo del SAISE

Una vez diseñada la arquitectura de CNN y teniendo gran parte de la programación de los algoritmos necesarios para implementar la clasificación, se tuvo una proyección de los requisitos de memoria y periféricos necesarios para seleccionar un microcontrolador que cumpliera con ellos, además que el empaquetado fuera el adecuado para un dispositivo portable como el indicado y tuviera opciones de bajo consumo. Se seleccionó entonces el microcontrolador STM32L4P5CEU6 que cumplía con lo anterior y tenía un empaquetado de los más pequeños que maneja STM; UFQFPN48 de 7.3x7.3mm.

El diseño se realizó en *Circuit Maker*, una versión gratuita de *Altium* que trabaja desde archivos en la nube. Se tuvieron que diseñar varias “huellas” de componentes (*footprints)* como la de los micrófonos MEMS entre otros.

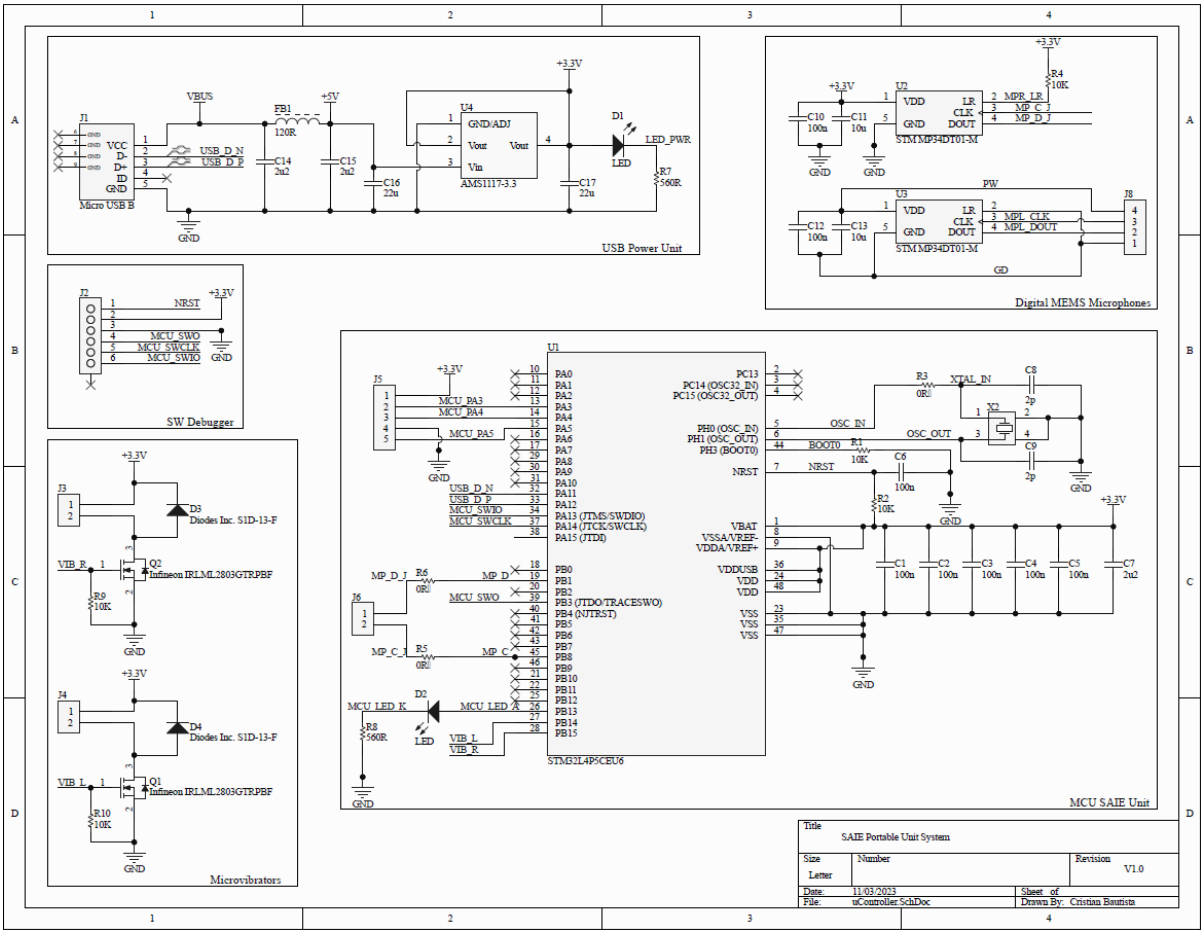


Ilustración 25.- Diseño del esquemático para el prototipo SAISE (elaboración propia).

Una vez hecho el esquemático y el trazado de las rutas, el sistema renderizado quedó de la siguiente manera.

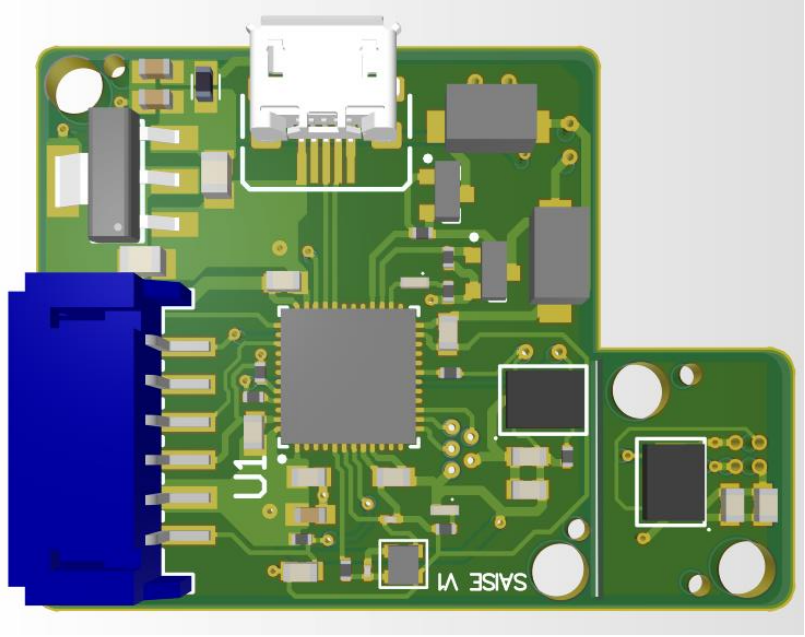


Ilustración 26.- Renderizado de la tarjeta PCB final (elaboración propia).

