



ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

Ingeniería Informática – Ingeniería de Computadores

Sistema de reconocimiento de problemas cardíacos basado en clasificación de sonidos mediante deep learning

**Realizado por
José Andrés González**

**Dirigido por
Juan Pedro Domínguez Morales**

**Cotutora
Lourdes Durán Lopez**

**Departamento
Arquitectura y Tecnología de Computadores**

Sevilla, Diciembre 2022

Agradecimientos

Primeramente, me gustaría agradecer a todo el profesorado y el personal de la universidad, que han hecho que adquiriera todos los conocimientos y habilidades para poder realizar el trabajo que está presente.

Especialmente me gustaría agradecer a Juan Pedro Domínguez, tutor de este TFG. Por la gran ayuda que me ha ofrecido para poder llevar lo mejor posible la elaboración de este.

Debo de agradecer a mis padres y familia por todo lo que me han dado para ser la persona de ahora, capaz de afrontar todos los retos que han surgido durante el camino.

Quisiera agradecer a todas las personas que han formado durante todos estos últimos años durante el recorrido de mis estudios, hasta llegar justo al momento de ahora. Se han convertido en una parte indispensable de mi vida y han hecho posible que pueda llegar hasta aquí. Gracias por escucharme, apoyarme en los momentos más complicados y compartir con ellos los mejores momentos por mi paso por la universidad.

Resumen

Con la realización de este Trabajo de Fin de Grado se ha pretendido crear un sistema capaz de predecir si una persona sufre de alguna enfermedad cardíaca.

Las enfermedades relacionadas con el corazón son de las que producen un mayor número de muertes al año, además de haber un porcentaje relativamente alto de la población en padecerlo. En muchas ocasiones resulta complicado detectar la enfermedad y esto puede llevar a problemas graves si no es detectada a tiempo.

Con motivación de esto, ha nacido el objetivo de crear un sistema que permita facilitar dicha tarea. La detección de esta enfermedad, como primer paso realizado por el médico, es escuchar los latidos del corazón. Ya que, cuando se sufre de alguna patología, los sonidos que son producidos por los latidos tienden a ser diferente, dependiendo del tipo de enfermedad variará en cuanto a las vibraciones de sonido que produce, como la duración o los momentos donde se produzcan esa especie de ruido.

La idea es crear un sistema que, a partir de la escucha de una muestra de audio de los latidos del corazón, sea capaz de identificar si la persona sufre alguna enfermedad.

Para ello, el sistema va a consistir en crear un modelo clasificador utilizando redes neuronales. Este modelo va a ser construido a partir de un proceso de aprendizaje empleando un conjunto de datos con muestras de audio de latidos que han sido previamente etiquetados como corazones sanos o que sufren de alguna patología.

Con la creación de este prototipo se pretende que pueda servir de gran ayuda al personal sanitario, al facilitar la tarea de detectar la enfermedad y que se pueda reducir lo máximo posibles predicciones erróneas, especialmente los casos de falsos negativos.

Abstracts

The aim of this Final Degree Project is to create a system capable of predicting whether a person suffers from heart disease.

Heart-related diseases are among those that cause the highest number of deaths per year, in addition to the fact that a relatively high percentage of the population suffers from them. It is often difficult to detect the disease and this can lead to serious problems if it is not detected in time.

With this in mind, the objective of creating a system to facilitate this task was born. The detection of this disease, as a first step performed by the doctor, is to listen to the heartbeat. Since, when suffering from any pathology, the sounds that are produced by the heartbeat tend to be different, depending on the type of disease will vary in terms of sound vibrations produced, such as the duration or the moments where this kind of noise is produced.

The idea is to create a system that, by listening to an audio sample of the heartbeat, will be able to identify whether the person is suffering from a disease.

To do this, the system will consist of creating a classifier model using neural networks. This model will be built from a learning process using a dataset with audio samples of heartbeats that have been previously labelled as healthy hearts or suffering from a pathology.

With the creation of this prototype, it is intended that it can be of great help to health personnel, facilitating the task of detecting the disease and reducing erroneous predictions as much as possible, especially cases of false negatives.

Índice

Agradecimientos.....	1
Resumen.....	2
Abstracts.....	3
Índice.....	4
Índice de figuras.....	6
Índice de tablas.....	7
Bloque 1: Descripción del proyecto.....	8
1 Introducción y Motivación.....	8
2 Objetivos del proyecto	12
2.1 Objetivos a nivel profesional.....	12
2.1.1 Objetivos a interés del aporte del producto a la sociedad.....	12
2.1.2 Objetivos de interés propio a querer conseguir con el producto.....	13
2.2 Objetivos a nivel educativo	13
3 Estado del arte	14
4 Elicitación de requisitos y análisis de riesgos	17
4.1 Requisitos.....	17
4.1.1 Requisitos técnicos	17
4.1.2 Requisitos funcionales	17
4.1.3 Requisitos de Interfaz	18
4.1.4 Requisitos de calidad	18
4.1.5 Requisitos de evolución	18
4.1.6 Requisitos de proyecto.....	18
4.1.7 Requisitos de soporte	18
4.2 Análisis de riesgos.....	19
4.2.1 Tabla de riegos.....	20
4.2.2 Matriz de riesgo.....	21
Bloque 2: Ejecución del proyecto.....	22
5 Diseño del sistema	22
6 Implementación.....	24
6.1 Tecnologías empleadas.....	24
6.1.1 Python	24
6.1.2 TensorFlow	25
6.1.3 Keras.....	26
6.1.4 Entornos de trabajo empleados para el desarrollo	27
6.1.4.1 Google Colab	27
6.1.4.2 Visual Studio Code:.....	28

6.1.4.3	Notepad++ con Anaconda Prompt:	28
6.1.5	GitHub.....	29
6.2	Desarrollo.....	30
6.2.1	Procesamiento de los datos	30
6.2.1.1	No segmentar	31
6.2.1.2	Segmentar en función de los latidos del corazón	32
6.2.1.3	Segmentación del audio por una duración de tiempo fija dada	32
6.2.2	Procesamiento del audio:.....	33
6.2.2.1	Transformada rápida de Fourier (FFT).....	34
6.2.2.2	Espectrograma de Mel	34
6.2.2.3	MFCCs (Mel Frequency Cepstral Coefficients)	35
6.2.3	Modelo de entrenamiento (Red Neuronal).....	37
6.2.3.1	Redes Neuronales	37
6.2.3.2	Métricas de evaluación	41
6.2.3.3	Matriz de Confusión	43
	(Solutions 2016)	43
6.2.4	Parámetros e hiperparámetros de la implementación de la red	43
6.2.5	Análisis de los resultado y comparación de modelos	44
6.2.6	Demostración de las metodologías empleadas.....	46
6.2.7	Modelo definido usando Keras.....	47
6.2.8	Modelo VGG16	49
6.2.9	Modelo Xception	52
6.2.10	Modelo RestNet.....	54
6.2.11	Modelo final	56
	Bloque 3: Planificación del proyecto.....	57
8	Planificación Inicial.....	57
8.1	División de la planificación.....	57
8.2	Diagrama de Gantt	60
9	Planificación temporal final	61
10	Planificación de costes iniciales	61
10	Planificación de costes final	63
11	Estudio de mercado.....	64
	Trabajo Futuro.....	2
	Conclusiones.....	2
	Bibliografía.....	2
	Anexos.....	2

Índice de figuras

ILUSTRACIÓN 1. GRÁFICO DEL PORCENTAJE DE LA GENTE DIVIDIDO POR EDADES QUE PADECEN DE UNA ENFERMEDAD CARDIACA	10
ILUSTRACIÓN 2. FONENDOSCOPIO ACÚSTICO	15
ILUSTRACIÓN 3. FONENDOSCOPIO ELECTRÓNICO O DIGITAL	15
ILUSTRACIÓN 4. ECGARDIOGRAMA 1	16
ILUSTRACIÓN 5. ECGARDIOGRAMA 2	16
ILUSTRACIÓN 6. MATRIZ DE RIESGOS	21
ILUSTRACIÓN 7. DIAGRAMA DEL DISEÑO DEL SISTEMA.....	22
ILUSTRACIÓN 8. ESQUEMA DEL FUNCIONAMIENTO DE PYTHON	24
ILUSTRACIÓN 9. ESQUEMA DE ABSTRACCIÓN DE CREACIÓN DE UNA RED NEURONAL.....	26
ILUSTRACIÓN 10. ORGANIZACIÓN DE LOS DATOS.....	30
ILUSTRACIÓN 11. MODOS DE SEGMENTACIÓN POR TIEMPO FIJO.....	33
ILUSTRACIÓN 12. ECUACIONES DEL ESPECTROGRAMA DE MEL	34
ILUSTRACIÓN 13. GRÁFICO DE LA CURVA ENTRE LA RELACIÓN DE LA ESCALA DE MEL Y LA FRECUENCIA.....	35
ILUSTRACIÓN 14. IMAGEN ORIGINADA TRAS APLICAR MEL A UNA MUESTRA DE AUDIO	35
ILUSTRACIÓN 15. DIAGRAMA DEL PROCESO DE APLICAR MFCC	36
ILUSTRACIÓN 16. IMAGEN RESULTANTE TRAS APLICAR MFCC A UN AUDIO	36
ILUSTRACIÓN 17. SEÑAL DE AUDIO, MEL Y MFCC.....	37
ILUSTRACIÓN 18. ASPECTO DE UNA RED NEURONAL	37
ILUSTRACIÓN 19. SUMA DE LAS PONDERACIONES DE UNA NEURONA	38
ILUSTRACIÓN 20. FUNCIONES DE ACTIVACIÓN	38
ILUSTRACIÓN 21. CÁLCULO DEL ERROR CUADRÁTICO	39
ILUSTRACIÓN 22. GRÁFICO DE LA EVOLUCIÓN DEL ERROR DURANTE EL ENTRENAMIENTO DE LA RED	40
ILUSTRACIÓN 23. GRÁFICOS DE LA EVOLUCIÓN DEL ERROR DEL MODELO 1	47
ILUSTRACIÓN 24. GRÁFICO DE LA EVOLUCIÓN DE LA PRECISIÓN DEL MODELO 1	47
ILUSTRACIÓN 25. MATRIZ DE CONFUSIÓN DEL MODELO 1.....	48
ILUSTRACIÓN 26. GRÁFICO DE LA EVOLUCIÓN DEL ERROR DEL MODELO 2	49
ILUSTRACIÓN 27. GRÁFICO DE LA EVOLUCIÓN DE LA PRECISIÓN DEL MODELO 2	50
ILUSTRACIÓN 28. MATRIZ DE CONFUSIÓN DEL MODELO 2.....	51
ILUSTRACIÓN 29. GRÁFICO DE LA EVOLUCIÓN DEL ERROR DEL MODELO 3	52
ILUSTRACIÓN 30. GRÁFICO DE LA EVOLUCIÓN DE LA PRECISIÓN DEL MODELO 3	52
ILUSTRACIÓN 31. MATRIZ DE CONFUSIÓN DEL MODELO 3.....	53
ILUSTRACIÓN 32. GRÁFICO DE LA EVOLUCIÓN DE LA PRECISIÓN DEL MODELO 4	54
ILUSTRACIÓN 33. GRÁFICO DE LA EVOLUCIÓN DEL ERROR DEL MODELO 4	54
ILUSTRACIÓN 34. MATRIZ DE CONFUSIÓN DEL MODELO 4.....	55

Índice de tablas

TABLA 1. VALORES DE LA PROBABILIDAD DE UN INCIDENTE.....	19
TABLA 2. VALORES DEL IMPACTO DE UNA INCIDENCIA.....	19
TABLA 3. RIESGOS POSIBLES QUE OCURRAN DURANTE EL DESARROLLO.....	20
TABLA 4. REPRESENTACIÓN DE CADA RIESGO EN LA MATRIZ.....	21
TABLA 5. ETIQUETADO DE LOS DATOS.....	30
TABLA 6. NUEVO ETIQUETADO TRAS EL PROCESADO.....	31
TABLA 7. ESTRUCTURA DE UNA MATRIZ DE CONFUSIÓN.....	43
TABLA 8. VALORES OBTENIDOS PARA CADA UNA DE LAS METODOLOGÍAS.....	46
TABLA 9. TIEMPOS OBTENIDOS EN EL ENTRENAMIENTO Y COMO PREDICTOR DEL MODELO.....	46
TABLA 10. MÉTRICAS OBTENIDAS EN EL MODELO 1.....	48
TABLA 11. VALORES DE LOS TIEMPOS DEL MODELO 1.....	49
TABLA 12. MÉTRICAS OBTENIDAS DEL MODELO 2.....	50
TABLA 13. VALORES DE LOS TIEMPOS PARA EL MODELO 2.....	51
TABLA 14. MÉTRICAS OBTENIDAS DEL MODELO 3.....	53
TABLA 15. VALORES DE LOS TIEMPOS DEL MODELO 3.....	53
TABLA 16. MÉTRICAS OBTENIDAS DEL MODELO 4.....	55
TABLA 17. VALORES DE LOS TIEMPOS DEL MODELO 4.....	55
TABLA 18. DISTRIBUCIÓN DE LOS COSTES INICIALES.....	61
TABLA 19. GASTOS EN MATERIALES INICIALES.....	62
TABLA 20. PRECIO MEDIO POR PROFESIÓN.....	63

1 INTRODUCCIÓN Y MOTIVACIÓN

En este proyecto se pretende hacer un estudio y evaluación de los resultados obtenidos por medio de distintos algoritmos de inteligencia artificial y diversos modos de procesados sobre señales de audio, para la creación de un sistema capaz de determinar enfermedades cardiovasculares.

La ciencia y la tecnología han avanzado a pasos agigantados en los últimos años consiguiendo resolver problemas que hace años serían impensables. Uno de los campos que lleva a realizar mayores investigaciones y estudios es la medicina. Debido a la gran complejidad que conlleva el entendimiento del cuerpo humano, la mente, las enfermedades, etc. Actualmente aún se desconoce el funcionamiento del cerebro y conceptos como los sentimientos, las emociones y los pensamientos quedan alejados de nuestro entendimiento.

Entre uno de los grandes desafíos que se encuentra en el panorama actual es la lucha por las enfermedades. Éstas van surgiendo nuevas, evolucionando o mutando de las que conocemos, haciendo que su tratamiento sea ineficaz o algunas que son muy complicadas de combatir como es el caso del cáncer. Del cual aún no se ha logrado descubrir una cura capaz de lograr eliminarlo con facilidad.

La ingeniería ha producido muchísimas mejoras en este sector con la creación de máquinas que ofrecen de una gran ayuda al personal sanitario como son una máquina de ECG, unidades electro quirúrgicas, desfibriladores o monitores de pacientes. En un hospital podemos encontrarnos todas ellas y muchas más por todos lados, convirtiéndose indispensables en el día a día de los médicos.

Desde estos últimos años con la era de la digitalización, la informática ha pasado a tener también un papel importante en esta disciplina. Tanto es así que actualmente existen estudios específicos de las aplicaciones que tiene la informática en el mundo sanitario como pueden ser bioinformática, informática clínica...

Ésta es empleada para multitud de cosas, desde la agilización de todo el sistema administrativo, el desarrollo de maquinarias como las mencionadas anteriormente o usarse como apoyo para la investigación de los médicos. Hay muchos grupos de investigación en los que se aprovecha las capacidades computacionales para hacer experimentos y obtener resultados a conclusiones a partir de grandes cantidades de datos. Avances en la inteligencia artificial, el big data, las telecomunicaciones, han hecho posible que se puedan llevar estudios de muchas índoles donde intentar descubrir curas a enfermedades actuales o la predicción de padecer alguna patología a alguna enfermedad específica entre tantas otras propuestas que se están llevando a cabo.

La inteligencia artificial es uno de los campos especializados de la informática que más ha crecido en los últimos años y aún está en pleno desarrollo. Esta ha sido muy impulsada por matemáticos e informáticos que se han dedicado a llevar a cabo muchos estudios para mejorar las técnicas empleadas y muy fomentado tanto por el mundo científico como las organizaciones empresariales. Esto es debido a que se ha descubierto a la cantidad de aplicaciones en la que se puede hacer uso de ella y resolver multitud de problemas.

También el avance tecnológico de estos últimos años y las capacidades computacionales que tienen los supercomputadores actuales han hecho posible que pueda crecer los usos de esta de una manera exponencial.

Entre tantos de estos usos es la investigación en el mundo de la salud, donde se intenta poder resolver estos problemas que como comentamos anteriormente son de una gran complejidad.

En el mundo existen muchos tipos de enfermedades como son las infecciosas o parasitarias, tumores malignos, endocrinas, mentales, del sistema inmunológico, etc. Unas de las más relevantes son las cardiovasculares debido al índice elevado de casos existentes, las cuales por ello hay que tener muy en cuenta y es en estas a las que nos vamos a centrar a lo largo del proyecto.

Estas enfermedades las hay de varios tipos y pueden ser producidas por malos hábitos en nuestra vida como el sedentarismo, la alimentación, tabaquismo, etc. También algunas son hereditarias por lo que una de las prioridades es conocer si algún pariente del paciente ha padecido de alguna de estas enfermedades.

Entre las más comunes son insuficiencia cardíaca, hipertensión arterial, angina de pecho, infarto de miocardio...

Según la ONS, se estima que 17,9 millones de personas murieron por enfermedades cardiovasculares en 2019, lo que representa un 32 % de todas las muertes del planeta. Se estima que para 2030 casi 23.6 millones de personas pueden morir por una de estas afecciones. (Anón s. f.-c) Este tipo de enfermedades supone la causa que cobra más vidas en todo el planeta.

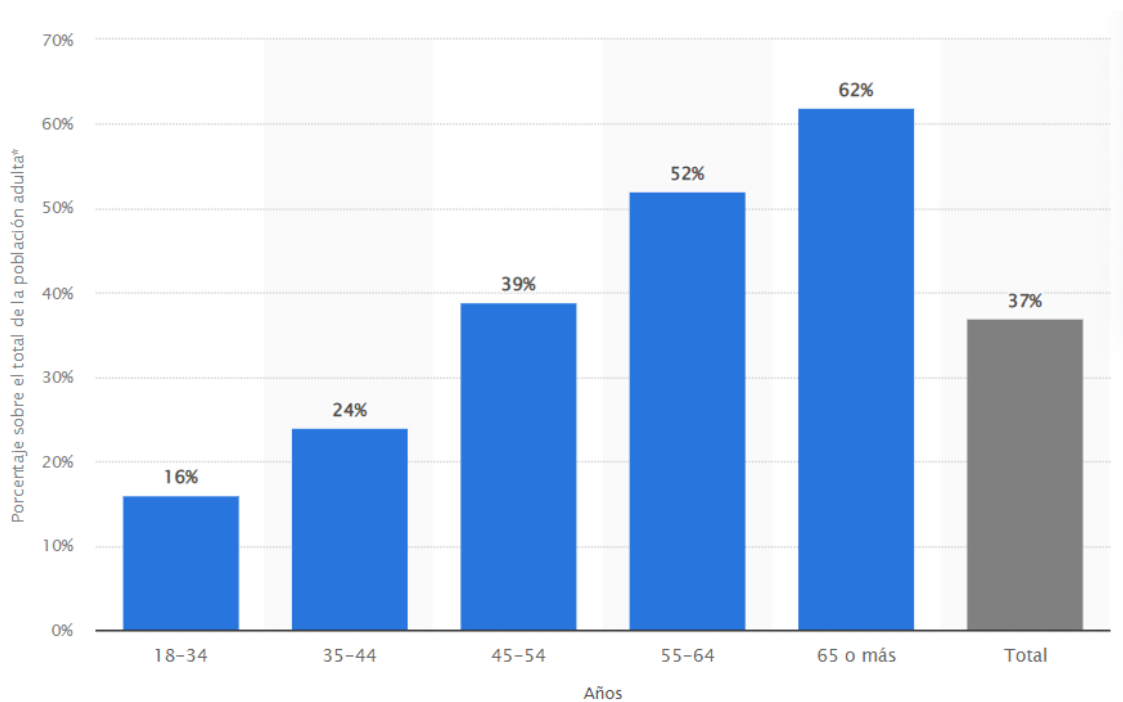


Ilustración 1. Gráfico del porcentaje de la gente dividido por edades que padecen de una enfermedad cardiaca

(Anón s. f.-c)

Uno de los incentivos para la realización de este viene dado debido a la ayuda que puede ofrecer a los médicos a la hora de poder detectar alguna patología cardiovascular. Los médicos para llevar a cabo su labor de comprobar el correcto funcionamiento del corazón usan aparatos como el estetoscopio para poder escuchar los latidos. Con esto son capaces de detectar si se produce algún ruido que está fuera de lo normal como puede ser un soplo o cualquier otra patología.