Se diseño en dos capas. Se agregó un puerto micro USB para comunicaciones y alimentación del sistema. El componente de color azul es un conector para la programación mediante *Serial Wire*. La tarjeta que sobresale es separable ya que es el otro micrófono, y se une mediante cable soldado a la placa principal. Lo mismo para los motores vibradores. La dimensión final de la tarjeta principal es de 3x3 cm.

Al ser en su totalidad componentes de montaje superficial (SMD) y rutas muy pequeñas, se decide hacer la manufactura y ensamble a un tercero, para sólo realizar la soldadura de los cables que conecten al micrófono y vibradores. La compañía elegida es la empresa china *PCBWay*, especializada en la creación de PCB y que ofrece además servicio de compra y ensamble de componentes, a buena calidad y a un precio aceptable. El costo de la tarjeta fue de $1100.00 pesos mexicanos aproximadamente (PCB, componentes y ensamble), más $750.00 por costos de importación y manejo por parte de la paquetería. A partir de realizado el

pago y haberse mandado los archivos correspondientes (BOM, *Gerber*, *NCDrill* y *PickPlace)*, el tiempo de entrega hasta el domicilio fue 20 días aproximadamente. La tarjeta ensamblada se muestra en la siguiente imagen.

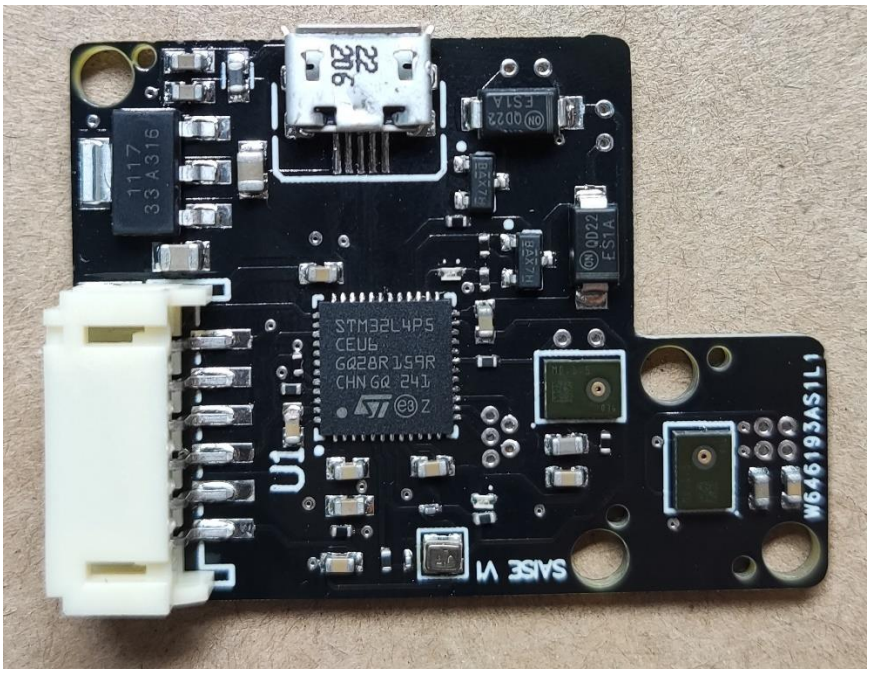


Ilustración 27.- Ensamble final de la tarjeta (fotografía propia).

El diseño fue correcto desde el inicio y no hubo necesidad de realizar ninguna modificación o corrección, sin embargo, ciertos puntos de mejora fueron detectados para futuras versiones, lo cual es discutido en la sección de conclusiones.

3.4 Resultados de la integración del sistema

Para validar el funcionamiento se realizaron dos pruebas diferentes: datos idénticos enviados por comunicación serial y datos adquiridos a través del micrófono. En primer lugar, era necesario comparar la resolución de los datos de las diferentes operaciones y comprobar que no existían diferencias significativas. Para ello se tomaron audios de prueba de referencia. Con estos y mediante el modelo seleccionado, se realizaron inferencias individuales en Python para observar los resultados de cada etapa del preprocesamiento y las probabilidades de su clasificación.

3.4.1 Dirección de llegada del sonido (DOA)

Las muestras de estos mismos audios fueron enviados a través de comunicación serial (*baud rate*=115200, 8 bits, sin paridad) al microcontrolador y se compararon los resultados obtenidos de cada etapa de preprocesamiento (RFFT, MEL, 10*log(x)) con los obtenidos en Python, con el fin de ver la consistencia de las operaciones. Posteriormente se compararon las probabilidades de inferencia obtenidos. Un ejemplo de las diferencias numéricas de cada cálculo es mostrado en la tabla siguiente.

	Prueba por comunicación serial: diferencia en los cálculos de preprocesamiento							
Audio	**Ejemplos de RFFT Python**		**Ejemplos de RFFT STM**		**Ejemplos de MEL Python**	**Ejemplos de MEL STM**	**Ejemplo 10*log Python**	**Ejemplo 10*log STM**
rs_AM442.wav	-67569	+0j	-67569	0j	53703.65	53698.5625	47.300037	47.2996292
	-221067.44	-226959.69j	-221067.438	-226959.703j	47962.97	47962.1523	46.809063	46.8089867
	27399.34	+4235.2227j	27399.3359	4235.22363j	44586.453	44586.3281	46.492027	46.4920197

Tabla 12.- Resultados de preprocesamiento utilizando los mismos datos de audio tanto en Python como en el embebido (elaboración propia).

Se pueden observar diferencias mínimas en los resultados, la mayoría del orden de las milésimas o inferiores. Aunque en etapas intermedias (como la transformación a frecuencias MEL) pueda haber diferencias de unidades, al convertir dicho valor a dB con la operación logarítmica, esa diferencia se vuelve insignificante y finalmente, es dicho valor el que formará parte de los datos de la capa entrada de la CNN. Con esto, se comprueba que la FPU del microcontrolador y su resolución de 32 bits, las funciones de la biblioteca CMSIS y las operaciones programadas, están realizando el cálculo de manera correcta.