Para poder llevar a cabo dicha función es necesario que pasen por un largo periodo de entrenamiento. Se someten a escuchar una extensa lista de audios de latidos de corazones, los cuales están divididos entre corazones que se encuentran sanos sin ningún problema o en cambio sufren alguna patología. El objetivo al final de esto es que sus oídos se acostumbren a los sonidos que producen los latidos para que sean capaces de determinar cuando están escuchando a un paciente si su funcionamiento es el correcto.

Como es normal el oído no es perfecto y aunque haya sido pasado por un periodo de entrenamiento puede dar a fallos ya que no todos percibimos del mismo modo los sonidos. Además, como se es bien sabido, estos con el tiempo van perdiendo capacidades y puede hacer mucho más complicada la tarea por parte del profesional sanitario cuando alcanzan una cierta edad. (Fonendo 2021)

Estos fallos pueden ser drásticos en muchas situaciones ya que un error al detectar un falso caso positivo supondría de muchos gastos en recursos tanto materiales, de personal y económicos para llevar a cabo un estudio completo y preciso. Un error en el caso contrario para un falso negativo supondría poner a un gran riesgo al paciente, ya que no se estaría tratando de modo alguno de la enfermedad que está padeciendo.

Como se puede observar en el gráfico anterior, la probabilidad de padecerlas aumenta progresivamente con el paso de los años, siendo con 65 años la edad con más frecuencia en ser detectados.

Por lo que a ciertas edades es fundamental llevar a cabo un control más riguroso para ser capaces de predecir alguna patología. La identificación de éstas, muchas veces no se realiza como debería varios motivos. Estos pueden ser, como contábamos antes, debido al fallo que puede tener un médico a la hora de la audición del corazón, la falta de tiempo por parte del personal sanitario debido a la saturación que supone realizar este proceso a la gran cantidad de pacientes o incluso a veces la falta de medios entre otros. Estos factores pueden suponer a que sean identificadas más tarde y por lo tanto una mayor dificultad para su tratamiento.

A partir de todos estos motivos es por lo que se da la implementación del sistema con la idea de poder facilitar la tarea, tanto para agilizar el proceso, como para servir de apoyo a los médicos y poder usarlo como herramienta que sirva de comprobación y contraste en base a su resultado obtenido de una posible enfermedad. Además de intentar incluso que con dicho sistema se pueda reducir considerablemente el porcentaje de falsos positivos y especialmente falsos negativos, consiguiendo evitar casos de defunciones por estos motivos. Además de agilizar los tiempos de espera por parte de los pacientes al poder conseguir los resultados en poco tiempo y en simultáneo.

2 OBJETIVOS DEL PROYECTO

El objetivo principal va a ser el diseñar un prototipo de un sistema clasificador para determinar enfermedades del sistema cardiovascular de un paciente a través de los sonidos del corazón.

Para poder llevar a cabo dicha meta, se ha realizado una subdivisión de pequeños objetivos agrupados en dos categorías:

2.1 OBJETIVOS A NIVEL PROFESIONAL

2.1.1 Objetivos a interés del aporte del producto a la sociedad.

- **Dar apoyo y soporte a los médicos.** Conseguir que los médicos a través de esta herramienta puedan comprobar a modo de verificación que su diagnóstico es correcto. O en caso contrario optar a la realización de más pruebas.
- **Agilizar el proceso de determinación de alguna enfermedad.** Aprovechar las capacidades innovadoras que ofrece este sistema para poder analizar a varios pacientes al mismo tiempo con el mismo gasto de personal sanitario. Consiguiendo un primer resultado en poco tiempo.
- **Reducir los costes en recursos hospitalarios.** Evitar tener que realizar un mayor número de pruebas para poder determinar la posibilidad de padecer alguna enfermedad. Disminuyendo tanto tiempo de esperas por parte de los pacientes, los costes en personal y en el uso de maquinaria hospitalaria para los diagnósticos.
- **Aminorar la posibilidad de un diagnóstico erróneo.** Prevenir la posibilidad de dar un diagnóstico incorrecto tanto dando un posible falso positivo que conllevaría a mayores gastos en recursos sanitarios como un falso negativo lo que podría llevar a un tratamiento tardío de la enfermedad. Pudiendo poner en un grave riesgo al paciente

2.1.2 Objetivos de interés propio a querer conseguir con el producto.

- **Aportar un producto novedoso.** Llevar a cabo el desarrollo de un producto que aporte innovación al entorno de los productos actualmente disponible en el mercado.
- **Hacer llegar el producto a la mayor cantidad de clientes potenciales posibles.** Crear un producto funcional dando el servicio prometido con la mayor fiabilidad con el precio más competitivo posible.
- **Soporte a necesidad actual.** Las enfermedades del corazón según estudios van en aumento con el paso de los años, por lo que resulta de interés social que se creen medios con los que mejorar la situación.

2.2 OBJETIVOS A NIVEL EDUCATIVO

- **Manejo de redes neuronales.** Conocer algunos detalles de la librería de TensorFlow y usarla para la implementación de distintas redes neuronales convolucionales.
- **Usar varios modos de procesamiento de señales.** Aplicar varios algoritmos de procesamiento de audio para obtener unos nuevos que sean válidos para nuestro sistema de entrenamiento.
- **Bioinformática.** Conocer detalles y aspectos destacables de como aplicar la informática para resolver problemáticas con el ámbito sanitario.
- **Estudio de modelos.** Tener los conocimientos para comprender las características y el funcionamiento de los modelos aplicados para una red neuronal y poder crear uno propio en caso de necesidad. Así como valorar cual es más conveniente a usar para la solución que se pretende proponer.
- **Examinar validez de los datos.** Se va a hacer uso de una gran cantidad de datos para entrenar el modelo del sistema. Hacer una comprobación previa de los datos a usar, analizándolos para dar una valoración de las cuales son los más idóneos para el entrenamiento.
- **Análisis de los resultados.** Tras los resultados obtenidos por cada uno de los modelos usados en cada una de las pruebas, saber hacer una comparación exhaustiva entre ellos y evaluar cual representa una mejor solución.

3 ESTADO DEL ARTE

En el panorama actual existen en el mercado productos con el fin de poder determinar enfermedades cardiovasculares a través del sonido producido por los latidos del corazón como se propone en este proyecto.

Estetoscopio o Fonendoscopio

El primer paso que realizan los médicos cuando llega un paciente para realizar una revisión clínica general es usar un estetoscopio. Es un dispositivo acústico que amplifica los ruidos corporales para lograr su mejor percepción. Son empleados para auscultar el corazón, los pulmones y abdomen. Con ellos los médicos pueden escuchar ruidos cardíacos, respiratorios, intestinales o soplos debido a flujos anómalos sanguíneos en arterias y venas.

Actualmente en el mercado existen dos tipos, los electrónicos y los acústicos o mecánicos. Dentro de los mecánicos se pueden subdividir a su vez en otros dos tipos, los convencionales que son los empleados para la revisión que hace el médico en las consultas u otros específicos para escuchar los latidos del corazón de un feto. (Anón s. f.-e)

Estetoscopio acústico

Este tipo funciona colocando sobre el paciente una membrana o campana que transmite las señales acústicas a través de unos tubos llenos de aire, llegando hasta los oídos del médico. Usando dicha membrana se detectan las altas frecuencias, permitiendo escuchar el corazón.

Realmente para el funcionamiento del sistema que se intenta proponer en este proyecto, hace uso de este dispositivo como medio para poder captar los sonidos del corazón y realizar su correspondiente clasificación.

Si lo comparamos con nuestro sistema, éste tiene la desventaja, como es de suponer, de un coste mayor económicamente.

En cuanto a las ventajas, como comentábamos anteriormente, usando solo el fonendoscopio, dependemos de la evaluación del resultado por parte del oído del médico para determinar si el paciente padece de alguna enfermedad. El cual este puede llevar a fallo por la imprecisión que puede sufrir por parte del médico, además que resulta ser más subjetivo ya que varía según la persona que realice la auscultación. Por lo que como contrapartida tenemos que no es tan fiable como lo que se intenta proponer con nuestra solución, además otra ventaja a tener en cuenta es que podría emplearse para agilizar en cierta medida el proceso ya que los resultados se obtienen instantáneamente, solo dependiendo del tiempo de escucha de los latidos y un breve tiempo de respuesta por parte del sistema, pero, pudiéndose realizar varios en paralelo. Mejorando así la agilización del sistema sanitario para este tipo de diagnósticos al disminuir el tiempo de espera medio por paciente y reduciendo el gasto en personal para la realización de dicha tarea.

Realmente la idea que se propone con este proyecto es crear un sistema donde se sustituya o al menos sirva como corroboración. Al proceso que hace el médico de escucha del corazón y a partir de esta, deduzca si el paciente sufre de algún síntoma relacionado con alguna enfermedad cardiovascular.

Estetoscopio Electrónico

En cuanto a los electrónicos, aportan algunas ventajas sobre los anteriores como poder amplificar los sonidos, poderlos observar en una pantalla, reproducir y grabar las señales de audio para tener un registro clínico o sirviendo para la enseñanza. Como inconveniente es que presenta una menor ligereza y manejabilidad, un mayor coste de compra, mantenimiento y reparación, precisan de baterías y son más difíciles de desinfectar. El uso de este no está tan extendido como el anterior, solo es más recomendable para situaciones muy específicas como pueden ser por dificultades de audición, para una especialidad médica, en un entorno ruidoso o como herramienta para la educación.

Si hacemos una comparación con nuestro dispositivo, tenemos como principal ventaja que se consigue obtener un primer resultado clínico concreto. Así conociendo en primera instancia y rápido, si el paciente sufre de alguna patología de riesgo con la que tener que hacer un estudio de mayor profundidad e iniciar su correspondiente tratamiento. Como desventaja es que nuestra solución no ofrece tanta información, ni un registro de los datos que manejar posteriormente el médico, pero al obtener el resultado directamente se puede prescindir de estos detalles. En cuanto al aspecto económico ambos sistemas pueden tener un coste similar por lo que no es relevante. (Anón s. f.-h)



*Ilustración 3. Fonendoscopio
Electrónico o digital*



Ilustración 2. Fonendoscopio acústico

Ecocardiograma

Otra herramienta que es empleada para hacer un estudio más preciso de padecer alguna cierta enfermedad es el ecocardiograma. Esta máquina nos da una información muy detallada sobre el funcionamiento del corazón, ya que a partir de ondas sonoras genera una imagen precisa del corazón en el que se puede visualizar el movimiento que produce y observar cómo circula la sangre a través del mismo y por las válvulas cardíacas. Ofreciendo así una gran ayuda para saber si una válvula se ha estrechado o tiene fugas. Haciendo una comparativa con nuestro sistema, tenemos como desventaja con respecto a nuestro prototipo, que se puede conocer con mayor fiabilidad y en mayor medida, cualquier problema o mal funcionamiento del corazón. Ya que aporta información a más niveles y se pueden conocer un mayor conjunto de enfermedades o alguna posible derivación a otra. Como contra es que requiere de mayores gastos tanto a nivel económico, personal, en mayor espera de tiempo para la realización de la prueba, etc. (Anón s. f.-b)



Ilustración 4. Ecocardiograma 1



Ilustración 5. Ecocardiograma 2

Investigaciones usando IA

Si podemos encontrar diversos estudios donde intentan acercarse a crear un sistema muy similar al que vamos a describir a lo largo de toda la memoria. En el que a partir de algún tipo de información ya sea visual o sonora, poder conocer con la mayor fiabilidad posible si el paciente sufre de algún tipo de enfermedad. Pero todos ellos están a nivel de estudios e investigativos en el cual no son usados en la práctica en ningún centro sanitario. Ante estos estudios ya realizados no hay unas ventajas o desventajas sobre nuestra solución. Pero sí proponer una que a partir de varias pruebas y sobre el análisis de los resultados obtenidos, pueda ser de las que consigan tener un gran desempeño y evaluando la posibilidad a que sea llevado a una práctica de uso en un entorno real.

4 ELICITACIÓN DE REQUISITOS Y ANÁLISIS DE RIESGOS

4.1 REQUISITOS

En este apartado se va a tratar de todos los requisitos que deberá de cumplir el sistema a distintos niveles, haciendo un desglose de ellos. Con el fin de que sean satisfechos los objetivos preestablecidos.

4.1.1 Requisitos técnicos

- Se debe de tener un control de la calidad de los datos a usar, para poder realizar un buen sistema de entrenamiento. Además de tener creada una estructuración específica para poder después manipular los datos correctamente. El sistema ha de usar una gran cantidad de datos y tienen que estar etiquetados correctamente. Además de tener que estar divididos en grupos según algún criterio riguroso a usar.
- El sistema para poder realizar un buen entrenamiento ha de recibir los audios con una calidad decente. Para ello se va a usar distintas técnicas de procesamiento de audio para conseguir que la señal sea lo más identificable posible y así que el modelo pueda caracterizar las partes más críticas para poder distinguir las muestras con o sin una patología,
- La solución debe de estar implementada con un modelo fiable con el cual, tras todo el proceso de entrenamiento, haber hecho el test y usar pruebas aleatorias de muestras dadas, consiga obtener un porcentaje de acierto alto. El modelo será escogido tras la comparación entre varias pruebas usando distintas arquitecturas de redes neuronales para la construcción del modelo. Estas redes serán implementadas dando el diseño de su arquitectura o empleando alguna existente ofrecida por algunas de las librerías.

4.1.2 Requisitos funcionales

- El producto debe de ser capaz de poder usarse indefinidamente sin necesidad de tener que reiniciarlo cada vez que vaya a hacerse un nuevo diagnóstico, ni tampoco tener que realizar ninguna configuración.
- El sistema tras recibir un audio con una duración variable debe de ser capaz de dar una respuesta sin ningún tipo de problemática.
- El tiempo de respuesta desde la percepción del audio hasta la obtención de la clasificación debe de ser lo más rápida posible sin repercutir en la validez del resultado obtenido.

4.1.3 Requisitos de Interfaz

- La interfaz debe de ser clara y sencilla para que pueda ser empleada sin la necesidad de tener que dedicar mucho tiempo a conocer el funcionamiento de ella.
- La interfaz debe de permitir que el usuario haga un uso conozca fácilmente el resultado que se ha obtenido del paciente.

4.1.4 Requisitos de calidad

- Debe de tener un sistema que informe al usuario en caso de tener problemas en el sistema por motivos de actualización, configuración, etc.
- Los audios que el sistema reciba por parte del paciente deben de ser con una calidad mínima. En caso de que se detecte que no es así, avisar al personal médico de realizar otra muestra para evitar falsas predicciones.

4.1.5 Requisitos de evolución

- El sistema debe de ser capaz en la mayoría de lo posible capaz de adaptarse a las nuevas exigencias por parte de los usuarios como de las nuevas tecnologías resurgentes. Además de actualizar el modelo periódicamente en caso de encontrar soluciones más eficientes o de que surgieran nuevas patologías y enfermedades cardiovasculares que tener que caracterizar.
- También poderse adaptar a cambios en el centro donde se haga uso del producto. Haciendo que sea lo más íntegro posible a la infraestructura para llevar un uso sencillo de la aplicación.

4.1.6 Requisitos de proyecto

- Tener un alto grado de ajuste con la planificación dada tanto a nivel temporal como financiera. Para que los márgenes de coste y el tiempo de desarrollo sean coherentes.
- Hacer varias pruebas de testeo y usar una métrica fiable para evaluar los resultados y poder ir ajustando el modelo para conseguir mejorar su fiabilidad y aplicarlo como solución a la implementación final.

4.1.7 Requisitos de soporte

- Previamente a ser usado en un entorno real sobre pacientes. El personal sanitario debería de llevar a cabo una serie de pruebas previas como soporte, para familiarizarse con el funcionamiento y entender cómo se muestran los resultados para dar un diagnóstico correcto.

4.2 ANÁLISIS DE RIESGOS

Para realizar un correcto análisis de riesgos es importante distinguir entre varias (subdivisiones) para clasificar los posibles riesgos, indicando la posibilidad de que ocurran y el impacto que supondría en el desarrollo del proyecto.

El nivel de riesgo ha sido diferenciado en dos. El impacto que tendrían en el correcto desarrollo del proyecto y la probabilidad en la que estos sucedan.

En las siguientes tablas se muestran los valores de dichos impactos:

Tabla de Probabilidad	
Valor	Descripción
Baja	Puede no llegar a ocurrir en todo el desarrollo del proyecto
Media	La amenaza puede presentarse muy ocasionalmente durante el ciclo de desarrollo
Alta	El riesgo de que suceda puede darse un día o alguno más por cada mes
Muy Probable	Es un problema muy común que puede llegar a suceder cada semana

Tabla 1. Valores de la probabilidad de un incidente

Tabla de Impacto	
Valor	Descripción
Bajo	El daño presentado no tiene consecuencias relevantes para la organización
Moderado	El daño percibido puede tener consecuencias reseñables para el proyecto
Alto	El daño derivado puede tener consecuencias graves reseñables para el desarrollo
Catastrófico	El daño presentado puede suponer unos de gran magnitud, complicando mucho su recuperación

Tabla 2. Valores del impacto de una incidencia

4.2.1 Tabla de riesgos

En la siguiente tabla se van a mostrar los posibles problemas identificados que pueden suceder a lo largo de todo el desarrollo del proceso. Junto con los valores mencionados en las tablas anteriores. Además, añadiendo una posible solución en caso de que se llegara a suceder.

Etiqueta	Riesgo	Probabilidad de ocurrencia	Impacto	Plan de mitigación
R0	Incumplimiento con los tiempos	Media	Alto	Control riguroso en cuanto al avance de estas con una alta frecuencia
R1	Problemas en el control de versiones	Media	Moderado	Archivar todas las versiones realizadas anteriormente
R2	Fallos en la implementación del código	Alta	Alto	Analizar el comportamiento del código periódicamente
R3	Errores de los datos a emplear para el sistema	Baja	Moderado	Comprobar los datos previamente de su uso y buscarlos de fuentes fiables
R4	Falta organización	Baja	Moderado	Seguir la planificación propuesta inicialmente
R5	Mal uso o importación de las librerías	Alta	Alto	Estudiar el manejo de ellas antes de usarlas y comprobar que es compatible con el proyecto para su importación
R6	Pérdida de los resultados obtenidos	Baja	Catastrófico	Realizar subidas automáticas sobre los avances del proyecto

Tabla 3. Riesgos posibles que ocurran durante el desarrollo

4.2.2 Matriz de riesgo

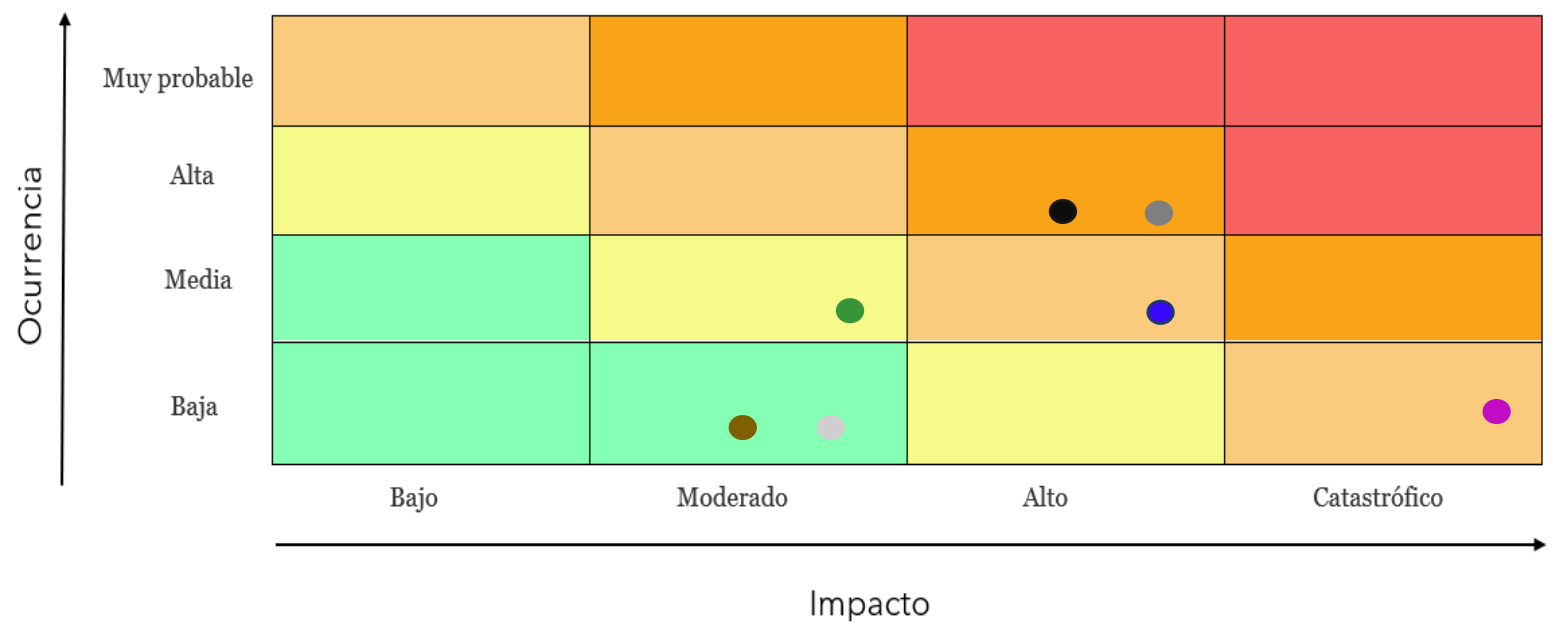


Ilustración 6. Matriz de riesgos








Etiqueta	Riesgo	Representación
R0	Incumplimiento con los tiempos	
R1	Problemas en el control de versiones	
R2	Fallos en la implementación del código	
R3	Errores de los datos a emplear para el sistema	
R4	Falta organización	
R5	Mal uso o importación de las librerías	
R6	Pérdida de los resultados obtenidos	