Una vez comprobado esto y siguiendo con las pruebas por comunicación serial, se procedió a comparar los resultados de la inferencia, y así comprobar que la carga de la CNN en el microcontrolador y el manejo de memoria era el correcto. Muchos parámetros son los involucrados para el funcionamiento de la CNN en el embebido, y el correcto uso de ellos es imprescindible para una clasificación correcta.

En la tabla 13 podemos observar las diferencias en la probabilidad de las inferencias. Todas fueron coincidentes, es decir, si en Python existía una clasificación correcta o incorrecta, en el microcontrolador sucedía lo mismo con una probabilidad que se pudiera considerar idéntica (diferencia despreciable).

Audio	Clase	Inferencia Python	Inferencia STM
rs_AM481.wav	Bebé	0.0837536300	0.0837596804
	Claxon	0.0543656400	0.0543683842
	Grito	0.0007499100	0.0007498294
	Sirena	0.3691431000	0.3690656130
	X	0.4919877000	0.4920565190
rs_BC442.wav	Bebé	0.0208257500	0.0208233651
	Claxon	0.0014029200	0.0014027065
	Grito	0.9512942000	0.9512985940
	Sirena	0.0228287400	0.0228274092
	X	0.0036483500	0.0036479852
rs_CL441.wav	Bebé	0.0000000480	0.0000000480
	Claxon	0.9994946700	0.9994946720
	Grito	0.0005044995	0.0005045702
	Sirena	0.0000000371	0.0000000371
	X	0.0000006634	0.0000006634
g71.wav	Bebé	0.0000924168	0.0000924188
	Claxon	0.0001035342	0.0001035393
	Grito	0.9995243550	0.9995242360
	Sirena	0.0002575371	0.0002575498
	X	0.0000221471	0.0000221471
rs_Ladrido.wav	Bebé	0.0000002273	0.0000002273
	Claxon	0.0000014566	0.0000014568
	Grito	0.0000004567	0.0000004567
	Sirena	0.0000047655	0.0000047659
	X	0.9999930900	0.9999930860

Tabla 13.- Comparativa de las probabilidades inferidas tanto en Python como en el microcontrolador STM. Clases más altas marcadas (elaboración propia).

Los audios usados corresponden en orden a sirena, bebé, claxon, grito y X (ladrido de un perro) y no forman parte de los usados para el entrenamiento. Estos fueron enviados por serial, preprocesados y finalmente se realiza la inferencia por parte de la CNN. Con esto se comprueba la correcta implementación de la CNN en el embebido.

3.4.2 Pruebas con audios reproducidos y capturados por el micrófono

Para la segunda prueba, se reprodujeron los mismos audios de prueba y otros audios aleatorios en una bocina conectada a la computadora y desde diferentes plataformas (YouTube, SoundCloud, archivos en disco), dentro de un cuarto con poco ruido ambiente. La captura fue realizada por el micrófono (ver algoritmos en 3.2.3 Preprocesamiento del audio capturado: Criterio de intensidad de sonido válido) y se realizaban las inferencias en tiempo real.

Las primeras pruebas corresponden a las realizadas en la tarjeta de desarrollo B-L475E-IOT01A2 y con las siguientes pautas de adquisición y extracción de características: algoritmo de corrección de *offset* por valor máximo y mínimo, criterio de intensidad de sonido válido sin respaldo, sin aplicación de ventana Hann, sin compensación de volumen y CNN de referencia.

Audio	**Clase**	**Inferencias continuas a lo largo del tiempo (5 s)**			
rs_AM481.wav	Bebé	0.000211	0.000061	0.000096	0.000004
	Claxon	0.000018	0.000197	0.000023	0.000261
	Grito	0.006698	0.000741	0.016357	0.001368
	Sirena	0.992756	0.998762	0.983469	0.998359
	X	0.000317	0.000240	0.000055	0.000008
rs_BC442.wav	Bebé	0.000065	0.212035	0.262504	0.973291
	Claxon	0.788390	0.495533	0.051027	0.021767
	Grito	0.210511	0.286688	0.279554	0.000116
	Sirena	0.001017	0.004701	0.406253	0.004385
	X	0.000016	0.001043	0.000660	0.000441
rs_CL442.wav	Bebé	0.000053	0.000055	0.00002	0.000029
	Claxon	0.772756	0.939617	0.165773	0.955577
	Grito	0.047858	0.001055	0.000794	0.003213
	Sirena	0.178701	0.058452	0.833246	0.041142
	X	0.000633	0.000821	0.000167	0.000039
g71.wav	Bebé	7.00E-06	0.001945	No hay registro por ser un sonido muy corto	
	Claxon	0.000002	0.008997		
	Grito	0.010002	0.045581		
	Sirena	0.989988	0.936201		
	X	0.000001	0.007276		
rs_Ladrido.wav	Bebé	0.001206	0.000714	0.004753	0.000911
	Claxon	0.000104	0.001883	0.000056	0.000012
	Grito	0.000141	0.00129	0.000243	0.000044
	Sirena	0.013126	0.021432	0.929681	0.983862
	X	0.985423	0.974682	0.065267	0.015171

Tabla 14.- Inferencias con datos de audio capturados por micrófono (elaboración propia).

Se usan los mismos audios que las pruebas seriales para poder hacer la comparativa de datos idénticos contra datos capturados por micrófono. Se puede observar como las inferencias cambian a lo largo del tiempo, debido a que estos audios no contienen en su totalidad sonidos exclusivos de los buscados, y pudieran estar sucediendo un poco después del inicio. También hay ocasiones que la clasificación no fue correcta, y necesito otro segmento de audio para realizar la clasificación correctamente. Algo resaltable es que, en existencia del sonido de emergencia, la CNN devuelve un valor de los 4 relevantes y no lo deja pasar con la clase X, lo cual da la pauta de pensar en una super clase de "emergencia" que pudiera generalizar a las demás.