Tabla 4. Representación de cada riesgo en la matriz

5 DISEÑO DEL SISTEMA

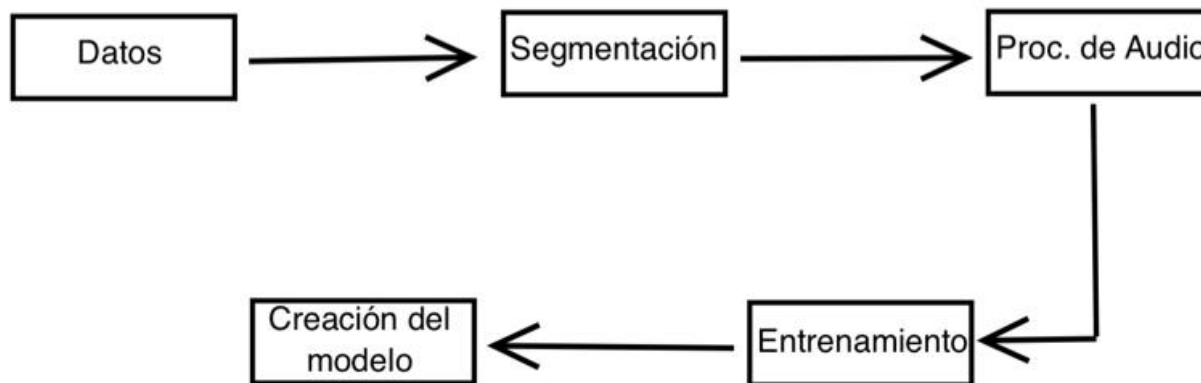


Ilustración 7. Diagrama del diseño del sistema

Datos:

Es el almacén que va a contener todos los datos a emplear por el sistema. Como contenedor para almacenar los datos se va a usar una carpeta y un fichero tipo csv para direccionar cada elemento de la carpeta. La carpeta está estructurada de tal manera que los datos estén divididos en varios conjuntos con las mismas propiedades, usándose uno de ellos como entrenamiento, otro como validación y un último dedicado al test. Cada dato va a consistir en un archivo de audio, en concreto una muestra de los latidos de corazón captados de algún paciente, de una duración variable, pero siempre inferior a 30 segundos y cada audio, va a tener asociada una etiqueta indicando que tipo de patología padece o en caso de ninguna, etiquetándose como normal.

Segmentación:

Se trata en realizar un proceso en el que crear un nuevo almacén de los datos a partir de una subdivisión de los anteriores. Cada muestra de audio va a ser dividido en varios datos nuevos con la misma etiqueta que la del audio completo. La división puede ser variable, una de las opciones posible sería que dividir el audio original en trozos de tres segundos. Los parámetros para la segmentación del audio dependerán de la prueba en concreta a realizar para analizar los resultados que obtiene el modelo con dichos parámetros.

Procesamiento del audio:

Va a realizar sobre todas las muestras de audio una serie de modificaciones para que sean más legible y por lo tanto faciliten el entrenamiento para conseguir un modelo con mayor precisión.

Se van a aplicar varias técnicas de procesamiento. Uno de estos es aplicar la transformada de Fourier como medición de audio y acústica. Con ella el audio es descompuesto en sus componentes espectrales individuales y proporcionándonos información sobre su composición.

Otro interesante a usar es MFCC (Mel Frequency Cepstral Coefficients) son coeficientes para la representación del habla basados en la percepción auditiva humana. Esto permite un procesado de datos más eficiente, por ejemplo, en compresión de audio.

Red Neuronal, Entrenamiento, Construcción del modelo:

Tras tener ya preparados los datos para el entrenamiento, se procede a usar algún modelo de red neuronal que cambiará en función de la prueba. Este puede ser tanto definido en cuanto a su arquitectura o puede ser usado algunos de los ya implementados que ofrece Keras.

Evaluación y clasificador

Una vez ya tenido el clasificador creado tras haber entrenado el modelo. Se evaluarán a partir de los resultados cual es el que ofrece una mayor fiabilidad. Este es el que se usará como solución y se usará como herramienta de clasificación para determinar a partir de un audio de latidos de corazón dado si padece de alguna anomalía.

6 IMPLEMENTACIÓN

6.1 TECNOLOGÍAS EMPLEADAS

6.1.1 Python

Características:

Python es un lenguaje de programación de alto nivel, orientado a objetos, con una semántica dinámica integrada. Es un lenguaje de programación de propósito general e interpretado. Por lo que el código fuente se ejecuta directamente, instrucción a instrucción sin un proceso de compilación, a través de un programa llamado intérprete que lee las instrucciones en tiempo real y las ejecuta. El intérprete comúnmente utilizado es la Máquina Virtual de Python (PVM).

Python es relativamente simple, esto reduce el costo de mantenimiento y de desarrollo del programa. Uno de los beneficios más importantes de Python es que tanto la librería estándar como el intérprete están disponibles gratuitamente y todas las herramientas necesarias están disponibles en todas las plataformas principales. (Anón s. f.-a)

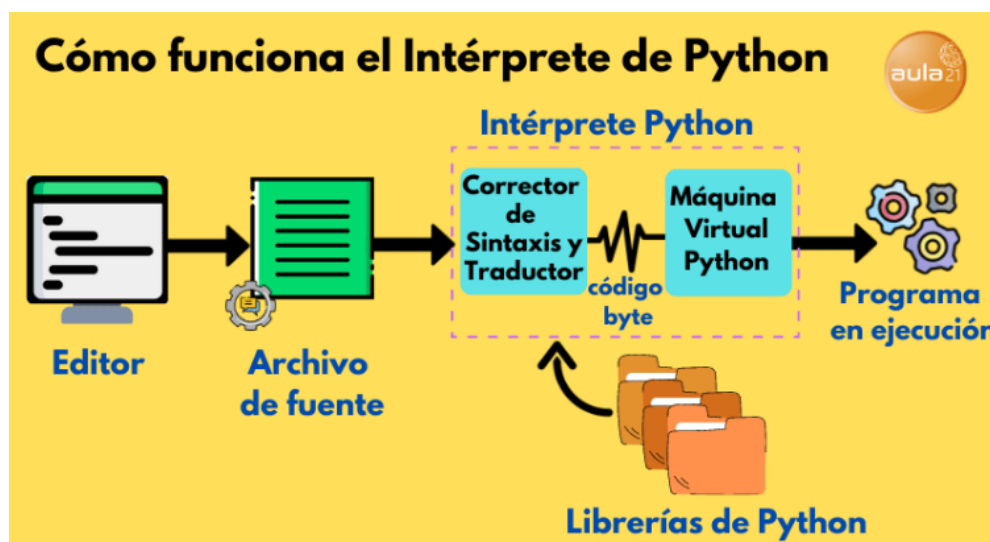


Ilustración 8. Esquema del funcionamiento de Python

Usos:

Python, aunque es de propósito general es principalmente usado debido a sus características para el desarrollo web y de aplicaciones informáticas y el análisis de datos. En este último ámbito Python adquiere una gran relevancia en campos emergentes como la ciencia de datos. Aplicándose como lenguaje más empleado en campos como la inteligencia artificial, machine learning, data mining, etc. También, puede ser usado para

procesar texto, mostrar números o imágenes, resolver ecuaciones científicas y guardar datos.

Es un lenguaje que con el tiempo hay ido ganando relevancia en la industria siendo empleado actualmente por grandes corporaciones como por ejemplo en el motor de búsqueda de Google o YouTube, el sistema de transacciones de la bolsa de Nueva York o la NASA para programar sus equipos y maquinaria espacial. Además, ha conseguido afianzarse en los sistemas académicos dado a facilidad a la hora de aprender y la versatilidad que ofrece.

- **Ventajas**

Es multiplataforma. Gracias a que el intérprete suele estar en varios sistemas operativos, no es necesario adaptar el código a una plataforma específica.

Aumento de rendimiento en entornos web. Al ser interpretado es ejecutado por el cliente web, consiguiendo así disminuir la carga de trabajo del servidor.

Portabilidad. Por lo comentado anteriormente el programa puede ser llevado de una plataforma a otra sin tener problemas de compatibilidad.

- **Desventajas**

Velocidad de ejecución. Cuando hacemos el uso de un compilador, estamos transformando el programa a código máquina que es en el que trabaja el procesador. Mientras que con el intérprete hay que hacer la correspondiente traducción de cada parte del código en tiempo real lo cual conlleva a una pequeña ralentización. Además, este proceso hay que hacerlo siempre que se vaya a ejecutar el programa.

Necesidad de intérprete. Aunque es multiplataforma, siempre va a depender de algún framework o máquina virtual que haga la función de intérprete.

6.1.2 TensorFlow

Es la plataforma de Aprendizaje Profundo más importante del mundo. Es un desarrollo open-source de Google que, debido a su flexibilidad y gran comunidad de desarrolladores, lo han posicionado como la herramienta líder en el campo del Deep Learning.

Es una biblioteca de software para computación numérica, que utiliza gráficos de flujo de datos. Sirve para construir y entrenar redes neuronales, que permiten detectar y descifrar patrones y correlaciones, análogos al aprendizaje y razonamiento usados por los humanos.

Su arquitectura flexible permite implementar el cálculo a una o más CPU o GPU en equipos personales, servidores o dispositivos móviles con una sola API.

A partir de la popularización y mejora de esta, Google desarrolló Tensor Processing Unit (TPU). Se trata de una arquitectura construida en ASIC específica para el aprendizaje automático. Es utilizado para ejecutar modelos más que para entrenarlos. Tras un largo periodo de pruebas demostraron que su rendimiento era mucho mayor para aplicaciones de aprendizaje automático que con los sistemas tradicionales. (Anón s. f.-j)

Usos:

En cuanto algunas de las aplicaciones que se le está dando actualmente es la mejora de fotografías en smartphones, para ayudar a diagnósticos médicos, en el procesamiento de imágenes

- **Ventajas**

Permite ser ejecutado en la nube o de modo local.

Gran escalabilidad pudiendo ejecutar el mismo programa en varias máquinas.

- **Desventajas**

Requiere de otras herramientas para poder hacer uso de ella de la manera más cómoda.

Si quieres obtener todas las posibilidades que ofrece usando los servicios de Google en la nube requiere de una suscripción de pago.

6.1.3 Keras

Es una de las bibliotecas más reconocidas en Deep Learning. Es de código abierto y trabaja con TensorFlow. Como objetivo tiene simplificar la programación de algoritmos basados en aprendizaje profundo. Ofreciendo así un conjunto de abstracciones más intuitivas y de alto nivel. Haciendo uso de TensorFlow a bajo nivel. Actualmente Keras pertenece al paquete de TensorFlow. Además de ser de código abierto, se encuentra escrita en Python, facilitando su uso y reproducción.

Keras cuenta de dos formas para generar una arquitectura de un modelo. API Secuencial donde se instancia un objeto de tipo Model y a partir de ahí, se van añadiendo las capas que conforman la arquitectura una detrás de otra. API Funcional se define una entrada de la que, a partir de la misma, se va definiendo el resto de la arquitectura. Indicando a cada capa cual es la entrada a cada capa de trabajada. Obteniendo como resultado un objeto modelo. (Anón s. f.-i)



Ilustración 9. Esquema de abstracción de creación de una red neuronal

6.1.4 Entornos de trabajo empleados para el desarrollo

Es una aplicación que proporciona al usuario de una serie de herramientas para facilitar el desarrollo de un proyecto. Para este proyecto se han llevado a cabo varias distinguiéndolo entre aquellos alojados en la nube o los que son usados como aplicaciones instaladas de manera local.

6.1.4.1 *Googel Colab*

Es un entorno de Google Research. Permite a cualquier usuario escribir y ejecutar código arbitrario de Python en el navegador. Es especialmente adecuado para tareas de aprendizaje automático, análisis de datos y educación. Específicamente es un servicio de cuaderno alojado de Jupyter que no requiere configuración y que ofrece acceso sin coste adicional a recursos informáticos, como GPUs a través de los servidores de la propia compañía.

- **Ventajas**

Proporciona un entorno en el que desarrollar de manera rápida y sin la necesidad de configuración del entorno.

El proyecto está almacenado en la nube lo que ofrece una gran portabilidad, pudiendo continuar el desarrollo en cualquier equipo conectado a internet.

Ofrece la posibilidad de compartir y trabajar por varias personas en el mismo proyecto, usando el mismo entorno.

Te permite usar de las capacidades computacionales de sus servidores, como sus GPUs, TPUs, etc.

- **Desventajas**

Los recursos son limitados y si hay necesidad de un acceso más fiable a los servicios, hay que optar por la suscripción de Colab Pro la cual es de pago.

El entorno está limitado, no permitiéndote el desarrollo de aplicaciones fuera del ámbito de estudio y el análisis de datos. Así como no poder alojar archivos multimedia, conectarse a proxies, usar un escritorio remoto o SSH, minar criptomonedas, etc.



6.1.4.2 Visual Studio Code:

Visual Studio Code es un editor de código optimizado con soporte para operaciones de desarrollo como depuración, ejecución de tareas y control de versiones. Su objetivo es proporcionar las herramientas que un desarrollador necesita para un ciclo rápido de compilación y depuración de código y deja los flujos de trabajo más complejos para los IDE con funciones más completas.

Es usado para una infinidad de proyectos de distintos propósitos. El entorno adquiere un gran valor gracias a las extensiones. Con estas te permite programar en cualquier lenguaje de programación, modificar los estilos de la sintaxis para que sea lo más acorde al tipo de proyecto, incluir frameworks o herramientas de trabajo. Entre los proyectos más frecuente en el que se hace uso para su desarrollo son: el desarrollo web, aplicaciones móviles o para la programación de hardware como microcontroladores.

- **Ventajas**

Es multiplataforma, estando disponible en todos los sistemas operativos.

Ofrece una edición de código, autocompletado y resaltado de sintaxis bastante completo, permitiendo que sea más ágil escribir código.

También cuenta con un depurador que facilita la detección de errores del programa de manera más precisa.

Es compatible con herramientas como GitHub, teniendo un mayor control en las versiones del proyecto en desarrollo.

- **Desventajas**

No ofrece todas las posibilidades que tiene un IDE convencional, teniendo algunas limitaciones en cuanto a realizar algunas configuraciones o en usos como el depurador que es menos versátil.

Resulta más incómodo al realizar tareas sencillas como cambiar de archivo, usar el comando git, cambiar de configuración.

Muchas funcionalidades deben de ser incorporadas a través de plugins.

6.1.4.3 Notepad++ con Anaconda Prompt:



Notepad++ es un editor de texto y software de código abierto basado en Microsoft Windows. Ayuda a editar código fuente usando varios lenguajes de programación. Como c++, Java, C, R, JSON, XML, Python...

Anaconda es una distribución de los lenguajes de programación Python y R para computación científica. Tiene como ventaja simplificar la gestión e implementación de paquetes. La distribución incluye paquetes de “data science” adecuados para todos los sistemas operativos.

- **Ventajas**

Hace que sea muy fácil el editar y escribir algoritmos complejos, simplificando la implementación de proyectos de ciencia de datos.

No requiere de una compleja instalación y configuración para su uso.

- **Desventajas**

Es muy limitado en cuanto a los servicios que puede ofrecer un entorno de desarrollo.

Hace que sea imposible el uso de ninguna herramienta para detectar errores por lo que dificulta mucho la tarea.

6.1.5 GitHub

GitHub es una plataforma de alojamiento de código para el control de versiones y la colaboración. Te permite a ti y a otros, colaborar y realizar cambios en un proyecto a su vez que mantienen un seguimiento detallado de su progreso desde cualquier lugar. Un repositorio se suele utilizar para organizar un solo proyecto. Los repositorios pueden contener carpetas y archivos, imágenes, videos, hojas de cálculo y conjuntos de datos, cualquier cosa que necesite su proyecto. El control de versiones es un sistema que ayuda a rastrear y gestionar los cambios realizados en un archivo o conjunto de archivos. Permitiendo a los desarrolladores analizar todos los cambios y revertirlos sin repercusiones si se comete un error.



- **Ventajas**

Búsqueda muy rápida en la estructura de los repositorios.

Amplia comunidad y fácil encontrar ayuda.

Ofrece herramientas muy prácticas de cooperación, además de contar con una buena integración a Git.

Es fácil integrar con otros servicios externos.

El servicio gratuito es bastante completo.

- **Desventajas**

Tiene limitaciones de espacio, sin poder exceder los 100MB por un archivo y un repositorio está limitado a 1GB en la versión gratis.

No es abierto del todo, además de tener que aceptar la política de manejo de tus proyectos para los intereses propios de la compañía.

6.2 DESARROLLO

Esta fase se va a dividir en varias partes de las que ha constado la realización del proyecto. Especificando en cada una de ellas varios aspectos como su funcionamiento o las decisiones tomadas a la hora de que metodología usar, además de algunos otros aspectos más específicos a destacar.

6.2.1 Procesamiento de los datos

Esta etapa ha sido la primera en llevar a cabo ya que es requisito indispensable para poder realizar las posteriores. Dentro de esta se podría subdividir en otras subfases claramente definidas.

Los datos a usar para el sistema corresponden a audios de latidos de corazones con sus correspondientes etiquetas indicando que tipo de patología sufre el paciente de dicha muestra o ninguna, en caso de que así fuese indicándolo correspondientemente. Estos datos han sido recolectados de una página donde había una recopilación de ellos subdivididos en dos grupos, los cuales llamaremos A y B. Ambos grupos presentan prácticamente las mismas características, compuestos por muestras de audios las cuales serán diferentes en cada grupo. La principal diferencia está en las etiquetas dadas, las cuales en cada grupo hay algunas etiquetas que no aparece en el otro, pero, que según el tratamiento que hemos dado a los datos como ahora se explicará, no afectará al sistema, haciendo que esta diferencia sea prescindible.

Antes de realizar cualquier tipo de tratamiento de los datos es importante dar una estructuración correcta a los mismos. Los datos están etiquetados en dos ficheros tipo “csv”, uno para cada grupo de los datos mencionados, en el que se indica la ruta del audio, para facilitar la posterior manipulación en el programa y su etiqueta correspondiente a dicho audio indicando el tipo de patología. Quedando el esquema como se muestra en la siguiente figura.

```
dataset, fname, label, sublabel
a, set_a/201012172012.wav, artifact,
a, set_a/201105040918.wav, artifact,
b, set_b/Btraining_extrastole_127_1306764300147_C2.wav, extrastole,
b, set_b/Btraining_extrastole_128_1306344005749_A.wav, extrastole,
```

Ilustración 10. Organización de los datos

Como se puede observar en la parte señalada en rojo. Esas son las etiquetas como mencionábamos.

Donde para cada grupo de los datos nos encontramos las siguientes etiquetas.

A	Artifact	Murmur	Normal	-	Extrahls	
B	Extrastole	Murmur	Normal	-	noisynormal	noisymurmur

Tabla 5. Etiquetado de los datos

Para facilitar la creación del sistema y con el fin de poder llegar a una solución con unos mejores resultados, todas las etiquetas que no son “Normal” van a ser modificadas a “Abnormal”. Quedando así la tabla de etiquetas.

A	Normal	Abnormal	-
B	Normal	Abnormal	-

Tabla 6. Nuevo etiquetado tras el procesado

La etiqueta “-” corresponde a que dicha muestra de audio no tiene ninguna de las etiquetas anteriores asignadas, ya sea porque no fue indicada o porque la calidad del audio es muy mala. Los datos con dicha etiqueta serán eliminados del sistema.

Una vez que ya conocemos como están estructurados los datos y cuáles son los que vamos a emplear, va a ser necesaria una división de estos en dos conjuntos. Uno dedicado al entrenamiento y otro como test o validación, siendo la distribución de un 80%-20% respectivamente. Para que la distribución sea válida, los datos son distribuidos de manera desordenada y aleatoria a cada uno de estos dos conjuntos. Así se evita la posibilidad de que para tanto el conjunto de entrenamiento como el de test no haya sólo elementos con características similares, permitiendo que en cada grupo haya en la mayor posibilidad una distribución uniforme de los datos.