Después de esto, se reproducen 150 audios de los sonidos en condiciones reales a través de la plataforma de *YouTube* (30 audios por clase) y se proponen las variantes del algoritmo de criterio de intensidad de sonido válido con respaldo de *frames*, compensación de volumen, aplicación de ventana Hann y variaciones correspondientes en el modelo de CNN. Se realizan varias ejecuciones de este proceso para tener comparativas de la exactitud de los diferentes modelos y la influencia de los algoritmos de captura y compensación de volumen.

		% de exactitud de la clase					Error de clasificación mayormente con:				
Características de la prueba	**% exactitud Promedio**	**Sirena**	**Bebé**	**Claxon**	**Grito**	**X**	**Sirena**	**Bebé**	**Claxon**	**Grito**	**X**
Modelo d1, d2=88,88 Sin ventana Hann Captura sin *frames* de respaldo Sin compensación de volumen	73.12	73.33	48.27	90	80.65	73.33	Grito	Sirena Grito	Grito	Sirena	Claxon Sirena
Modelo d1, d2=88,88 Sin ventana Hann Captura sin *frames* de respaldo Con compensación de volumen	77.06	96.66	64.28	86.21	54.84	83.33	Grito	Grito Sirena	X	Sirena	Sirena
Modelo d1, d2=88,88 Sin ventana Hann Captura con *frames* de respaldo (2) Sin compensación de volumen	75.9	83.33	82.76	56.67	76.67	80	Grito	Grito	Sirena Grito Bebé	Sirena	Bebé
Modelo d1, d2=88,88 Con ventana Hann Captura con *frames* de respaldo (3) Con compensación de volumen	77.82	93.1	79.31	73.33	53.33	90	Claxon Grito	Sirena Claxon	X	Claxon	Claxon

Tabla 15.- Resultados de la clasificación de los 150 audios (elaboración propia).

Los resultados previos de la clasificación y su probabilidad se obtienen en cada reproducción a través de comunicación serial y se visualizan en la terminal *Tera Term*.

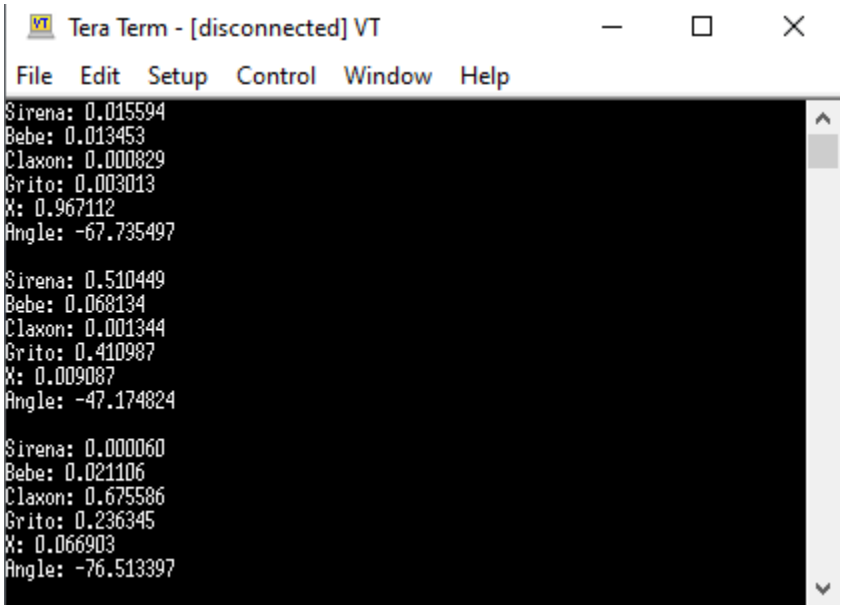


Ilustración 28.- Ejemplo de la visualización de la clasificación, su probabilidad y su ángulo de DOA (elaboración propia).

En la tabla 15 se observan las diferencias entre las distintas redes y aplicación de los algoritmos. En todas las pruebas, una clase es la que ocasiona que el promedio general baje. El mejor resultado general es del 77.82% cuya red es la elegida para cerrar este desarrollo y que supera por un margen muy corto lo buscado en las metas de este proyecto, que era una tasa de clasificación correcta de al menos 75%.

Estas pruebas se realizaron con una adquisición-inferencia de manera continua, sin existir pausas entre cada nueva adquisición. Se observó que esto influye con la clasificación, lo que se supone es debido al momento intermedio de captura de un sonido. Como trabajo futuro, un algoritmo complementario después de la inferencia sería útil, como determinar cuántas inferencias correctas darían paso a una notificación o, utilizar otra serie de características del sonido (como cruce por cero o centroide espectral) para obtener un porcentaje complementario para

fortalecer la decisión de clase, podrían ser de utilidad para afinar la clasificación o evitar cambios significativos entre inferencias continuas. Además, la constante clasificación y notificación puede resultar abrumador para las personas usuarias, por lo que después de la activación de un *tactón*, podría existir una pausa para iniciar nuevamente el proceso. De igual manera, se observó que algunas confusiones son esperables, ya que en ocasiones el llanto de un bebé puede ser clasificado como grito general, también debido a que los audios de entornos reales contienen múltiple ruido de fondo o incluso otros sonidos relevantes, como en el caso de las ambulancias o sirenas de policías, cuyos sonidos muchas veces vienen acompañados de sonidos de claxon.

Desde el punto de vista de super clase de “emergencia” y la no existencia, la clasificación tiene un gran desempeño. En el peor caso, de las 150 reproducciones, solo resultaron 4 falsos negativos y 9 falsos positivos.

		Predicha		
	Superclase	**Emergencia**	**No Emergencia**	
Real	**Emergencia**	116	4	**120**
	No emergencia	9	21	**30**
		125	**25**	

Tabla 16.- Análisis a considerando la clasificación binaria con la superclase “Emergencia” (elaboración propia).

Basado en estos datos, la *Precisión* como clasificador binario es del 92.8%, el *Recall* es de 96.67% y el *score F1* es igual a 94.70%. Estos resultados aun tomando en cuenta el peor caso obtenido, son prometedores y muestran otra posible ruta de aplicación del sistema

3.4.3 Resultados de DOA

El algoritmo de detección de la dirección de llagada tuvo un comportamiento satisfactorio, aunque con una baja resolución. Se definieron 5 regiones dentro de 180° (ver 3.2.4) para realizar una activación de la siguiente manera: sonido

completamente a un lado, sólo un vibrador activado; sonido desde frente-un lado, ambos vibradores activados, un vibrador más intenso que otro y; sonido frontal, ambos vibradores activados a la misma intensidad. La diferenciación del sonido es buena, aunque una activación constante resulta abrumadora. Más pruebas son necesarias para determinar una tasa justa de activación.