Antes de realizar dicha división, es importante hacer una serie de pasos previos con los datos. Los conjuntos de datos mencionados anteriormente son creados a partir de realizar un procesado sobre estos datos. Antes de nada, se va a considerar que todos los datos que estén formados por muestras de audios inferiores a 3 segundos van a ser descartados. Por lo que lo primero es comprobar la duración de todas las muestras de audio. Todas aquellas que superen el umbral comentado antes, se les va a realizar un procesado que consiste principalmente en hacerles una segmentación.

La segmentación consiste en coger cada muestra de audio y a partir de esta dividirla generando nuevos. Esta segmentación debe ser efectuada usando algún criterio específico. Para el desarrollo de nuestra solución se ha escogido como criterio la división del audio por una marca de tiempo. Hay también que mencionar que se han considerado otras métricas para la realización de dicha separación. Vamos a comentarlas cada una de ellas, las problemáticas que se han encontrado con cada una y porque se ha escogido la mencionada.

6.2.1.1 No segmentar

Es la solución más simple ya que no requiere de hacer tantas operaciones. Hay un factor importante que tener en cuenta. Para que nuestro sistema funcione del modo más correcto requiere que los datos sean heterogéneos, que no muestren grandes diferencias entre ellos. Esto implica que todas las muestras de audio deben de tener una duración igual. Por lo que el sistema no puede recibir directamente todas las muestras de audio sin realizar ningún tipo de tratamiento.

Esto se podría solucionar fácilmente y sin considerar el segmentar los audios. Para ello sólo habría que determinar una duración para que todos la compartiesen. Como primera problemática que tenemos es determinar esa duración. Una posible solución sería la de la duración media de todos los audios. Si se hace el cálculo, se obtiene en torno a una duración de 6`5 segundos. Usando este tiempo implicaría el descarte de todas las muestras de audio inferior a esa duración. Haciendo un conteo de todos los datos menores a esa marca, se obtiene que son cerca de la mitad del total. Esto nos llevaría al descarte de una gran cantidad de los datos, lo cual no es conveniente ya que interesa tener la mayor cantidad posible para poder obtener una solución más precisa. Además, que los audios que si se encontrasen por encima de la frontera estarían perdiendo parte de la información que para los audios de una duración mucho mayor puede llegar a resultar crítico. Debido a estos motivos el uso de esta ha sido descartada.

6.2.1.2 Segmentar en función de los latidos del corazón

Este puede ser un gran modo de dividir los audios ya que la parte que más interesante de los audios son los pulsos de los latidos del corazón. La idea es analizar las muestras de audio y detectar los instantes de tiempo donde se producen un latido. Así segmentado el audio en intervalos marcados por cada uno de los instantes de tiempo registrados previamente. La problemática de esta es poder reconocer en el audio los latidos. Tras una búsqueda de posibles soluciones propuestas para poder conseguir dicho objetivo, no se ha conseguido encontrar ninguna que cumpliese con dicho problema. Como alternativa sería construirnos un sistema capaz de modelar el problema y que supiese reconocer los latidos. Pero requeriría de una mayor complejidad por todo lo que supondría, tanto de tener que buscar un conjunto de datos, muy similar al que usamos, pero en el que el etiquetado tuviese un listado de todos los tiempos donde se produce un latido para cada muestra de audio. Con el cual no contamos y además de crear el modelo específico para construir la solución, sin tener unas garantías de llegar a obtener un resultado lo medianamente decente para poder hacer lo que se pretende.

6.2.1.3 Segmentación del audio por una duración de tiempo fija dada

Esta es la solución finalmente optada para la implementación del sistema. Consiste en dividir el audio en partes por un tiempo determinado. En concreto, la división se ha hecho con un tiempo de 3 segundos considerando que, con esa duración, hay suficiente información, ya que siempre va a haber latidos del corazón y, además, no es un tiempo elevado permitiendo que se dividan más veces el audio y teniendo así más muestras, lo que es beneficioso para el sistema. Con el fin de lo comentado de crear más muestras, se ha determinado por la división de modo que el último segundo se solape con el primero del siguiente, así se generan más audio a consta de que se repita un poco de información

la cual no repercute a la hora usar los datos el sistema. Así entonces a modo de ejemplo, Si un audio dura 9 segundos como se propone aquí se crearía nuevos datos que constaría con una división del audio original en los siguientes trozos:

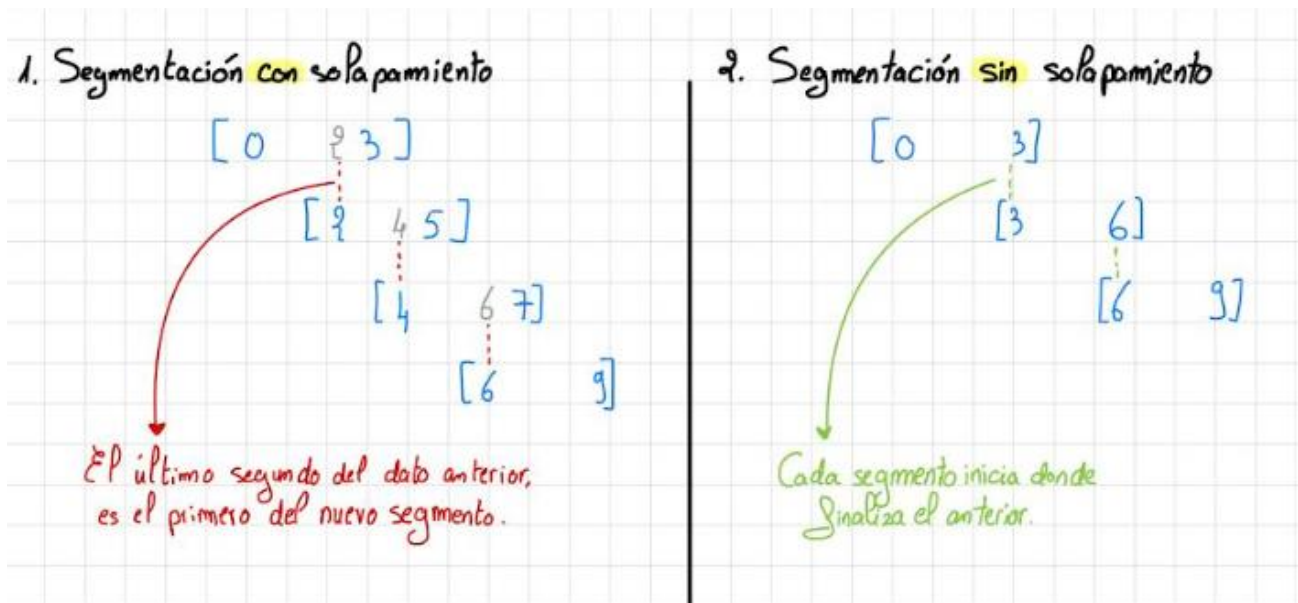


Ilustración 11. Modos de segmentación por tiempo fijo

Cada corchete representa un nuevo dato y los valores de su interior es el intervalo de tiempo de la muestra de audio original con la que se ha formado.

De este modo como comentábamos se crean más datos a partir de una muestra de audio a consta que cada nuevo dato repite un segundo de información con respecto al anterior.

Este ha sido el criterio escogido para la implementación de la solución debido a que no requiere de una gran complejidad, se consigue aumentar la cantidad de los datos y se unifica la duración de todos ellos, lo cual es un requisito fundamental como se mencionaba antes.

Finalmente, tras todo lo explicado en este apartado se consigue tener los datos distribuidos en dos conjuntos de la manera adecuada y habiendo aplicado la correspondiente segmentación para que los datos estén con las características idóneas para el sistema.

6.2.2 Procesamiento del audio:

Tras lo conseguido en la fase anterior se puede hacer el procesamiento de los audios con el fin de obtener una mayor validez en los mismos, para facilitar y conseguir una mejor solución del sistema.

Este procesamiento consiste en aplicar algunas de las técnicas más populares sobre el procesado de señales, específicamente de audio. Se va a explicar en qué consiste cada una de ellas y mostrando sus cualidades para argumentar que opción ha sido la aplicada para la implementación final.

6.2.2.1 Transformada rápida de Fourier (FFT)

Es un importante método de medición usado para audio y acústica con una gran variedad de aplicaciones. En esencia FFT es un algoritmo eficiente que permite calcular la transformada discreta de Fourier (DFT) y su inversa. Lo que se consigue tras su aplicación es descomponer una señal en sus componentes espectrales individuales y así proporciona información sobre su composición. En cuanto al tratamiento de señales, FFT impone algunas limitaciones en la señal y en el espectro resultante ya que la señal muestreada y que se va a transformar debe consistir en un número de muestras igual a una potencia de dos.

Aplicando este método sobre cada audio se obtiene una lista en la que cada valor representa una frecuencia. Usar sólo esto resulta ser ligeramente más eficiente al ser algo más rápido en el procesado, pero lo obtenido no es conveniente para el sistema puesto que lo obtenido para cada uno de los datos no tiene la dimensión requerida para la fase siguiente de la implementación del sistema. En esto se profundizará más adelante. (Anón s. f.-f)

6.2.2.2 Espectrograma de Mel

Un problema que tiene la representación en tiempo-frecuencia es que todas las bandas frecuenciales son tratadas del mismo modo, aunque en aplicaciones de audio no todas tienen la misma importancia. El oído tiene un comportamiento en frecuencia que no es lineal, por lo que aumentar una misma cantidad la frecuencia de un sonido no tiene el mismo efecto sensorial a una frecuencia u otra. Si comparamos como es percibido el cambio de 100 Hz a 200 Hz, tenemos una variación muy notoria de la frecuencia, mientras que pasar de 10000 Hz a 10100 Hz es apenas perceptible.

El comportamiento del oído frente a la frecuencia se puede modelar de manera aproximada como un logaritmo. Esto equivale a que, si se multiplica la frecuencia por un mismo factor, la sensación de variación de frecuencia que se obtiene es la misma, a independencia de la frecuencia con la que se está realizando.

El inconveniente del espectrograma en escala lineal de frecuencias es que sobre representa las altas frecuencias. Pero, la representación lineal otorga la misma relevancia que representa una sola octava (de 10 kHz a 20 kHz).

Si se expresa logarítmicamente, la frecuencia de cada octava dará la misma información independientemente de su posición en la escala. Sin embargo, el oído no tiene una respuesta logarítmica perfecta. A bajas frecuencias el comportamiento es más lineal ya partir de unos 1000 Hz el comportamiento es más logarítmico.

La escala de Mel se introdujo en la década de 1930 para reflejar mejor el comportamiento del oído. Se usa convirtiendo el valor de la frecuencia a una unidad en hercios llamada Mel usando la siguiente fórmula de conversión:

$$m = 1127 \ln \left(1 + \frac{f}{700} \right)$$

$$f = 700 \left(e^{\frac{m}{1127}} - 1 \right)$$

Ilustración 12. Ecuaciones del espectrograma de Mel

La curva siguiente muestra la forma de la curva en función de la frecuencia:

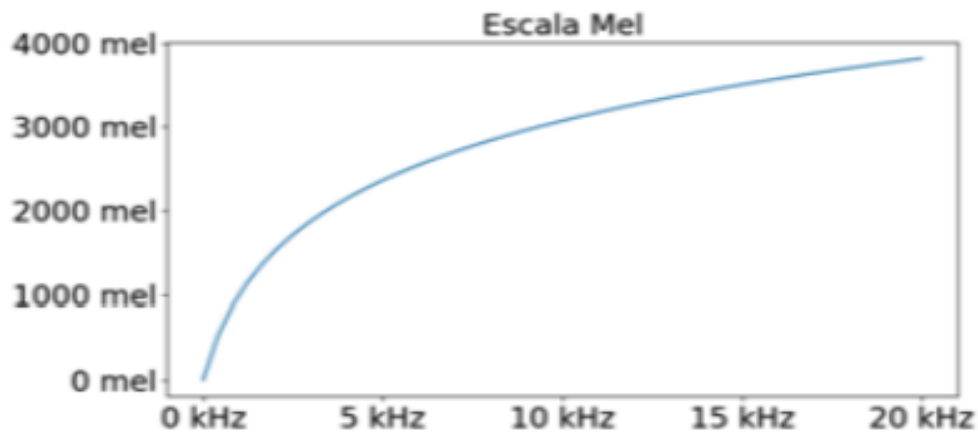


Ilustración 13. Gráfico de la curva entre la relación de la escala de Mel y la frecuencia

Si aplicamos en nuestra solución este procedimiento obtenemos por cada muestra de audio una imagen que representa al audio donde los distintos valores de los colores por cada uno de los píxeles son un valor dentro de la escala de Mel como se ha visto antes representando cada una de las frecuencias del audio. La siguiente imagen muestra el espectrograma de Mel obtenido tras haber sido aplicado a una muestra de audio.

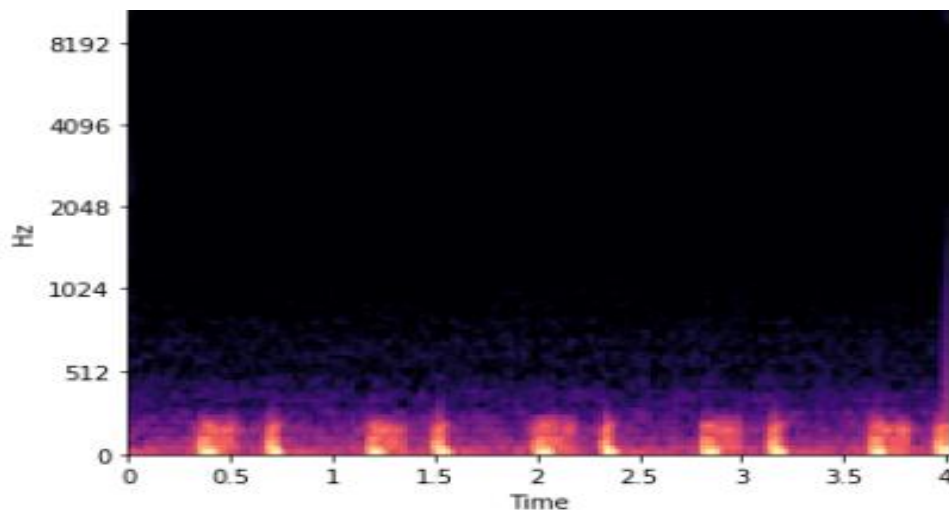


Ilustración 14. Imagen originada tras aplicar Mel a una muestra de audio

(Rodriguez y Garcia s. f.)

6.2.2.3 MFCCs (Mel Frequency Cepstral Coefficients)

MFCC es un coeficiente de representación del habla basado en la percepción auditiva humana. Su éxito nace por el auge dado en el área del reconocimiento de audio automático al extraer las características más relevantes de una señal de audio, despreciando todas aquellas partes que otorgan de una información poco relevante como el tono, el ruido o

el volumen. MFCC muestra las características locales de la voz. El sonido asociado al canal según el modelo fuente de filtro.

Una parte importante son los coeficientes cepstrales, estos se obtienen a partir de la transformada de Fourier o de la transformada discreta del coseno, pero la principal característica es que las bandas de frecuencia de la MFCC se encuentran logarítmicamente según la escala de Mel, donde es definido un punto de referencia. Este corresponde a un tono de 1000 Hz que está 40 dB por encima del umbral de audición del oyente, siendo concretamente 1000 tonos de Mel.

Su cálculo se realiza como muestra el siguiente diagrama. (Agarwal [2019] 2022)

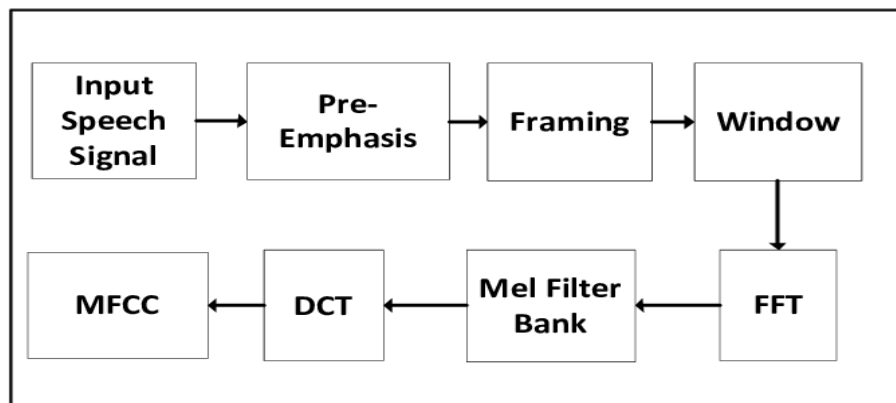


Ilustración 15. Diagrama del proceso de aplicar MFCC

(MFCC descripción del funcionamiento)

Aplicando a una muestra de audio el procedimiento, se obtiene la siguiente imagen como dicha representación.

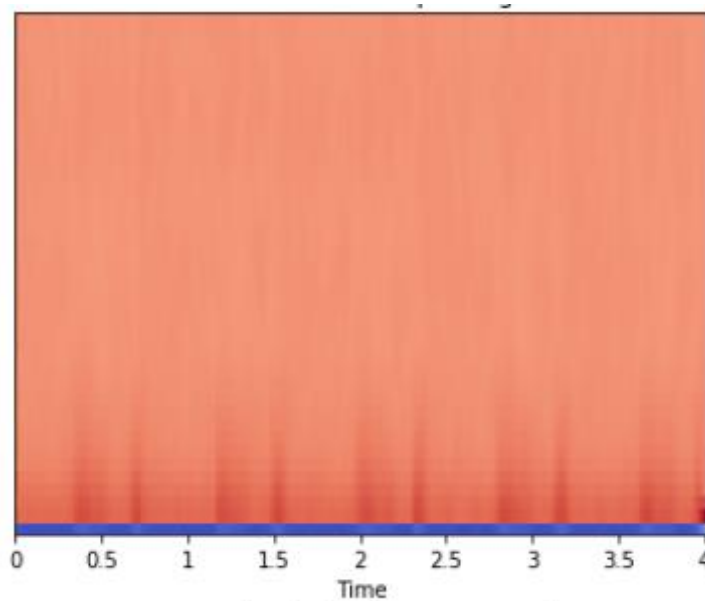


Ilustración 16. Imagen resultante tras aplicar MFCC a un audio

Aplicando lo comentado en esta fase para dos muestras de audio, una etiquetada como normal y otro como abnormal, se obtienen las siguientes representaciones.

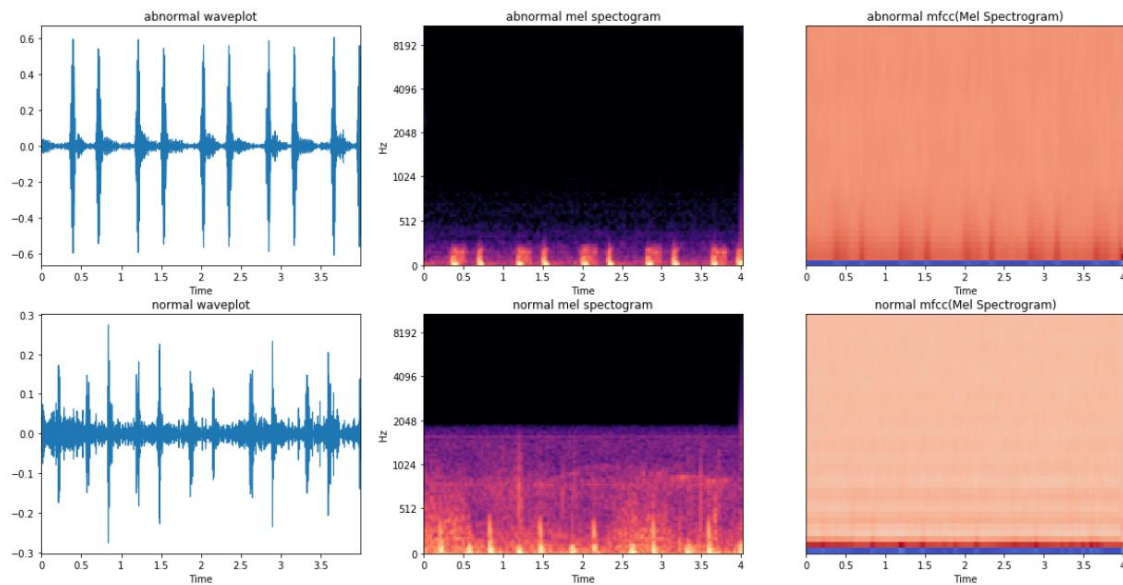


Ilustración 17. Señal de audio, Mel y MFCC

6.2.3 Modelo de entrenamiento (Red Neuronal)

6.2.3.1 Redes Neuronales

Las redes neuronales son un modelo computacional que emula el comportamiento del humano para procesar la información. Están formadas por multitud de neuronas. Estas neuronas son la unidad más básica y suelen estar organizadas en capas, conectándose entre sí para transmitir señales. Las redes normalmente se forman por una capa de entrada en la que cada unidad representa los parámetros de entrada, una o varias capas ocultas y una capa de salida donde cada unidad representa las posibles soluciones o clasificaciones.