En el caso de la confiabilidad de la detección de llegada, se ejecutaron 120 audios en las 5 direcciones de llegada. La correcta asignación de dirección de llegada fue de un 85% de las veces, notando saltos entre regiones vecinas en varias ocasiones. Otro factor que influía en el ángulo de llegada era la línea de percepción del micrófono, es decir, si el sonido llegaba alineado a la misma altura del micrófono MEMS, el ángulo era más certero, mientras si presentaba variaciones en la altura (más por debajo o por arriba del orificio de captura), había inconsistencias en la determinación. Se supone que este comportamiento es debido a la forma de la carcasa del auricular que envolvía el prototipo y los orificios realizados para la captación del sonido. La falta de conocimiento más profundo en cuestiones de acústica no permite ser más determinantes en esta explicación.

Si se considera la confiabilidad entre lado izquierdo, lado derecho y frente, el sistema se comportó muy estable, alcanzando hasta 95% de detecciones de llegada correctas. Estas direcciones podrían ser una alternativa viable para esta aplicación.

3.4.4 Prototipo SAISE

Para el prototipo se utilizaron unos audífonos supra aurales existentes, con el fin de poder probar las características de DOA, las compensaciones de volumen y las inferencias a partir de la captura estéreo. Las dimensiones aproximadas del audífono son de 4.5x6cm por 1cm de ancho, tamaño suficiente para la tarjeta diseñada. La potabilidad se solucionó con una batería de carga externa para realizar unas pocas pruebas de comportamiento, sin ser aún pruebas formales documentadas. Se realizaron 6 perforaciones por lado para permitir la entrada de sonido a los micrófonos, así como una ranura para poder conectar el puerto micro USB a la computadora y poder revisar las inferencias por serial.

Ilustración 29.- Prototipo del SE dentro de unos audífonos comerciales (fotografía propia).

Aunque básica, la adaptación en la forma de los audífonos permitió evaluar varios comportamientos y comprobar las hipótesis presentadas. En primer lugar, se observó que el algoritmo de selección de micrófono para los datos que pasan a inferencia funcionaba correctamente. Es decir, al ser una captura estéreo, no se podría depender exclusivamente de un solo micrófono para la extracción de los datos y su pase a la capa de entrada de la CNN, sino que debía ser adaptativo para lograr así que el sonido de mayor calidad (sin obstrucciones acústicas como la que podría provocar la cabeza) fuera del que se extrajeran los datos. Esto se comprobó al colocar los audios desde diferentes ángulos de llegada y obteniendo resultados consistentes de clasificación entre todas las opciones, sin importar su dirección. También, el prototipo fue de utilidad para valorar el comportamiento de DOA descrito en el apartado previo. Otra función relevante, fue el poder valorara las activaciones de los motores vibradores y comprobar el funcionamiento esperado. Se probó en dos personas, permitiendo así la determinación de los valores de ciclo útil de PWM necesarios para la determinación de la intensidad de vibración para los diferentes rangos de DOA. También fue posible la determinación de los patrones de vibración (*tactones*) propios de cada sonido y evaluar que las combinaciones de tiempos de activación fueran buenas para diferenciar entre ellos.

Mejoras restan para este prototipo, como la posibilidad de un diseño propio que permita un ensamble y cableado adecuado, discreto y con la mejor optimización de espacio, además de poder integrar el sistema de batería dentro del dispositivo, el botón de encendido y apagado y el fácil acceso a los puertos.

Definidos los *tactones* y con un diseño de carcasa propio, trabajo futuro serán las pruebas metódicas para evaluar el funcionamiento con diferentes personas, observar la facilidad de uso, el consumo de energía y la correcta clasificación en campo y durante periodos prolongados.

3.4.5 Indicadores generales

En este apartado se muestran algunas métricas relevantes del comportamiento del sistema.

Tiempo de ejecución del preprocesamiento ≈ 31.6 ms
(Corrección de offset al momento por promedio, cálculo para RMSE, Ventana Hann, RFFT, transformación MEL, operación logarítmica y almacenado, aplicado a un paquete de 1024 muestras, medido con osciloscopio a través de la puesta en alto de una salida digital al inicio del proceso y su conmutación al terminarlo).

Tiempo para la ejecución de la inferencia ≈ 80 ms
(A una frecuencia de reloj de 80 MHz con una complejidad de la red de 303,307 MACC. El valor es cercano con una estimación dada por la documentación de STM de 9 ciclos/MACC para ARM-Cortex M4).

Tiempo total necesario para la notificación ≈ 850 ms
(Captura de aproximadamente 0.8 segundos de audio: 12 paquetes de 1024 muestras, a una frecuencia de muestreo de 16 KHz más el tiempo necesario para realizar una inferencia).

Total de la memoria RAM usada = 67.13 KB, 20.98% de 320 KB

Total de la memoria FLASH usada = 451.59 KB, 88.20% de 512 KB

Discusión y conclusiones

Discusión

Comparando los resultados obtenidos con las propuestas hechas en el estado del arte, el SE presenta aportaciones significativas en algunas áreas como lo son la creación y manipulación de los audios para el enriquecimiento del conjunto de datos, la propuesta de arquitectura de CNN adecuada para las limitaciones de microcontroladores, las técnicas de captura a través de los micrófonos MEMS, la compensación de volumen para la inferencia estéreo y la propuesta para la detección del ángulo de llegada gracias a la forma elegida de audífonos.