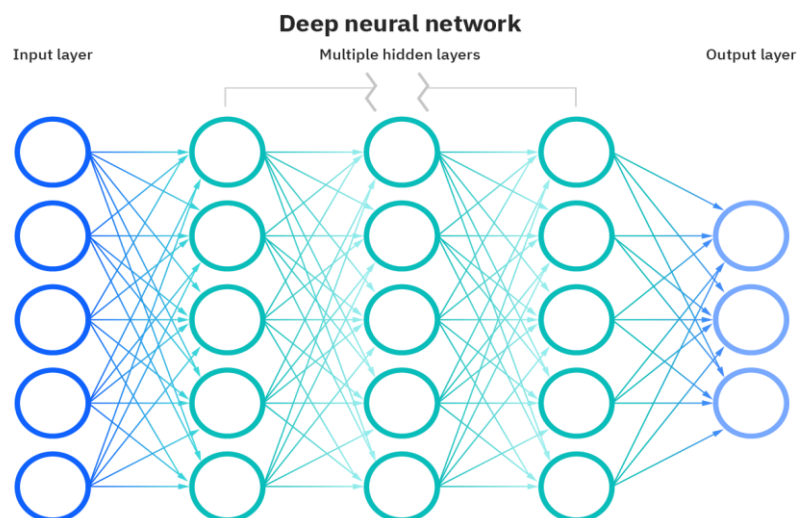


Ilustración 18. Aspecto de una red neuronal

(Anón 2021a)

Cada nodo individual o neurona funciona como un modelo de regresión lineal, obteniendo un resultado a partir de los datos de entrada, a los que se le ha aplicado una suma ponderada y un sesgo.

$$\sum_{i=1}^m w_i x_i + bias = w_1 x_1 + w_2 x_2 + w_3 x_3 + bias$$

Ilustración 19. Suma de las ponderaciones de una neurona

Para obtener una salida más compleja y que pueda conseguirse un modelo capaz de representar más acorde los datos, se le aplica algún tipo de función, que funciona como filtrado, a la salida obtenida por dicha suma de ponderaciones. Gracias a esto evitamos que el modelo obtenido no sea simplemente un modelo de regresión lineal y que se puedan encadenar las neuronas entre capas. Ya que, el resultado obtenido de una cadena de sumas de regresiones lineales obtiene como resultado otra regresión lineal, lo que equivaldría a que el modelo se realizase utilizando una única neurona.

A dicha función se le conoce como función de activación. Algunas de estas funciones más utilizadas son: Sigmoidal, Lineal, Gaussiana, Escalonada, ReLu, etc.

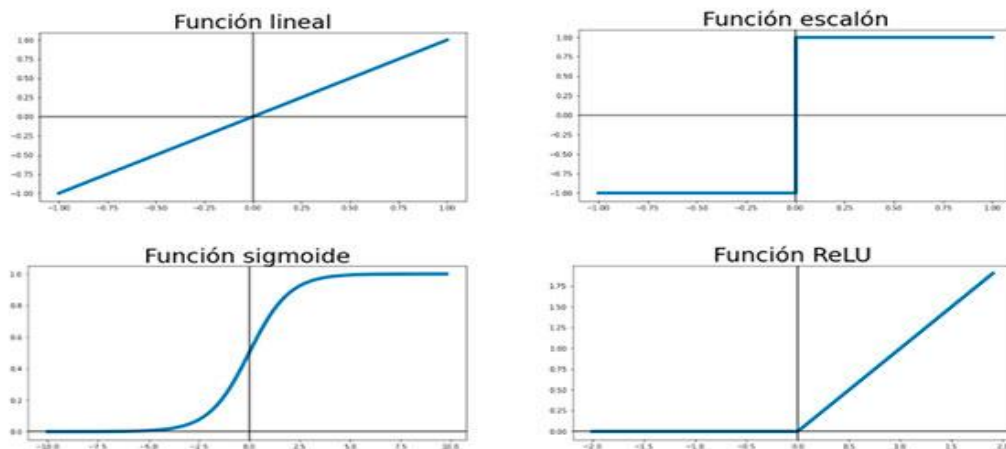


Ilustración 20. Funciones de activación

(Anón 2021c)

Las redes neuronales se pueden clasificar según su modo de aprendizaje. Este modo es el que define qué técnica va a usar para ajustar sus parámetros en el proceso de entrenamiento del sistema.

- **Aprendizaje supervisado**

Consiste en ir mejorando a la red, a partir de realizar el cálculo del error obtenido por el resultado predicho sobre el resultado real de los datos de entradas de los cuales conocemos su clasificación. Con el fin de hacer varias iteraciones hasta que los resultados obtenidos sean lo más cercano posible a la salida esperada.

- **Aprendizaje no supervisado**

En este paradigma se desconoce sobre los datos cuál es su correspondiente clasificación. Se basa en que la red sea capaz de aprender a través de la observación de los datos de entrada y que se reconozcan sobre ellos ciertos patrones, características, regularidades de similitud para poder determinar cuál es la estructura interna que genera los datos.

- **Aprendizaje por refuerzo**

Se basa en crear un sistema de recompensas en el cual los resultados correctos son premiados a través de un refuerzo positivo, mientras que los erróneos son disuadidos.

- **Aprendizaje híbrido**

Utiliza una combinación del resto de paradigmas mencionados. La red aprende examinando cada uno de los datos de entrada, por cada uno de ellos realiza una predicción y si el resultado es incorrecto, ajusta las ponderaciones. Este proceso es repetido por muchas iteraciones, hasta haber alcanzado algún criterio de parada. A dichas iteraciones se les conoce como épocas.

Todas las ponderaciones con las que inicia pueden ser dadas por algún modelo ya creado o generadas de modo aleatorio. Por lo que, al inicio lo habitual es que la red de unas respuestas erróneas. La mejora de su predicción se consigue a través del entrenamiento que consiste en ese ajuste de las ponderaciones comentado anteriormente. El ajuste se consigue a través del cálculo del error obtenido tras los resultados obtenidos por varias predicciones. Este sirve para modificar las ponderaciones de la red y ajustar de modo gradual en cada iteración. Las ponderaciones ayudan a conocer la importancia que tiene cada una de las variables de entrada, siendo las de valor mayor las que dan más relevancia e importancia a dicha entrada. Con el cálculo del error se ajustan las ponderaciones, de modo más agresivo a dichas ponderaciones que han tenido mayor peso para obtener la predicción errónea.

La siguiente fórmula representa el cálculo del error cuadrático medio, el cual es el más comúnmente utilizado.

$$MSE = \frac{1}{2m} \sum_{i=1}^m (\hat{y} - y)^2$$

Ilustración 21. Cálculo del error cuadrático

Para hacer el cálculo de dicho error y reajuste de las ponderaciones según el criterio comentado antes, se usa una técnica denominada **Backpropagation**:

Que consiste en realizar el cálculo de los ajustes de las ponderaciones, a partir del error obtenido como salida para cada una de todas las neuronas que componen la red. El resultado del error de salida es propagado hacía en sentido inverso. Desde la capa de salida hacía la capa de entrada. Así se reduce mucho el tiempo de cálculo para ajustar las neuronas, además de que se detecta fácilmente cuales han influenciado en mayor medida para el resultado de la predicción.

Con el avance de las iteraciones y mejora de los ajustes de las ponderaciones, se consigue hacer que la solución se acerque a puntos locales o puntos de convergencia. Tras un proceso largo iterativo se puede producir lo que se conoce como **Overfitting**:

El modelo obtenido se ha conseguido tras sobreentrenar el sistema de aprendizaje, obteniendo un sobreajuste de los parámetros. Por lo que para los datos de entrenamiento se consiguen unos resultados con un alto grado de precisión en el acierto de las predicciones y con un resultado de error muy bajo. Pero cuando se usa el modelo para predecir datos que no han sido usados en la etapa de entrenamiento consigue obtener un porcentaje muy alto de error. Esto se produce porque el modelo representa con mucha fiabilidad a los datos de entrenamiento, sin ser capaz de generalizar. Esto es lo que se conoce como que la red ha memorizado los patrones de los datos.

En la imagen se muestra lo que podría ser el valor del error de un sistema por cada iteración.

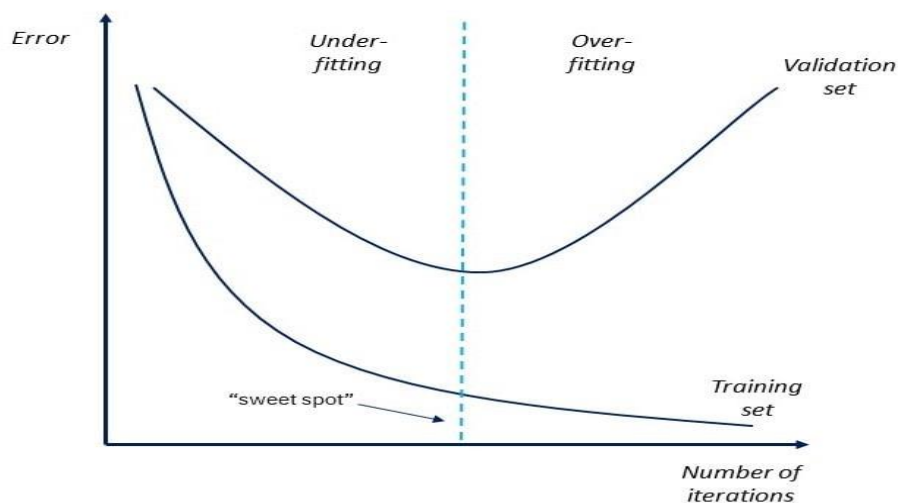


Ilustración 22. Gráfico de la evolución del error durante el entrenamiento de la red

El caso opuesto a Overfitting es cuando el sistema no es capaz de encontrar un ajuste de los parámetros para poder crear un modelo que represente en la medida de lo posible al conjunto de datos;

A esto se le conoce como **Underfitting** y puede ser producido por muchos motivos, ya sea que la red empleada requiere de más capas, más neuronas por capas, un ajuste más agresivo sobre el cálculo del error...

Estos suelen ser de los problemas más habituales cuando se trata de trabajar un sistema de red neuronales donde de los factores más importantes a tener en cuenta es ser capaz de crear un modelo con la mayor generalización posible que sea representativo para la mayor cantidad de datos.

Las redes según la complejidad del problema y el tipo de datos de entrada se pueden clasificar según su tipología y disposición de las neuronas. Siendo entre las más comunes:

- **Redes neuronales monocapa:** Son la red más básica y simple con una conexión de una sola capa. Usadas para clasificar patrones que son linealmente diferenciables.
- **Redes neuronales multicapa:** Se colocan una serie de capas ocultas intermedias entre la de entrada y salida. Consiguiendo crear modelos más complejos y que permitan eliminar información que no es relevante para la solución.
- **Redes neuronales recurrentes:** Estas cuentan con retroalimentación entre neuronas de diferentes capas, que se encuentran en la mismo o con una neurona así mismo.
- **Redes neuronales convolucionales:** Son las más complejas. Cuentan con varias capas diferentes, en las que cada una de ellas está entrenada para llevar a cabo una tarea diferente. Están diseñadas emulando un comportamiento muy similar al que hacen las neuronas de la corteza visual primaria de un cerebro biológico. Están especialmente diseñadas para el problema que supone el crear un modelo capaz de identificar imágenes grandes. Debido a que son muchos los parámetros de entrada.

Este último tipo es el usado para la implementación de nuestra solución. Debido a las grandes capacidades que ofrece y ya que los datos de entrada empleado para el sistema se componen de muchas características, esto implica que la red neuronal va a tener muchas neuronas de entrada y justo para eso las convolucionales son las más apropiadas. (Anón 2021b)

6.2.3.2 Métricas de evaluación

Para evaluar un modelo de redes neuronales se utilizan distintas métricas con los que medir su calidad y conocer fiabilidad como clasificador.

Las siguientes métricas hacen uso de una serie de términos que son referidos al resultado obtenido tras una clasificación. Estos términos muestran un balance de lo obtenido tras el resultado obtenido de clasificar todo el conjunto de datos.

Dichos términos son:

- **True Positives (TP):** Estos son los valores positivos predichos correctamente, lo que significa que tanto el valor de la clase real como el de la clase predicha son sí.
- **True Negatives (TN):** Estos son los valores negativos predichos correctamente, lo que significa que el tanto valor de la clase real como el de la clase predicha son no.
- **False Positives (FP):** Cuando la clase real es no o no existente y el resultado de la predicción lo atribuye a alguna clase, asignándolo como sí.
- **False Negative (FN):** Cuando una clase es detectada como positiva, pero, se clasifica erróneamente a otra clase, asignándose como negativa.

Aplicando estos parámetros se puede realizar el cálculo de las siguientes métricas:

- **Accuracy**

La precisión es la medida de rendimiento más simple y se trata de la relación existente entre la observación correctamente predicha y el total de observaciones. Esto podría llevar a usarla a modo, que, si tenemos una alta precisión, nuestro modelo es el mejor. En cierta medida sí podría ser una buena métrica para conocer la calidad de nuestro modelo, pero sólo es realmente válido cuando se tienen conjuntos de datos simétricos en los que los valores de los falsos positivos y los falsos negativos son casi iguales. Por lo tanto, hay que tener en cuenta otros parámetros para evaluar el rendimiento del modelo.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

- **Precision**

Es la relación entre las observaciones positivas predichas correctamente y el total de observaciones positivas predichas. La alta precisión se relaciona con la baja tasa de falsos positivos.

$$Precision = \frac{TP}{TP + FP}$$

- **Recall**

Es la relación entre las observaciones positivas predichas correctamente y todas las observaciones donde la clase es sí.

$$Recall = \frac{TP}{TP + FN}$$

- **F1 Score**

Es la media ponderada de Precision y Recall. Por lo tanto, esta puntuación tiene en cuenta tanto los falsos positivos como los falsos negativos. No es tan intuitivo como Accuracy, pero F1 suele ser más útil, especialmente si se tiene una distribución de clases desigual. Accuracy funciona mejor si los falsos positivos y los falsos negativos tienen un coste similar. Si el coste de los falsos positivos y de los falsos negativos es muy diferente, es mejor tener en cuenta tanto Precision como Recall.

$$F1\ Score = 2 * \frac{Precision * Recall}{Precision + Recall}$$

6.2.3.3 Matriz de Confusión

Es una herramienta que permite visualizar el desempeño obtenido por un modelo. Representa los resultados obtenidos dividiendo cada columna el número de predicciones de cada clase y las filas en las etiquetas de las clases reales. Esto permite ver de modo claro que tipos de errores y aciertos está cometiendo el modelo a la hora de predecir sobre los datos del conjunto de entrenamiento.

		Actual	
		Positivo	Negativo
Predicción	Positivo	Verdadero Positivo (TP)	Falso Positivo (FP)
	Negativo	Falso Negativo (FN)	Verdadero Negativo (TN)

Tabla 7. Estructura de una matriz de confusión

(Solutions 2016)

6.2.4 Parámetros e hiperparámetros de la implementación de la red

Los parámetros son aquellos fijos determinados por la arquitectura de la red para que funcione correctamente. En cuanto a lo hiperparámetros, estos si son modificables y sirven para ajustar el modelo, cambiando como va a actuar durante el entrenamiento, con el fin de que consiga realizar mejores predicciones. Estos son los siguientes:

- **Batch size:** Definen el número de datos que pasarás por la red para realizar el correspondiente cálculo del error y corregir los pesos.
- **Learning rate:** Especifica la agresividad con la que va a actuar en la modificación de los pesos de las neuronas a partir del error calculado. Esta puede influir mucho en la velocidad con la que el modelo consiga ir mejorando o que el modelo avance de un modo más escalonado o lineal en la mejora de su predicción y reducción del error. Su valor oscila entre 0 y 1. Cuanto más cercano es a 1 realiza una modificación mayor sobre los pesos. Mientras que con un valor bajo cercano a 0, los pesos sufren una modificación más leve por cada iteración. Un valor muy diminuto podría hacer que se tardase mucho tiempo en alcanzar una solución óptima. Por otro lado, un valor muy alto haría que se modificasen muchos los pesos en cada iteración haciendo que cometiese muchos errores y dificultando mucho la labor de encontrar un modelo con un buen índice de predicción.
- **Epochs:** Establece el número de iteraciones de entrenamiento que se van a realizar.
- **Callbacks:** No es un requisito usarlo, pero resulta útil para conseguir que durante el entrenamiento se hagan ciertas peticiones que se solicitan a través de esta llamada. Algunas de estas pueden ser; que el modelo se vaya guardando por cada iteración en el que se ha conseguido mejorar, o que sí a partir de un número de épocas determinadas no ha conseguido mejora que deje el entrenamiento. Emplea las siguientes llamadas: ModelCheckpoint, Early stop.

6.2.5 Análisis de los resultado y comparación de modelos

La solución propuesta en la implementación ha sido el resultado de realizar varias pruebas aplicando las diferentes posibilidades que se pueden hacer para cada una de las distintas partes comentadas anteriormente. Así, la idea es mostrar el resultado que se ha conseguido con la solución final y hacer una comparación entre las distintas soluciones que se ha hecho de la implementación con el objetivo de comprobar porque la escogida final ha sido la mejor.

A la hora de implementar un sistema de redes neuronales, hay que tener en cuenta los hiperparámetros por lo comentado antes. Por lo que se va a argumentar cuales han sido los valores determinados para cada uno de ellos con el fin de conseguir un sistema óptimo. Consiguiendo un modelo que sea aceptable en cuanto a su porcentaje de predicción y error, y que no requiera de un tiempo excesivo para conseguirlo

Para que el análisis pueda sea más riguroso y justo, se ha decido optar por usar dichos hiperparámetros con los mismos valores para todas las pruebas realizadas con los distintos modelos. La elección de los valores de los parámetros ha sido tras comprobar que se consigue un buen resultado al usarlos en dos modelos distintos.

Los valores empleados para dichos hiperparámetros a lo largo de todas las implementaciones son los siguientes:

→ **Batch size = 128.**

Este valor suele ser muy típico empleado en un entrenamiento de redes. Un valor muy alto dificulta mucho realizar un buen ajuste de los pesos, mientras que un valor bajo necesita de más épocas para conseguir un mejor ajuste. Por lo que con este valor se consigue un buen balance entre ambos casos.

→ **Early stop = 10.**

Es el número de épocas, que tienen que pasar, sin que mejore el valor de val_loss para que se detenga el entrenamiento. Se estima que si pasan este número de épocas sin mejorar, es debido a que no va a conseguir mucha mejoría a partir de muchas iteraciones siguientes o, lo más probable, que se haya alcanzado una situación de Overfitting

→ **ModelCheckpoint.**

Con este parámetro se está indicando que en cada iteración compruebe si se ha conseguido mejorar el modelo, para ello comprueba si el valor de val_loss ha disminuido, si ha disminuido este modelo es guardado automáticamente. Con esto se consigue que cuando se termine el entrenamiento, acabe registrado el modelo que ha conseguido un menor val_loss. De otro modo se almacenaría el último, el cual no garantiza que sea el mejor.

→ **Epochs = 100.**

Este valor realmente no repercute mucho en el entrenamiento ya que para la mayoría de entrenamientos se detienen antes de alcanzarlo debido al parámetro anterior. Y en caso de llegar ya se ha alcanzado un buen nivel de ajuste.

→ **Validation.**

Para la validación de cada iteración se usan el 20% del conjunto de datos como se comentó en la sección del procesamiento de datos.

→ **Learning rate = 0.001 y 0.0001.**

Este es el único valor al que se han usado dos distintos dependiendo del modelo. Esto es debido a que, para modelos más complejos, con unas grandes dimensiones, donde se usan una gran cantidad de neuronas, usar un coeficiente alto dificulta mucho que consiga mejorar por cada iteración. Mientras que para un modelo más simple con el learning rate bajo apenas consigue mejora, por lo que es necesario de una mayor.