Una de las principales comparaciones puede ser realizada con el trabajo de *(Dhruv Jain et al., 2020)*, la cual presenta un sistema de detección de evento sonoro combinando diferentes arquitecturas del sistema (solo reloj inteligente, reloj inteligente-celular, reloj inteligente-celular-nube, reloj inteligente-nube) usando también un método de clasificación mediante diferentes modelos de aprendizaje profundo de CNN. De todos los resultados obtenidos en dicha investigación, hay algunos que pueden servir como punto de comparación para este trabajo. Aunque su máxima cantidad de clases a clasificar podrían ser 20, existía la posibilidad de personalizar el modelo CNN para sólo clasificar 3 clases consideradas de emergencia. Tomando este último caso y con la arquitectura de solo reloj inteligente (arquitectura comparable con la de este trabajo para ser considerado *ML at the edge*) de versión *Android Ticwatch Pro watch* (4×1.2GHz, 1GB RAM), su mejor resultado fue usando la red CNN ya existente VGG, pero en una versión optimizada llamada VGG-*lite* (281.8 MB) alcanzando una exactitud promedio de 97.6% pero con un tiempo de latencia en la inferencia de 3397ms, y 5900ms desde la captura hasta la notificación. Se puede observar que su modelo más exacto, fue también el más lento. Su segundo mejor resultado en las condiciones de 3 clases de emergencia es de 78.1% usando el modelo ResNet-*lite* (178.3 MB), con un tiempo promedio de inferencia de 1615ms y 5s desde la captura hasta la notificación.

Aunque su resultado más alto de exactitud es mucho mejor que el obtenido en este trabajo (77.82% con cinco clases, las cuales afectan en el porcentaje; a mayor número de clases la exactitud tiende a ser menor), es justo mencionar que

es comparable con su segundo mejor resultado de 78.1%, además de considerar que el modelo de CNN desarrollado es significativamente más pequeño y adecuado para SE limitados (147 KB de memoria FLASH y 25 KB de memoria RAM) con un tiempo de latencia en la inferencia de aproximadamente 80ms y 850ms desde la captura hasta la notificación. La diferencia de las latencias juega un papel primordial al ser un sistema de detección de emergencia que, aunque la exactitud no sea alta, una respuesta más rápida del sistema puede ser sumamente relevante para una reacción a tiempo por parte de la persona usuaria. Además, las diferencias entre las capacidades de los embebidos utilizados son grandes. Mientras que *(Dhruv Jain et al., 2020)* utiliza un sistema ya comercial con 4 núcleos de 1.2GHz y 1GB de RAM (y solo desarrolla una aplicación para ser ejecutada por este), el sistema propuesto y diseñado se ejecuta en un microcontrolador de bajo consumo, con un solo núcleo a 80MHz y 320 KB de memoria RAM. Esta diferencia es una gran ventaja para lograr un producto final de costos más asequibles para la mayoría de la población, comparado con un producto de una gama más alta.

Otra comparación posible es con el trabajo realizado en *(Jain et al., 2022)*, el cual consistía en una aplicación móvil (*Protosound)* capaz de ser personalizada, característica del sistema relevante y deseada por los usuarios con la discapacidad y que este trabajo no contempla. La aplicación móvil permitía a los usuarios agregar clases deseadas realizando entrenamientos con pocas muestras sobre una red prototipo ya existente, MobileNetV2 (8MB). La exactitud variaba entre la locación donde se estuvieran realizando las inferencias con un promedio entre todas las locaciones de 87.4%. De la misma forma que el trabajo anterior, la exactitud de esta aplicación fue mayor, sin embargo, el sistema es aplicado en un dispositivo que pudiera ser considerado más de uso general y con mayores capacidades de cómputo como lo es un celular (parámetro común de muchos desarrollos) que, si bien es portable, no está siembre en nuestras manos para poder tomar acción inmediata en caso de algún suceso sonoro de emergencia.

Si consideramos el *score F1* obtenido en esta investigación, aun cuando no sea detallado qué sonido de emergencia es, la posibilidad de notificar que algún sonido de emergencia existe con una exactitud mayor al 92% también puede tener

relevancia para estos usuarios. Aunado, otro tipo de aplicaciones de detección de evento sonoro pueden ser desarrolladas a partir de este principio y con las mismas características como lo pueden ser la personalización para la detección del nombre propio de la persona usuaria o algún sonido relevante para ellos. Incluso fuera del contexto de las discapacidades, este sistema podría aplicarse para otros entornos como casos para motociclistas, detección de sonidos de falla en cierta maquinaria o espacios donde barreras acústicas son necesarias (audífonos de seguridad industriales), pero también la notificación sonora y portabilidad es relevante. En la siguiente tabla se puede encontrar un resumen al respecto.

	(Dhruv Jain et al., 2020)	**(Jain et al., 2022)**	**SAISE**
Dispositivo	Android Ticwatch Pro watch	Celular comercial	Propio
Características del dispositivo	4×1.2GHz núcleos, 1GB RAM	No indicado	1x80MHz núcleos, 320KB RAM
Tiempo de inferencia (ms)	3397	No indicado	80
Tiempo a la notificación	5900	No indicado	850
Modelo y tamaño	VGG-lite 281.8MB ResNet-lite 178.3 MB	MobileNetV2 + Prototypical Networks 8MB	Propio 147 KB FLASH y 25 KB RAM
Número de clases	3	5	5
Exactitud	VGG-lite 97.6% ResNet-lite 78.1%	87.40%	77.82%

Tabla 17.- Tabla comparativa de resultados de SAISE con diferentes investigaciones (elaboración propia).

Respecto al ángulo de llegada del sonido sí existen múltiples mejoras y pruebas por realizar. Las 5 regiones planteadas en 180° para la detección del sonido no aportan una resolución alta para determinar un ángulo preciso, aunque tampoco era un requerimiento estrictamente necesario, ya que la manera de reflejar esta información es mediante la vibración y las pequeñas variaciones posibles podrían pasar desapercibidas. Más pruebas para determinar una resolución adecuada y significativa para el usuario son necesarias. Además, existen retrasos en el aviso, donde la indicación del ángulo sucede correcto de la fuente sucede hasta la segunda inferencia. Comparado con *(Küçük & Panahi, 2019)*, cuya

resolución es de 20° y con una exactitud de 66.25%, este trabajo tiene una resolución de 36° con una exactitud en las pruebas de 85%. Es necesario mencionar que el trabajo de *(Küçük & Panahi, 2019)* es específico sobre DOA, mientras que para este proyecto era una característica complementaria de una meta sobre clasificación de sonidos del ambiente.

El tratamiento de los audios para la creación de la base de datos, puede ser una alternativa aplicable a otras investigaciones que dependan de pocos audios para realizar entrenamientos, o aquellos sonidos relativamente difíciles de conseguir. También la manera de procesar la captura en tiempo real puede ser un aporte, al tener la flexibilidad de decidir cuándo iniciar con el proceso de datos válidos para la inferencia de la red.

La aplicación del sistema embebido en audífonos también puede ser una alternativa relevante a las comunes encontradas, que usualmente recaen en los relojes inteligentes y aplicaciones de celular, siendo nulas las encontradas en esta forma. La posibilidad de DOA desde esta perspectiva de captura estéreo podría ser aplicada también a otras formas como collares o cinturones. Una serie de pruebas formales en campo y con variedad de usuarios en múltiples contextos son necesarias para evaluar la factibilidad y alternativas del producto.

Conclusiones

Se puede decir en términos generales que los resultados obtenidos son satisfactorios teniendo como parámetros los objetivos iniciales del proyecto. Este desarrollo de poco más de un año logró aportes que pueden ser funcionales para futuras líneas de aplicación. Entre los logros se encuentran que la tasa de exactitud propuesta fue superada por **2%**, se aplicaron métodos de aprendizaje profundo en sistemas embebidos de pocas capacidades (tendencia que seguramente se seguirá desarrollando en años posteriores), se realizó un algoritmo de ángulo de llegada, así como el diseño metodológico de un SE desde la concepción, pasando por pruebas en tarjetas de desarrollo y posteriormente la manufactura y ensamblado de un PCB para el prototipo. Sin embargo, una serie de pruebas de campo y aceptación

de usuario más extensas son necesarias para poder tener un producto completamente listo para salida a mercado y valorar las alternativas posibles para su aplicación. También es necesario hacer un análisis más exhaustivo de consumo de energía para tener parámetros confiables de tiempo de batería, algo relevante al tratarse de un sistema portable. Otro punto relevante es que la comparación con otros métodos de aprendizaje automático (como las redes neuronales recursivas) no logró llevarse a cabo, siendo un área de oportunidad para el proyecto.

Líneas de trabajo futuro podrían ser la posibilidad de personalización por parte del usuario de los sonidos a detectar o incluso de aportar nuevas clases de interés (algo abordado en otras investigaciones); la creación de un conjunto de datos de entrenamiento, pero con información capturada directamente del sistema embebido, que considere las peculiaridades de calidad y tipo de micrófono, así como variaciones acústicas natas de ellos. Dentro de esto, también se podría considerar crear el conjunto de datos desde las características de captura estéreo (no grabación desde un solo micrófono), reconstruyendo la “imagen” sonora a partir de ambos micrófonos separados por cierta distancia, y con ella observar si es posible una mejor clasificación y además encontrar información de DOA directamente. Algo similar la nuestra percepción del sonido.

Múltiples líneas de investigación y aplicación pueden ser encontradas a partir de estos resultados, y se espera que lo presentado en este trabajo pueda ser aun en lo más mínimo, un aporte valioso para mejorar la calidad de vida de las personas con discapacidades.

Bibliografía

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., & Levenberg, J. (2015). *TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems*.https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/45166.pdf

Abdullah Küçük, Ganguly, A., Hao, Y., & Panahi, I. (2019). Real-Time Convolutional Neural Network-Based Speech Source Localization on Smartphone. *IEEE Access*, *7*, 169969–169978. https://doi.org/10.1109/access.2019.2955049

Apple. (2023). *Apple Developer Documentation*. Developer.apple.com. https://developer.apple.com/documentation/accelerate/computing_the_mel_spectrum_using_linear_algebra

Blanton, M., Zhou, J., Moro, M. M., Zhang, D., Tsotras, V. J., Halkidi, M., Hainaut, J.-L., Tanin, E., Korn, F. R., Pedersen, T. B., Böhm, C., Plant, C., Zhang, Q., Strauss, M. J., Jensen, C. S., Snodgrass, R., Li, N., Kantarcioglu, M., Sebe, N., & Jaimes, A. (2009). Human-Computer Interaction. *Encyclopedia of Database Systems*, 1327–1331. https://doi.org/10.1007/978-0-387-39940-9_192

Caio, S., Loubach, D. S., & da Cunha, Adilson Marques. (2009). *An estimation model to measure computer systems development based on hardware and software*. 6.C.2-16.C.2-12. https://doi.org/10.1109/DASC.2009.5347441

Cheng, O., Abdulla, W., & Salcic, Z. (2005). *Performance Evaluation of Front-end Processing for Speech Recognition Systems School of Engineering Report No. 621*.

Dhruv Jain, Ngo, H., Pratyush Patel, Goodman, S. M., Findlater, L., & Froehlich, J. (2020). SoundWatch: Exploring smartwatch-based deep learning approaches to support sound awareness for deaf and hard of hearing users. In *The 22nd International ACM SIGACCESS Conference on Computers and Accessibility*.

Dim, C. A., Feitosa, R. M., Mota, M. P., & de Morais, J. M. (2022). Alert systems to hearing-impaired people: a systematic review. *Multimedia Tools and Applications*, *81*(22), 32351–32370. https://doi.org/10.1007/s11042-022-13045-1

Fanzeres, L. A., Vivacqua, A. S., & Biscainho, L. W. P. (2018). Mobile Sound Recognition for the Deaf and Hard of Hearing. *ArXiv:1810.08707 [Cs, Eess]*. https://arxiv.org/abs/1810.08707

Findlater, L., Chinh, B., Jain, D., Froehlich, J., Kushalnagar, R., & Lin, A. C. (2019). *Deaf and hard-of-hearing individuals' preferences for wearable and mobile sound awareness technologies* (pp. 1–13). Association for Computing Machinery. https://doi.org/10.1145/3290605.3300276

Gholami, A., Kim, S., Dong, Z., Yao, Z., Mahoney, M., & Keutzer, K. (2021). *A Survey of Quantization Methods for Efficient Neural Network Inference*.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. Deeplearningbook.org.