La solución se ha realizado probando entre 4 implementaciones distintas de la red neuronal. Una de ellas ha sido creando una arquitectura de red neuronal, definiendo como está compuesta cada capa, el número de capas que tiene en total, la función de activación que emplea, etc. Todo a través de las funciones que proporciona Keras, que facilitan mucho esta tarea. El resto de los modelos han sido de los ya probados y que están publicados en el portal de Keras, llamado Keras Applications, la cual les da soporte. Específicamente los modelos seleccionados para las pruebas son Xception, ResNet121v2

y VGG16. Estos modelos son incluidos como paquetes dentro de la biblioteca de Keras y han sido seleccionados para ello debido a que, han ganado competencias donde se intenta mostrar modelos con los que se consiguen llegar a unos grandes resultados de predicción. El uso de estos ya implica la definición de la arquitectura de la red e incluso unos valores iniciales de los pesos, aunque estos últimos no son necesarios de usar.

La comparación entre cada uno de los modelos mencionados se va a llevar a cabo usando todas las métricas y herramientas mencionadas en la sección anterior de las redes neuronales.

Antes de hacer una comparación de los distintos modelos, se va a realizar una, que demuestre lo expuesto durante las distintas secciones anteriores en cuanto a la segmentación seleccionada y el procesamiento del audio que se va a emplear. Para esta comparación se ha utilizado la misma red neuronal con los mismos hiperparámetros y con el mismo conjunto de datos para todas las pruebas. En concreto la red a usar es la definida, puesto que requiere poco tiempo por época y empleando el conjunto A de datos. Los hiperparámetros usados son los mismos que los comentados antes.

6.2.6 Demostración de las metodologías empleadas

Con el fin de visualizar de un modo simple y claro, una comparación entre los resultados obtenidos por las distintas metodologías expuestas en las secciones anteriores se ha empleado sólo los valores de val_Accuracy y val_Loss, con los cuales se puede tener una idea de su capacidad de realizar una correcta predicción y cómo es de grande el error cuando falla en la predicción.

	Todo con segmentación 2''	Procesamiento de audio solo Mel	Todo con segmentación 3''
Val_Accuracy	0'842	0'864	0'938
Val_Loss	0'336	0'284	0'127

Tabla 8. Valores obtenidos para cada una de las metodologías

	Todo con segmentación 2''	Procesamiento de audio solo Mel	Todo con segmentación 3''
Tiempo medio por época de entrenamiento	336ms	647ms	176ms
Tiempo medio de predicción del modelo final	15ms	20ms	6ms

Tabla 9. Tiempos obtenidos en el entrenamiento y como predictor del modelo

Tras la comparación de los resultados de la tabla anterior se pudo observar que la mejor opción es la que se emplean todas las técnicas de procesamiento de audio y con una segmentación de tres segundos.

Además, se consigue hacer el entrenamiento con un buen promedio del tiempo de entrenamiento y es el modelo que consigue la predicción en mejor tiempo, aunque estos valores tampoco son muy críticos puesto que en todos los casos se consigue un tiempo muy bajo por lo que es despreciable, pero, es interesante comentar por cómo podría influir en caso de contar con un conjunto de datos mayor.

Una vez ya analizado esto y teniendo en cuenta con lo concluido, se van a usar estos parámetros para las distintas implementaciones de los modelos diferentes que se van a comparar.

6.2.7 Modelo definido usando Keras

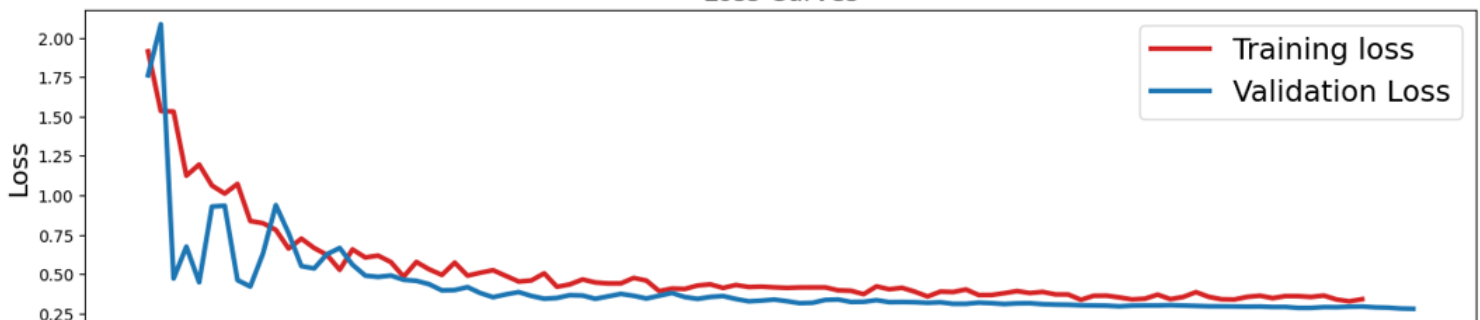
- **Descripción general de la arquitectura**

La arquitectura de la red definida es la siguiente. Está compuesta por 4 capas convolucionales en las que en todas las neuronas que forman cada una se le es aplicada a la salida una función tipo ReLu. También se le aplica un Dropout a la salida obtenida por cada capa es desconectada de la siguiente en mayor o menor medida en función del porcentaje establecido a dicho Dropout. La arquitectura exactamente de la implementación se puede ver en el anexo.

- **Gráficos**

Los siguientes gráficos muestran como ha sido la evolución de los valores de error y la predicción de validación, para cada una de las épocas de entrenamiento. Como se puede observar a partir de la época 25, la evolución tanto del error, como de la precisión, disminuyen y crecen respectivamente de forma lineal con una pendiente mínima. Por lo que, a partir de esa época, se puede dar ya cómo válido el modelo al que se consigue obtener.

Loss Curves



Accuracy Curves

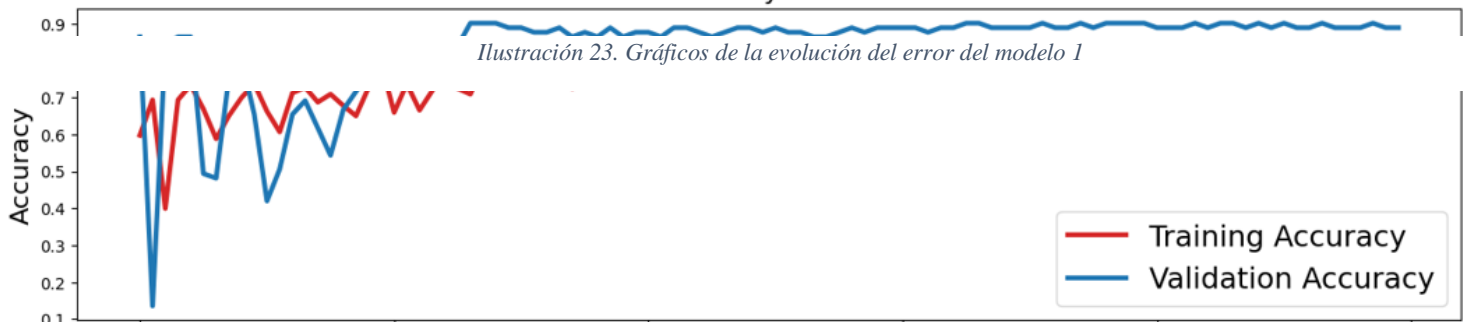


Ilustración 23. Gráficos de la evolución del error del modelo 1

Ilustración 24. Gráfico de la evolución de la precisión del modelo 1

- **Valores de las métricas**

Aplicando las formulas mencionadas a partir de los resultados que se han conseguido tras realizar una predicción sobre el conjunto de datos de testeo. Para cada una de las etiquetas de salida se obtiene lo siguiente:

	Precision	Recall	F1-Score
Abnormal	0.57	0.77	0.65
Normal	0.84	0.67	0.74

Tabla 10. Métricas obtenidas en el modelo 1

- **Matriz de Confusión**

A partir de los resultados obtenido por la predicción hecha por el modelo al conjunto de datos de test, se ha obtenido la siguiente matriz. Observándola se puede concluir que ha conseguido un buen acierto, especialmente detectando positivos. Y en cuanto a fallo ha dado un porcentaje medianamente alto de predicciones de falsos positivos, los cuales son mejores de cometer que un falso positivo como se mencionó al inicio de esta memoria, por lo que se comentaba del riesgo al que podría estar el paciente por no ser atendido a la enfermedad que padece, al creer que no la tiene.

1

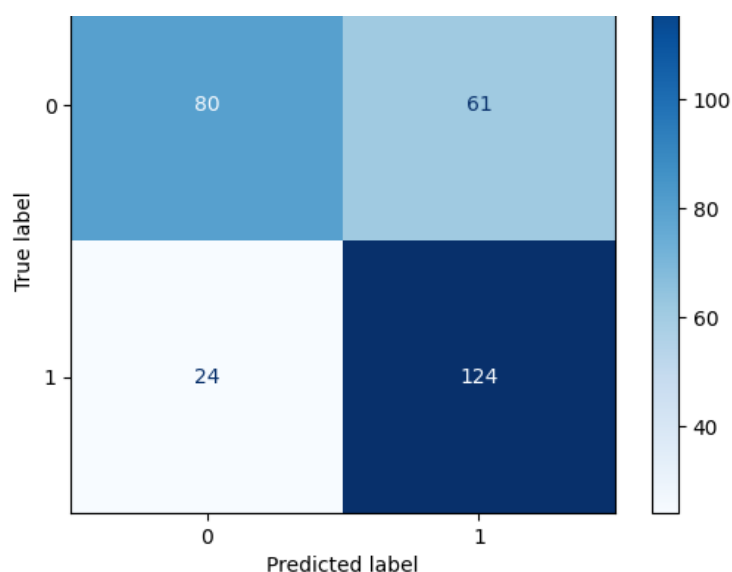


Ilustración 25. Matriz de confusión del modelo 1

- **Tabla de tiempos**

La tabla que viene a continuación muestra los tiempos que se han obtenido durante el proceso de entrenamiento y creación del modelo.

	Tiempos
Entrenamiento	2 segundos por época
Predicción del modelo construido	0.1 segundos
Número de épocas totales	100

Tabla 11. Valores de los tiempos del modelo 1

6.2.8 Modelo VGG16

- **Descripción general de la arquitectura**

Presenta una profundidad como red de 16 niveles con un total de 138'4 millones de parámetros con los que ajustar la construcción del modelo. Debido a esto como es de suponer es modelo que a nivel de almacenamiento es de los más pesados a usar. Y como es de suponer y se mostrará más adelante esto también repercute en el tiempo de entrenamiento para la construcción final del modelo.

- **Gráficos**

A partir del gráfico se puede ver que el error se va reduciendo prácticamente linealmente. Hasta alcanzar un mínimo que a partir de él se dispara y vuelve a ir reduciéndose de nuevo con una pendiente mayor que al inicio pero sin llegar a volver a alcanzar ese mínimo.

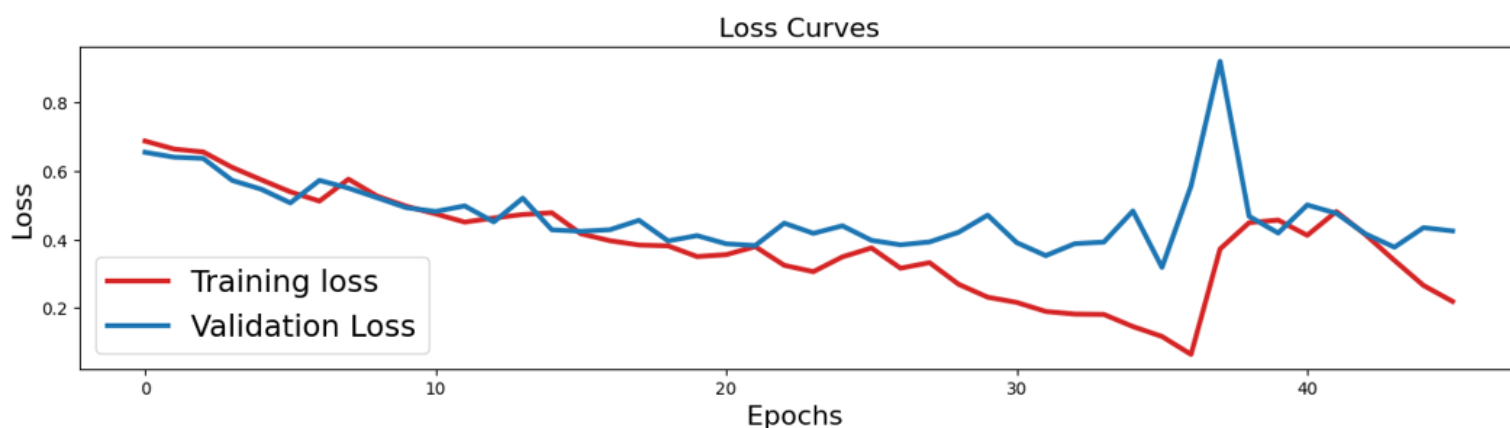


Ilustración 26. Gráfico de la evolución del error del modelo 2

En cuanto a la precisión se consigue que vaya aumentando con algunas escalonadas en su crecimiento, pero al igual que sucedió para el error. Se alcanza un punto máximo que viene seguido de un descenso con una gran pendiente, hasta iniciar a seguir una nueva curva de crecimiento, la cual no alcanza el máximo alcanzado antes.

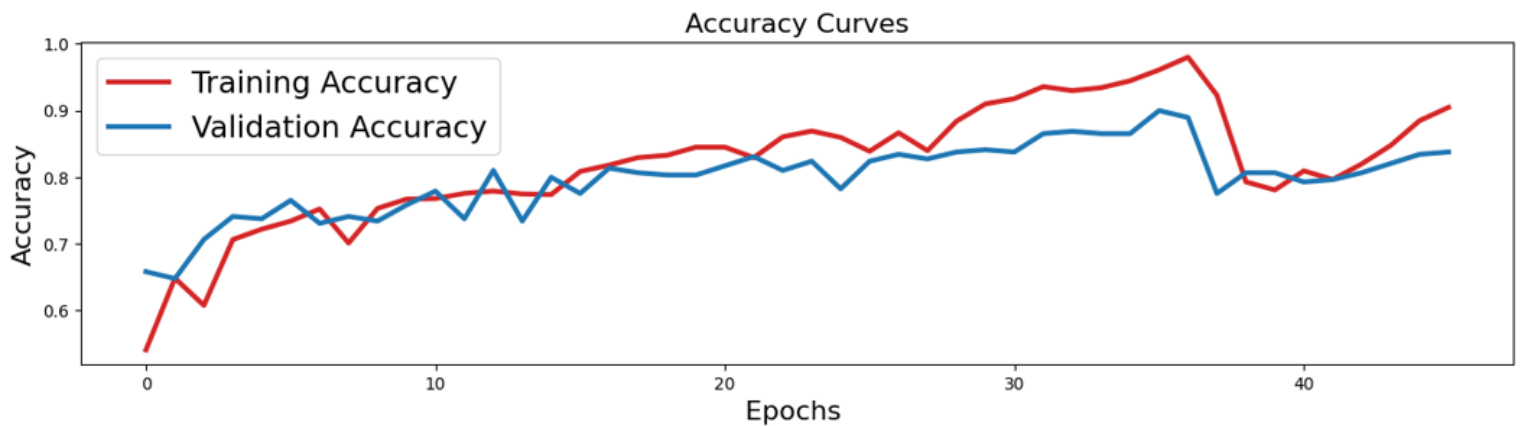


Ilustración 27. Gráfico de la evolución de la precisión del modelo 2

- **Valores de las métricas**

Lo obtenido en cuanto a las métricas aplicando este nuevo modelo es lo siguiente:

	Precision	Recall	F1-Score
Abnormal	0.86	0.93	0.89
Normal	0.94	0.87	0.91

Tabla 12. Métricas obtenidas del modelo 2

- **Matriz de Confusión**

Mirando esta matriz se puede ver muy claramente como consigue un gran porcentaje de acierto y que, al igual que pasaba con el modelo anterior, para los errores más críticos es donde consigue un menor fallo que en este modelo son aún menor que para el visto antes.

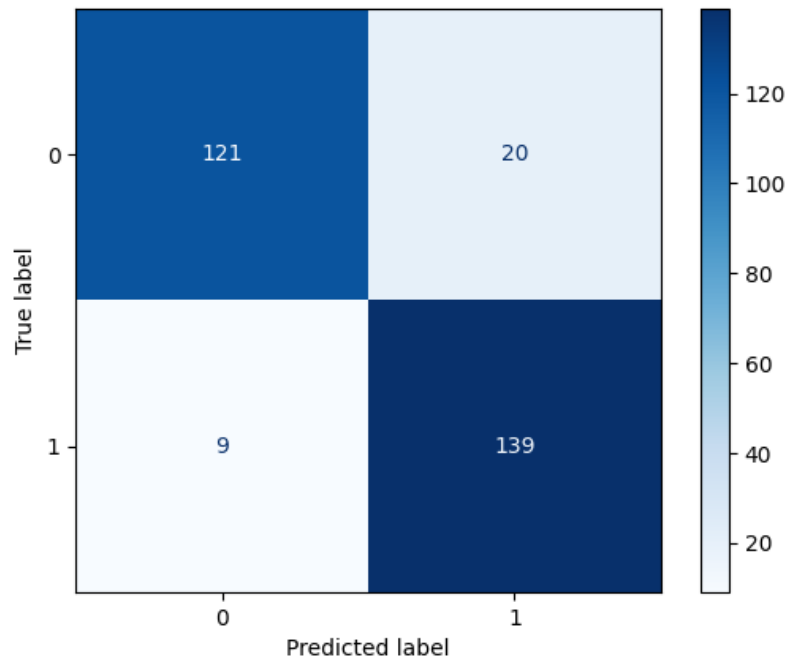


Ilustración 28. Matriz de confusión del modelo 2

- **Tabla de tiempos**

	Tiempos
Entrenamiento	107 segundos por iteración
Predicción del modelo construido	6 segundos
Número de épocas totales	47

Tabla 13. Valores de los tiempos para el modelo 2

6.2.9 Modelo Xception

- **Descripción general de la arquitectura**

Xception presenta un diseño con un número de 89 capas de profundidad, con un total de 22'9 millones de parámetros. Al contar con una mayor profundidad esto puede suponer un incremento en el tiempo de ejecución en cada iteración del entrenamiento. Especialmente para esta arquitectura suele tener un tiempo medio mayor a pesar de no contar con tantos parámetros debido a que realiza unas operaciones internas complejas que penalizan con un mayor tiempo de ejecución.

- **Gráficos**

Lo que se observa en este gráfico es que tras unas pocas épocas donde el error de validación si se está disminuyendo, se inicia a elevar y ya no consigue mejora a lo largo de las distintas épocas. Para el caso de la precisión es muy similar donde se ve que da una sensación de que se queda estancado y no consigue mejora. Esto es debido a que tras unas pocas épocas se inicia a producir overfitting como se ha comenta. Esto es debido a que el modelo al tener una capacidad tan potente consigue modelar a la perfección y tras pocas épocas al conjunto de entrenamiento. Es por ello que, al evaluarlo con el conjunto de test consigue empeorar su resultado.