Goodman, S. M., Liu, P., Jain, D., McDonnell, E. J., Froehlich, J. E., & Findlater, L. (2021). Toward User-Driven Sound Recognizer Personalization with People Who Are d/Deaf or Hard of Hearing. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, *5*(2), 1–23. https://doi.org/10.1145/3463501

Goodman, S., Kirchner, S., Guttman, R., Jain, D., Froehlich, J., & Findlater, L. (2020). *Evaluating smartwatch-based sound feedback for deaf and hard-of-hearing users*

across contexts. 1–13. https://doi.org/10.1145/3313831.3376406

Huitzil Muñoz, E. E. (2011). Háptica, una nueva propuesta. *Oral*, *12*(36), 661.

INEGI. (2020). *Población. Discapacidad*. Cuentame.inegi.org.mx. https://cuentame.inegi.org.mx/poblacion/discapacidad.aspx

Iván, J., Pablo, P., Alfredo De Jesús Gutiérrez Gómez, L., Beatriz, M., De La Cruz, E., & Teórico, M. (2015). *Propuesta de una aplicación para dispositivos móviles permitiendo una inclusión social de personas con problemas de sordera e hipoacusia*. https://static1.squarespace.com/static/55564587e4b0d1d3fb1eda6b/t/601b11a975160809972b7efe/1612386745128/Ciencias+y+Sustentabilidad+Tomo+13+CICS+Tuxpan+2015.pdf

Jain, D., Findlater, L., Gilkeson, J., Holland, B., Duraiswami, R., Zotkin, D., Vogler, C., & Froehlich, J. E. (2015). *Head-mounted display visualizations to support sound awareness for the deaf and hard of hearing*. 241–250. https://doi.org/10.1145/2702123.2702393

Jain, D., Huynh Anh Nguyen, K., M. Goodman, S., Grossman-Kahn, R., Ngo, H., Kusupati, A., Du, R., Olwal, A., Findlater, L., & E. Froehlich, J. (2022). ProtoSound: A Personalized and Scalable Sound Recognition System for Deaf and Hard-of-Hearing Users. *CHI Conference on Human Factors in Computing Systems*. https://doi.org/10.1145/3491102.3502020

Jain, D., Lin, A., Guttman, R., Amalachandran, M., Zeng, A., Findlater, L., & Froehlich, J. (2019). Exploring Sound Awareness in the Home for People who are Deaf or Hard of Hearing. *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. https://doi.org/10.1145/3290605.3300324

Küçük, A., & Panahi, I. M. S. (2019). Direction of arrival estimation using deep neural network for hearing aid applications using smartphone. *Proceedings of Meetings on Acoustics*. https://doi.org/10.1121/2.0001256

Manchaiah, V., Taylor, B., Dockens, A., Tran, N., Lane, K., Castle, M., & Grover, V. (2017). Applications of direct-to-consumer hearing devices for adults with hearing loss: a review. *Clinical Interventions in Aging*, *Volume 12*, 859–871. https://doi.org/10.2147/cia.s135390

National Library of Medicine. (2022). *Hipoacusia: MedlinePlus enciclopedia médica*. Medlineplus.gov. https://medlineplus.gov/spanish/ency/article/003044.htm

Nielsen, M. A. (2018). *Neural Networks and Deep Learning*. Neuralnetworksanddeeplearning.com; Determination Press. http://neuralnetworksanddeeplearning.com/index.html

Papalia, D. E., Sally Wendkos Olds, & Ruth Dustin Feldman. (2009). *Psicología del desarrollo*. Mcgraw-Hill.

Piczak, K. (2015). *ESC: Dataset for Environmental Sound Classification*. http://karol.piczak.com/papers/Piczak2015-ESC-Dataset.pdf

Project Management Institute. (2017). *Guide to the Project Management Body of Knowledge (PMBOK® Guide) (6th Edition)* (6th ed.).

Sharma, G., Umapathy, K., & Krishnan, S. (2020). Trends in audio signal feature extraction methods. Applied Acoustics, 158, 107020. https://doi.org/10.1016/j.apacoust.2019.107020

STMicroelectronics. (2017). *B-L475E-IOT01A - STMicroelectronics*. STMicroelectronics. https://www.st.com/en/evaluation-tools/b-l475e-iot01a.html

Subramani, D. K., & Araújo, S. (2022, June 15). *Demystifying machine learning at the*

edge through real use cases | AWS Machine Learning Blog. Aws.amazon.com. https://aws.amazon.com/es/blogs/machine-learning/demystifying-machine-learning-at-the-edge-through-real-use-cases/#:~:text=Machine%20learning%20at%20the%20edge%20(ML%40Edge)%20is%20a

TensorFlow. (2023, January 14). *Simple audio recognition: Recognizing keywords | TensorFlow Core*. TensorFlow. https://www.tensorflow.org/tutorials/audio/simple_audio?hl=en

Tokgoz, S., Kovalyov, A., & Panahi, I. (2020). Real-Time Estimation of Direction of Arrival of Speech Source using Three Microphones. *2020 IEEE Workshop on Signal Processing Systems (SiPS)*. https://doi.org/10.1109/sips50750.2020.9195217

Valenti, M., Diment, A., Parascandolo, G., Squartini, S., & Virtanen, T. (2016). *DCASE 2016 ACOUSTIC SCENE CLASSIFICATION USING CONVOLUTIONAL NEURAL NETWORKS*. https://dcase.community/documents/challenge2016/technical_reports/DCASE2016_Valenti_1011.pdf

Yeung, P., Alonzo, O., & Huenerfauth, M. (2020). Interest and Requirements for Sound-Awareness Technologies Among Deaf and Hard-of-Hearing Users of Assistive Listening Devices. *Universal Access in Human-Computer Interaction. Applications and Practice*, 147–158. https://doi.org/10.1007/978-3-030-49108-6_11

Zavala-Vargas, G., & García, H. (2018). Hipoacusia neonatal. La magnitud de un problema que aún no es escuchado. *Revista Mexicana de Pediatría*, *85*(4), 117–118.

Zhang, Z., & Li, J. (2023). A Review of Artificial Intelligence in Embedded Systems. *Micromachines*, *14*(5), 897. https://doi.org/10.3390/mi14050897