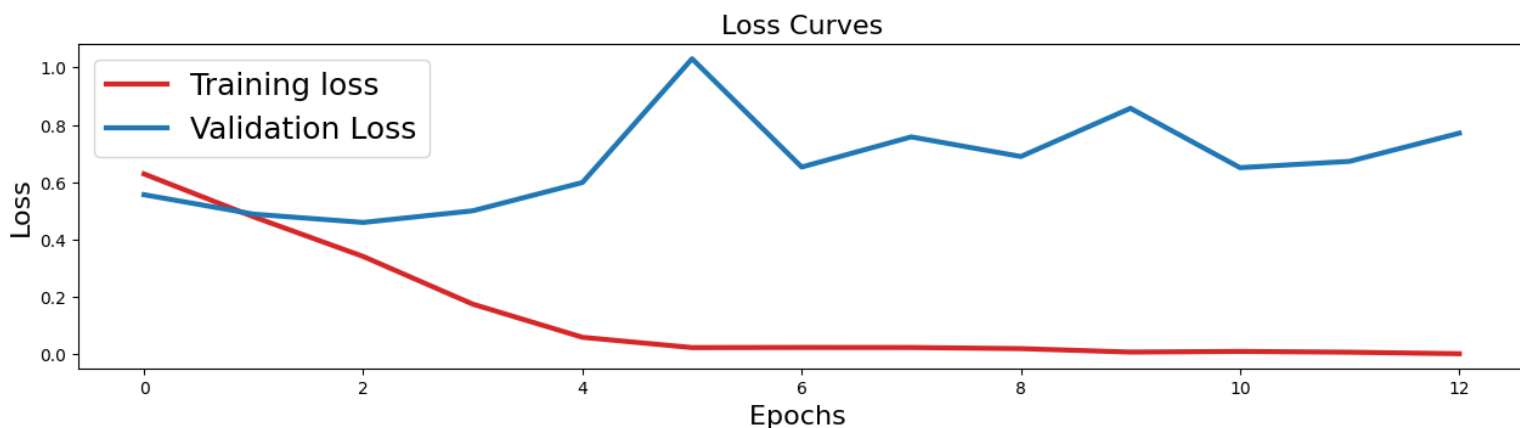


Ilustración 29. Gráfico de la evolución del error del modelo 3

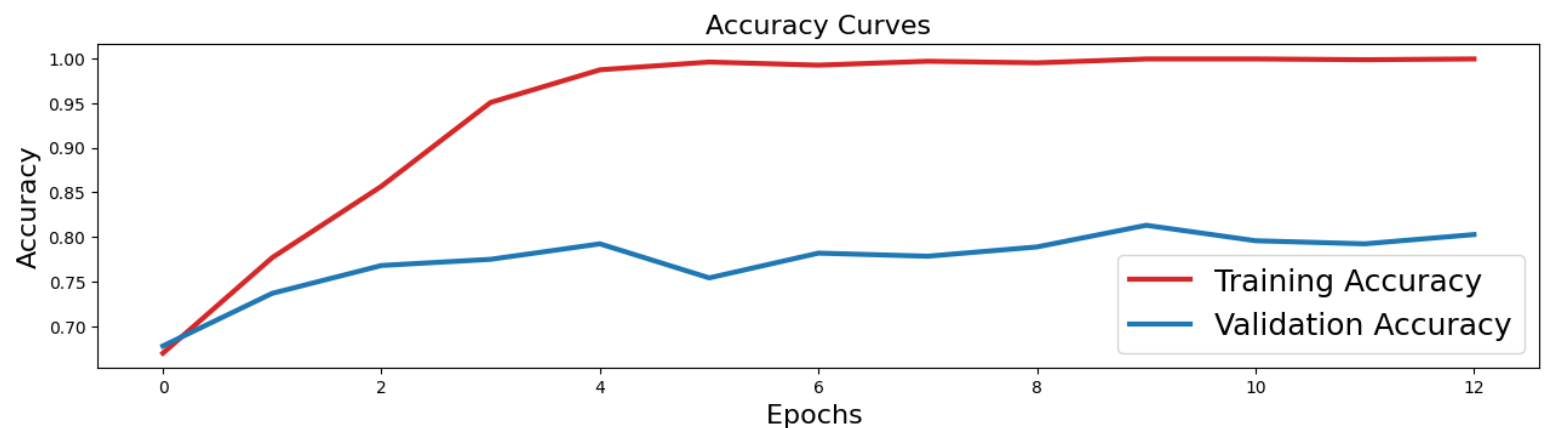


Ilustración 30. Gráfico de la evolución de la precisión del modelo 3

- **Valores de las métricas**

Para este modelo se consiguen los siguientes valores para las distintas métricas:

	Precision	Recall	F1-Score
Abnormal	0.65	0.83	0.73
Normal	0.87	0.72	0.79

Tabla 14. Métricas obtenidas del modelo 3

- **Matriz de Confusión**

Para este modelo se consigue también un buen porcentaje de acierto y al igual que en los anteriores el fallo que más comete es para el caso menos crítico.

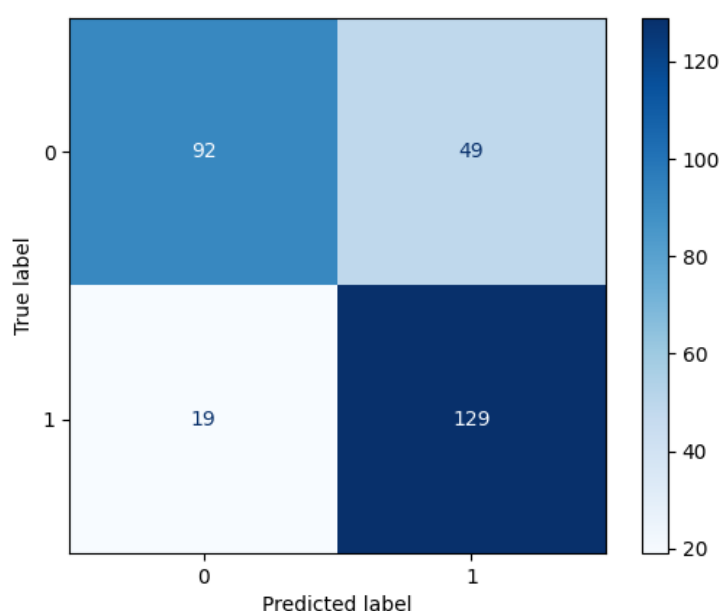


Ilustración 31. Matriz de confusión del modelo 3

- **Tabla de tiempos**

	Tiempos
Entrenamiento	105 segundos por iteración
Predicción del modelo construido	6 segundos
Número de épocas totales	12

Tabla 15. Valores de los tiempos del modelo 3

6.2.10 Modelo RestNet

- **Descripción general de la arquitectura**

La red está conformada por una profundidad de 205 niveles de profundidad y con un total de 44'7 millones de parámetros. Al contar con un nivel de tanta profundidad tiene la problemática que ante crear un modelo que no cuente con un conjunto de datos grandes y con una alta cantidad de parámetros de entrada. Conlleva a que tiende a producirse el efecto de overfitting muy fácilmente.

- **Gráficos**

Para este modelo sucede prácticamente lo mismo que para el modelo anterior esto es debido por la magnitud de su arquitectura que consiguen representar en pocas épocas al conjunto de entrenamiento impidiendo que generalice. En este caso como pequeña variante del anterior se puede que ver que cuando entra en esta fase de estancamiento en el que el error o la precisión de validación empeora y el de entrenamiento mejora. Se produce en un par de épocas el efecto rebote debido a la agresividad en el cambio del error en los ajustes de los pesos.

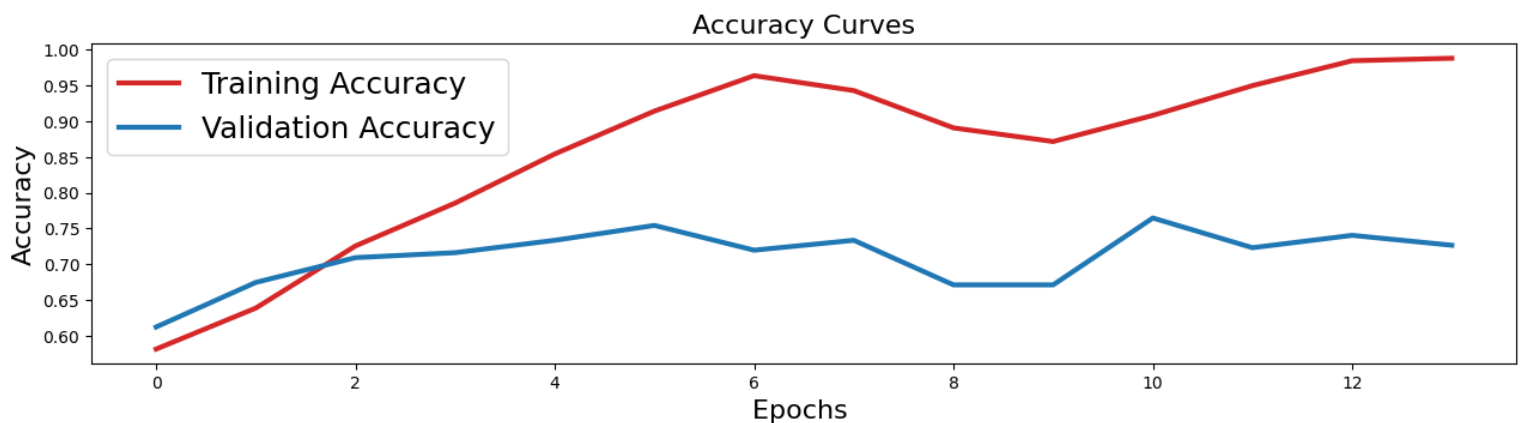


Ilustración 32. Gráfico de la evolución de la precisión del modelo 4

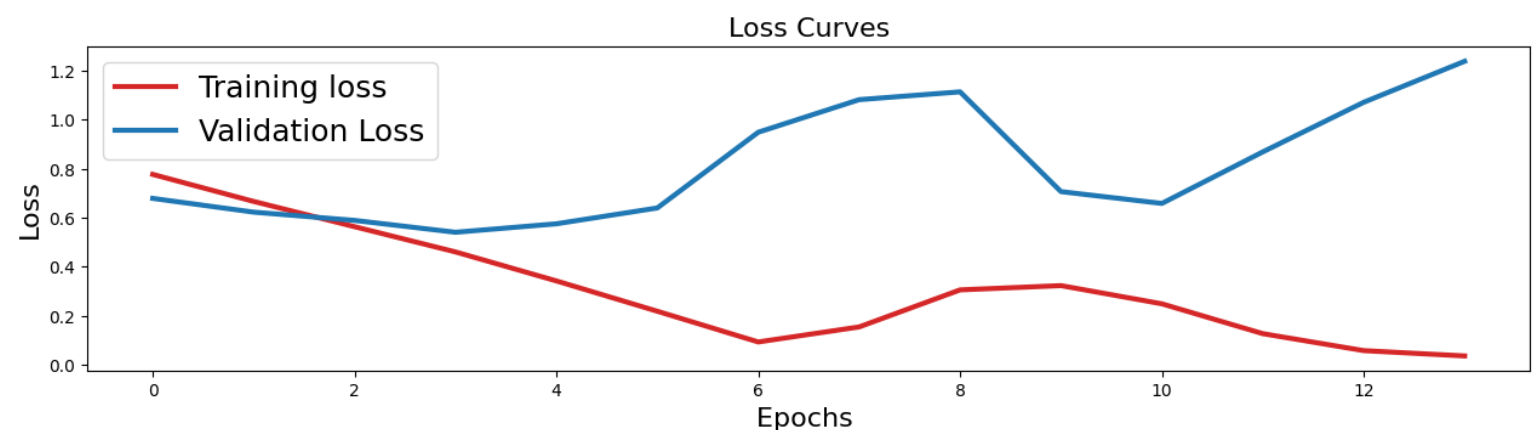


Ilustración 33. Gráfico de la evolución del error del modelo 4

- **Valores de las métricas**

	Precision	Recall	F1-Score
Abnormal	0.72	0.72	0.72
Normal	0.73	0.73	0.73

Tabla 16. Métricas obtenidas del modelo 4

- **Matriz de Confusión**

Mirando la matriz se puede ver como consigue también un buen porcentaje de aciertos, pero que en cuanto a fallo comente ligeramente más que los anteriores y la misma cantidad para ambos tipos de errores.

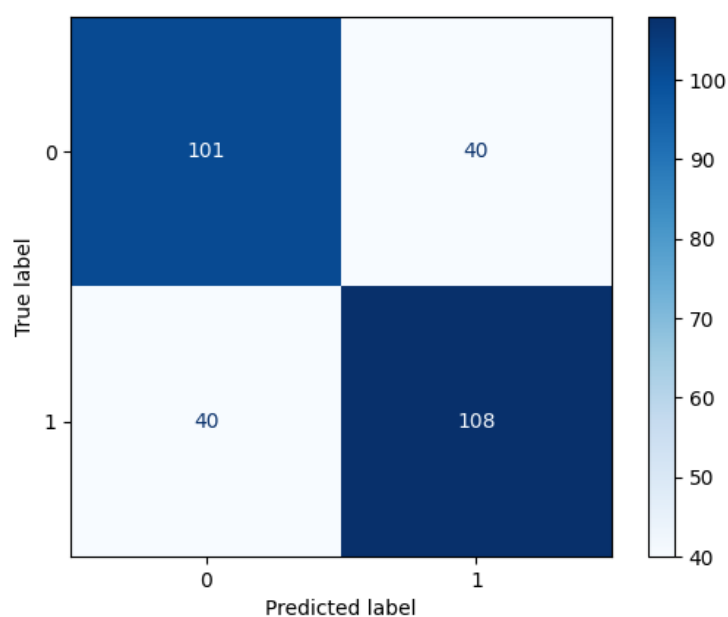


Ilustración 34. Matriz de confusión del modelo 4

- **Tabla de tiempos**

	Tiempos
Entrenamiento	84 segundos por época
Predicción del modelo construido	8 segundos
Número de épocas totales	14

Tabla 17. Valores de los tiempos del modelo 4

6.2.11 Modelo final

Tras haber visto el comportamiento y los resultados que han sido obtenidos por cada uno de los modelos. Podemos decir que el modelo que consigue unos mejores resultados, medidos por la cantidad de predicciones acertadas y que comete el mínimo de los falsos negativos, es el modelo 2 usando la arquitectura VGG16.

A decir también que, aunque sea el modelo que consigue obtener unas mejores predicciones, es el requiere de un mayor tiempo de entrenamiento tanto por época, como por el número de épocas totales realizadas para la construcción del modelo. Pero para este caso es de una mayor importancia tomar como medida el conseguir las mejores predicciones, aunque para ello sea a consta de tener que dedicar un largo tiempo para poder obtener el modelo.

Si es cierto que en algún otro ámbito donde fuese un requisito tener en cuenta una relación entre el tiempo requerido y que el modelo tuviese la mayor eficacia, el modelo que para ello tiene la mejor relación sería el primero. Ya que, consigue unos buenos resultados de predicción y es sin duda y por mucha diferencia el que requiere de un tiempo mucho menor para llevar a cabo el entrenamiento, así como también para realizar las predicciones de una muestra dada.

Esto puede ser de suma importancia y punto a tener en cuenta para las propuestas que serán mencionadas en la sección siguiente de Trabajo a futuro. Ya que, optar por una solución que reduce la fiabilidad, aunque sea mínimamente de los resultados, pero a consta de ser mucho más asequible para poder llevarlo a cabo en sistemas reales. Hace que esta solución también sea para tener muy en cuenta y muy válida como la primera mencionada a mejor solución

Planificación del proyecto

8 PLANIFICACIÓN INICIAL

8.1 DIVISIÓN DE LA PLANIFICACIÓN

Fase 1: Planificación, estudio, investigación

Consiste en conocer información sobre el ámbito del proyecto tanto a nivel de los conceptos, requisitos para su implementación...

Fecha Inicio: 12.09.2022

Fecha Fin:

26.09.2022

A01. Estudio sobre el tema y entorno del proyecto.

- **T0:** Hacer una investigación de la temática en la que el proyecto está destinado.
- **T1:** Comprender los conceptos, definiciones y objetivos a los que va a tratar el proyecto.

A02. Creación y estructuración de la memoria.

- **T0:** Crear la documentación con su respectiva organización de las distintas partes de las que va a constar y con los materiales necesarios a emplear.

A03. Requisitos y necesidades del proyecto.

- **T0:** Comprobar las posibilidades en cuanto a que entorno de desarrollo puede ser el mejor a emplear.
- **T1:** Defender cual es el mejor lenguaje de programación a usar para los objetivos que se pretende alcanzar.
- **T2:** Estudiar todas las librerías, frameworks, apis a poder necesitar durante el desarrollo.

Fase 2: Inicio del proyecto

Lo que se pretende en esta fase es que tras su finalización se tenga todo lo necesario para iniciar el desarrollo del proyecto tanto a nivel de instalación y configuración de las herramientas a emplear.

Fecha Inicio: 27.09.2022

Fecha Fin:

11.10.2022

A04: Instalación y configuración de todas las herramientas

- **T0:** Instalar los entornos de desarrollo, Python, la librería de Tensorflow y anaconda Prompt.
- **T1:** Configurar las herramientas de la tarea anterior para que funcionen de manera correcta conjuntamente.

A05: Inicio de la descripción del proyecto en la memoria

- **T0:** Documentar en base a toda la información recolectada realizada en la fase anterior e iniciar a argumentarla en la memoria con la estructura planteada.

A06: Comprobar la validez de los datos a usar para las pruebas

- **T0:** Buscar distintas bases de datos de audios de corazones.
- **T1:** Verificar que la base de datos está compuesta con las características necesarias y que los datos sean fiables y válidos para el sistema.

A07: Realizar la planificación del proyecto

- **T0:** Crear un itinerario temporal de la estructuración del proyecto que se pretende seguir.
- **T1:** Crear una planificación de los costos

Fase 3: Implementación y pruebas

En esta se va a iniciar a realizar diversas implementaciones de nuestra solución donde en cada una se irá probando con distintas modificaciones de las distintas partes de las que consta el desarrollo. Con el fin de hacer un recolecta de resultados a partir de cada solución dada.

Fecha Inicio: 12.10.2022

Fecha Fin:

20.11.2022

A08: Prueba inicial de la primera implementación.

- **T0:** Desarrollar un modo para manejar los datos que será usado a lo largo de todas las pruebas.
- **T1:** Decidir entre hacer una segmentación al audio o usarlo completo.
- **T2:** Escoger entre uno de los procesamientos de audio a través de alguna librería.
- **T3:** Usar una de entre las redes neuronales para obtener los resultados de la primera prueba.

A09: Empezar a realizar distintas modificaciones y pruebas para obtener distintos resultados.

- **T0:** A partir de la implementación inicial, hacer distintas pruebas modificando el modo de procesamiento de audio.
- **T1:** Probar también con distintas implementaciones de diversos modelos de redes neuronales.

A10: Seguir documentando las partes de desarrollo del proyecto

- **T0:** Dar un esquema general del diseño del proyecto.
- **T1:** Explicar en profundidad las herramientas usadas.
- **T2:** Exponer como ha sido el desarrollo.

Fase 4: Análisis

Tras los resultados obtenidos en la fase anterior, se va a estudiar y analizar cada uno de los obtenidos por las distintas soluciones. Donde se pretende evaluar la validez y fiabilidad que tendría el sistema final aplicando dicha implementación.

Fecha Inicio: 21.11.2022

Fecha Fin:

28.11.2022

A11: Analizar los resultados que se van obteniendo a lo largo de las distintas pruebas.

- **T0:** Recoger todas las versiones de las distintas implementaciones dadas con sus respectivos resultados obtenidos.
- **T1:** Crear un gráfico con la mejora que se ha obtenido por cada iteración de las etapas de la red para cada solución.

A12: Comprobar y revisar lo obtenido de los resultados de las pruebas

- **T0:** Estudiar cada resultado y ver el porcentaje de acierto que se ha conseguido.
- **T1:** Hacer una comparación de cada implementación con su evaluación.
- **T2:** Analizar cuál de todas las implementaciones ofrece una mejor solución.

Fase 5: Revisión, corrección y exposición final

Para concluir se hará la correspondiente documentación de la fase anterior y se revisará toda la presentación de la memoria a entregar añadiendo las partes a faltar, corrigiendo cualquier posible error y estilizando en la medida de lo posible hasta tener la versión final.

Fecha Inicio: 29.11.2022

Fecha Fin:

05.12.2022

A13: Exponer en la memoria los resultados de la investigación obtenidos tras todas las pruebas.

- **T0:** Importar a la documentación gráficos y tablas mostrando una comparación precisa y legible para una comprensión fácil de las diferencias entre cada solución.
- **T1:** Explicar de forma argumentada como ha sido desarrollada cada una de las implementaciones y defender el resultado que se obtiene a partir de ella.

A14: Realizar una corrección y revisión de toda la memoria.

- **T0:** Observar que concuerdan los índices, las imágenes, las tablas, etc.
- **T1:** Comprobar que es completo y correcto cada uno de los distintos apartados.
- **T2:** Corregir alguna cosa en caso de encontrar algún fallo.
- **T3:** Realizar las modificaciones que fuesen necesaria para estar en el formato correcto antes de la entrega.

8.2 DIAGRAMA DE GANTT

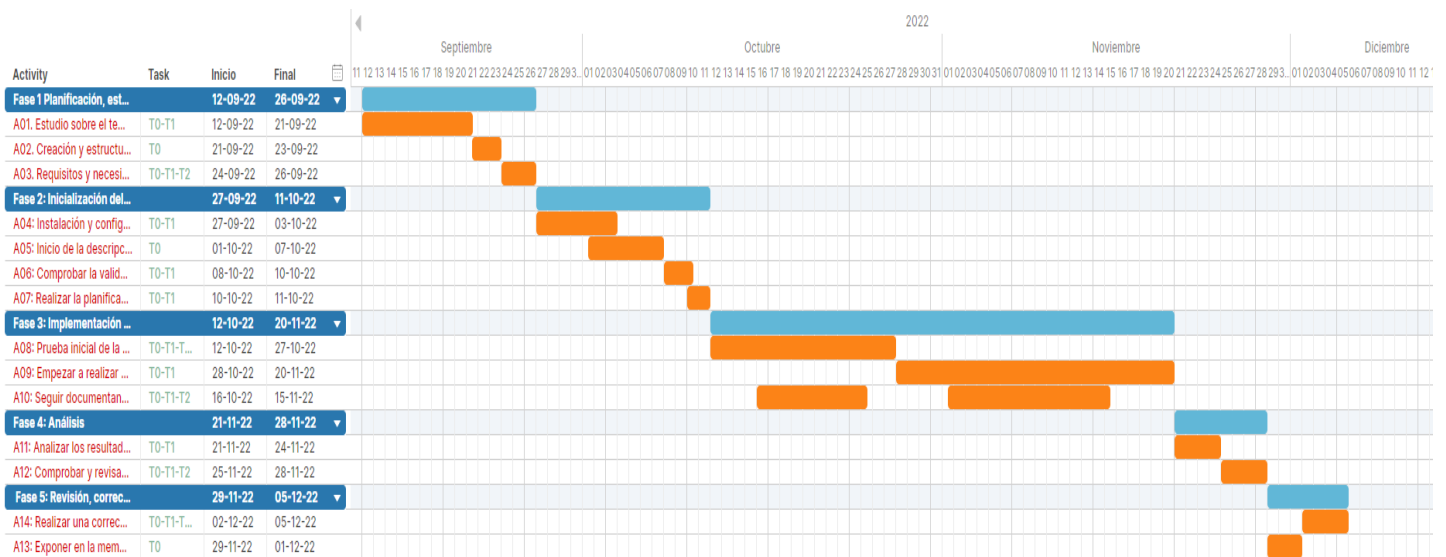
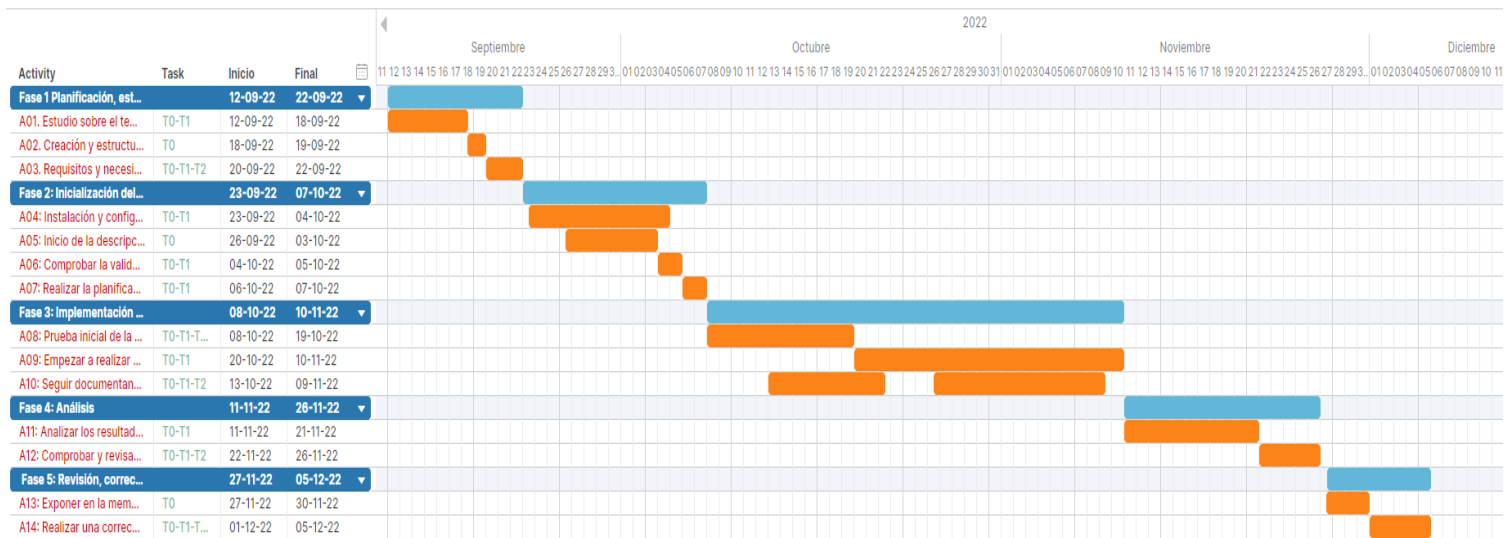


Figura 32. Diagrama de Gantt

9 PLANIFICACIÓN TEMPORAL FINAL

Con respecto a la planificación final, se compone de las mismas fases y tareas que las descritas en la planificación inicial, pero habiendo sufrido algunos cambios en cuanto a los tiempos finalmente dedicados para cada una de ellas con respecto a la que se pretendía realizar, debido a complicaciones surgidas a lo largo del desarrollo en algunas de las tareas a realizar, pero acabando y empezando sobre la fecha marcada. Gracias a hacer una mayor distribución de las horas donde conseguir antes de lo esperado algunas de las actividades, compensando aquellas que han supuesto la dedicación de más tiempo debido esos fallos o dificultades encontradas durante su desarrollo.



10 PLANIFICACIÓN DE COSTES INICIALES

Las tablas siguientes muestran las estimaciones de los costes a lo largo del desarrollo, los cuales pueden cambiar a lo largo del tiempo difiriendo con el coste del prototipo final.

Fase	Actividad	Coste	Fecha
Fase 1	A01 – A02 – A03	16%	12/09/2022
Fase 2	A04 – A05 – A06 – A07	18%	27/09/2022
Fase 3	A08 – A09 – A10	45%	12/10/2022
Fase 4	A11 – A12	10%	21/11/2022
Fase 5	A13 – A14	10%	28/11/2022

Tabla 18. Distribución de los costes iniciales

Producto	Precio	Cantidad	Subtotal
Estudio inicial	50 €	1	50 €
Herramientas	0 €	1	0 €
Pruebas	10 €	1	10 €

Tabla 19. Gastos en materiales iniciales

Gastos en personal

El desarrollo del prototipo se conforma por un equipo de un único integrante, el cual a lo largo del proyecto hará la función de distintos roles.

- **Jefe de proyecto:** Coordina las tareas y fases del proyecto, siendo el encargado de hacer que se siga con la mayor fiabilidad posible el itinerario marcado por la planificación inicial. Haciendo también en la mayor medida de lo posible que los gastos se ajusten lo máximo posible o en tal caso se disminuyan de lo estimado.
- **Desarrollador:** Realiza las implementaciones a las distintas soluciones que se irá realizando a lo largo del proyecto en todas las partes que lo constituye.
- **Investigador del producto (Marketing):** Es el que hará un estudio sobre la involucración del producto dentro de entorno, analizando a su vez lo que existe similar en el mercado, el público al que va dirigido, etc.
- **Analista:** Se encargará de evaluar los resultados obtenidos tras los diversos desarrollos de la solución, así como determinar cuál es la más interesante para el producto final.

Para hacer una estimación del coste se va a hacer un promedio del sueldo que supondría para cada uno de los diferentes perfiles. Asumiendo como gasto la duración total que ha conllevado la finalización del proyecto.

La siguiente tabla muestra un salario medio de cada uno de los perfiles descriptos previamente. A partir de estos datos se va a realizar el cálculo promedio del coste salarial.

Empleado	Jefe de proyecto	Desarrollador ingeniero junior	Investigador de mercado	Científico de datos
Salario medio/mensual Bruto	3.600€	2.160€	2.880€	3.360€
Salario medio/mensual Neto	3.000€	1.800€	2.400€	2.800€

Tabla 20. Precio medio por profesión

A partir de estos podemos estimar que el salario medio para el profesional en la realización del proyecto es de unos 2.500€ netos. Brutos serían unos 3.000€.

Por lo tanto, estimando que la media de horas de trabajo de un mes es de unas 160 horas. Podemos calcular a partir del salario anterior, que el salario por hora sería de unos 15'60€ netos y 18'72€ brutos.

Suponiendo que la duración ha sido alrededor de unos 3 meses, de los cuales invertidas unas 112 horas en cada uno de ellos. Se obtiene como horas totales empleadas en la finalización del proyecto, 336 horas.

Calculando a través del cómputo de números de horas totales invertidas en la realización del proyecto por el precio medio a la hora, sería $315 \text{ horas} * 15'60€ = 5.226€$ y $315 \text{ horas} * 18'72€ = 6.271'2€$

Así como gasto total predecible para el desarrollo del proyecto sería del coste personal calculado anterior sumado a los gastos en materiales que darían un total de $6.271'2 + 60 = 6.331'2€$

10 PLANIFICACIÓN DE COSTES FINAL

Tras la ejecución del proyecto se ha visto ligeramente modificado el precio final del desarrollo. Se ha conseguido reducir levemente el precio final de la realización debido que no se ha invertido en ningún tipo de gasto en materiales y el tiempo de desarrollo se ha visto reducido. El tiempo finalmente empleado para finalizar el proyecto ha sido de 310 horas siendo 26 menos de las que estaban predichas. Obteniendo como gasto personal real del proyecto $310 * 18'72€ = 5.803'2€$. Siendo este el gasto final.

11 ESTUDIO DE MERCADO

Se va a hacer un estudio completo del mercado para saber cuál sería el público más adecuado para promocionar el producto y que como de bien se desenvolvería en él.

Clientes potenciales

Como cliente está destinado especialmente a los centros sanitarios para su uso por parte del personal médico. También podría ser empleado por los propios pacientes y recibir un primer diagnóstico previo a la atención médica. En cuanto al grupo de edad más propenso a su uso, va directamente proporcional con el crecimiento de la edad. Así pues, las personas de entre 64 y 84 años sería el público objetivo al que iría principalmente destinado el producto. Ya que a esta edad crece significativamente la posibilidad de padecer alguna enfermedad y por ende la preocupación de poder detectar lo más rápido posible cualquier patología en caso de padecer alguna. Aunque a tener en cuenta aun así en cualquier grupo de edad.

Según la fundación del corazón española, en torno a 42% de la población española sufre de alguna patología cardiovascular de riesgo.

Si nos centramos a nivel nacional. Como se puede observar en el siguiente gráfico, en torno al 20% de la población española tienen más de 64 años.

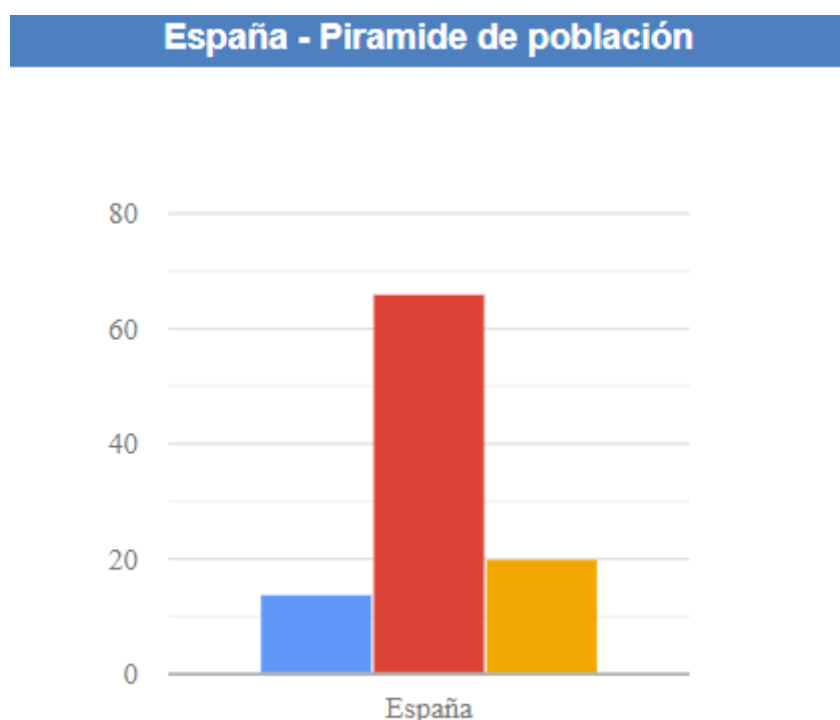


Figura 33. Gráfica de la población en España

(Datos demográficos en España)

Plan de comercialización

Tras ya conocer a los clientes a los que está destinado el producto, se va a diseñar un plan de amortización del producto.

El producto para comercializar se propone como la venta de un servicio. Consistiría en tener la implementación del sistema propuesto alojado en la nube. La compra del servicio se pondría a modo de suscripción, con distintos planes de esta. Los distintos planes consistirían en función de la cantidad de análisis que te ofrece cada una, pudiendo ser una distribución de la misma del siguiente modo; plan básico que te ofrece hacer 100 análisis, el cual puede ser conveniente para pequeños centros sanitarios como ambulatorios, un plan medio que ofrece 1.000 análisis, perfecto para centros sanitarios con una mayor atención a pacientes como podría ser un pequeño hospital y por último, un plan premium que ofreciese 100.000 análisis, pensado para aquellos hospitales o ciudades sanitarias con grandes capacidades que alberga la atención de un gran número de pacientes. Realizándose la renovación conforme el centro requiera de una mayor cantidad de muestras. Otra opción es la de hacer un pago único para realizar el análisis de una sola muestra, la cual podría ser útil para aquellas personas que requiriese de conocer su estado previo a la consulta con el médico.

El precio podría ser para cada una de las suscripciones; 100€ para el plan básico, 800€ para el plan medio y 5.000€ para el plan premium. En cuanto a realizar una prueba individual podría tener el precio de 1€.

Conclusiones

Con este trabajo de fin de grado se ha mostrado una solución a la problemática de detectar una enfermedad del corazón, a través de los sonidos que producen sus latidos, mediante el uso de redes neuronales.

A nivel general se puede estar satisfecho con el resultado final que se ha conseguido en el proyecto, ya que, después de lo seleccionado a emplear para cada una de las distintas secciones que se han ido mencionando a lo largo de toda la memoria. Se puede decir que se han conseguido cumplir todos los objetivos que estaban dispuestos a satisfacerse con la finalización de este. Además, en gran medida ha sido posible ceñirse a la planificación del proyecto dada, teniendo que sufrir de algunas modificaciones debido al retraso de la ejecución de algunas fases por ciertos imprevistos encontradas a lo largo del desarrollo. En cuanto al coste planteado, también ha sido posible llevarlo a cabo sin añadir costes extras por motivos imprevistos.

Tras la realización del trabajo se ha puesto en práctica muchos de los conocimientos que han sido adquiridos a lo largo de los estudios del grado. El desarrollo de este trabajo aúna conocimientos de muchas disciplinas distintas, además de servir como un acercamiento a un proyecto de investigación. Desde el punto de vista académico, ha servido para aplicar dichos conocimientos adquiridos, desde; el proceso de planificación y gestión de un proyecto, conceptos relacionados con materias vistas de algoritmia, inteligencia artificial, etc. E incluso para reforzar y ampliar conocimientos en los que no habían sido tratados en profundidad como es el caso del procesamiento de señales digitales. Así también, descubrir como aplicar estas herramientas aprendidas en entornos fuera del ámbito de estudio del grado, como es el de la salud, para el cual me ha servido para aprender muchos conocimientos y conceptos relacionados con esta especialidad.

La realización de este trabajo ha estado muy motivada por la capacidad de poder aplicar todo lo aprendido, para poder resolver una problemática que ayuda a la mejora de la salud de las personas. Así con esta motivación, ha sido posible dar forma a todos estos conceptos adquiridos para poder ser usados en la creación de un primer prototipo de la construcción de un sistema que podría resultar de una gran utilidad en el futuro en el ámbito sanitario.

Trabajo Futuro

Como propuesta a futuro y con el fin de la posibilidad de llevar lo presentado con este trabajo a la creación de un producto real en el mercado. Sería conveniente en profundizar y mejorar ciertos aspectos a partir de lo implementado y mencionado durante este proyecto.

Uno de estos aspectos sería la posibilidad de conseguir crear un modelo con mejores resultados. Con el que obtener una mayor fiabilidad de predicción, ya que, es muy importante que el error sea lo mínimo posible debido a que estos fallos, mencionados al inicio de este trabajo, suponen de un riesgo importante para la salud de las personas. Para poder mejorar esto sería conveniente de contar con un equipo más potentes capaz de ejecutar modelos más complejos, pero especialmente contar con una gran cantidad de datos para poder detectar una mayor variedad de casos diversos. Así obteniendo un modelo más completo y con un reconocimiento de patrones muy generalizado.

A partir de lo anterior, también sería muy interesante modificar el modelo para que fuese capaz de tener más clases de salida como predicción, donde, en lugar de clasificar mencionando si dicho paciente sufre o no de una enfermedad cardiaca, especificar que enfermedad exactamente es la que está sufriendo. Para esto nuevamente sería conveniente partir de una gran cantidad de datos etiquetados correctamente y con una distribución uniforme para poder identificar lo más correctamente posible cada una de las distintas salidas.


Otro de los puntos importantes sería crear un sistema, ya sea a modo de servicio, el cual estuviese integrado en el sistema sanitario con el que los médicos pudiesen usarlo. Para que, a partir de un registro de audio de los latidos del corazón de algún paciente, el sistema mostrase como resultado la enfermedad, o, en caso contrario etiquetarlo como sano. Además, que este sistema utilizase estas nuevas muestras para almacenarlas en una nueva base de datos con la que seguir mejorando el modelo y también puede ser útil para tener un registro que el médico pudiese usar a modo de consulta de los análisis de un paciente.

A parte del sistema anterior, a modo independiente o como complemento de este último. Diseñar otro sistema, pero hardware, que estuviese integrado a un estetoscopio para usarlo como entrada de audio al sistema y a partir de ello, que mostrase la predicción directamente a través de una interfaz gráfica como un display o por medio de una web o aplicación móvil, que mostrase los resultados realizados. Esto permitiría también la posibilidad de tener este aparato, para poderlo usar a modo personal, sin la necesidad de tener que ir a un centro sanitario, conociendo así, un primer detalle de la salud cardiaca del paciente.

Bibliografía

- Agarwal, Manan. [2019] 2022. «Heart Anomaly Detection by Analysing Stethoscope sounds using Deep Learning».
- Redes Neuronales. 2021a. «¿Qué son las redes neuronales?» (<https://www.ibm.com/ar-es/cloud/learn/neural-networks>).
- Redes Neuronales. 2021b. «Redes neuronales: definición, tipos, beneficios y casos de éxito». *Tomedes*. (<https://es.tomedes.com/blog-de-traduccion/redes-neuronales-definicion-tipos-beneficios-casos-de-exito>).
- Redes Neuronales. 2021c. «Redes Neuronales y Deep Learning. Capítulo 2: La neurona». *Future Space S.A.* (<https://www.futurespace.es/redes-neuronales-y-deep-learning-capitulo-2-la-neurona/>).
- Python. s. f.-a. «🐍 Lenguaje Python 【Características principales y Frameworks】». (<https://lenguajesdeprogramacion.net/python/>).
- Ecocardiograma. s. f.-b. «Ecocardiograma - Fundación Española del Corazón». (<https://fundaciondelcorazon.com/informacion-para-pacientes/metodos-diagnosticos/ecocardiograma.html>).
- Estadísticas enfermedades cardíacas. s. f.-c. «Enfermedades cardiovasculares: prevalencia por edad 2018». *Statista*. (<https://es.statista.com/estadisticas/576912/prevalencia-de-las-enfermedades-cardiovasculares-por-grupo-de-edad-espana/>).
- Pirámide de población. s. f.-d. «España - Piramide de población 2021 | Datosmacro.com». (<https://datosmacro.expansion.com/demografia/estructura-poblacion/espana>).
- Estetoscopio. s. f.-e. «ESTETOSCOPIO | Como funciona, partes, tipos y usos del estetoscopio». (<https://como-funciona.co/un-estetoscopio/>).
- Transformada de Fourier. s. f.-f. «FFT». (<https://www.nti-audio.com/es/servicio/conocimientos/transformacion-rapida-de-fourier-fft>).
- Mel. s. f.-g. «Figure 3: The Block Diagram for Extracting Mel Frequency Cepstral...» *ResearchGate*. Recuperado 5 de diciembre de 2022 (https://www.researchgate.net/figure/The-Block-Diagram-for-extracting-Mel-Frequency-Cepstral-Coefficients-MFCC_fig2_335879082).
- Fonendoscopio electrónico. s. f.-h. «Fonendoscopio electrónico/digital - SEPEAP». Recuperado 5 de diciembre de 2022 (<https://sepeap.org/fonendoscopio-electronicodigital/>).

Keras. s. f.-i. «Qué es Keras - PiperLab, Big data & Data science». Recuperado 5 de diciembre de 2022 (<https://piperlab.es/glosario-de-big-data/keras/>).

TF. s. f.-j. «¿Qué es TensorFlow? ¿Cómo funciona? -  Aprende IA». Recuperado 5 de diciembre de 2022 (<https://aprendeia.com/que-es-tensorflow-como-funciona/>).

Fonendo, Don. 2021. «Fonendoscopios para profesionales con pérdida de audición». Recuperado 5 de diciembre de 2022 (<https://donfonendo.com/fonendoscopios-para-medicos-con-problemas-auditivos/>).

Rodriguez, Nogueiras, y Hedo Garcia. s. f. «Diseño e implementación en Python de un sistema de identificación de segmentos musicales». 66.

Solutions, Exsilio. 2016. «Accuracy, Precision, Recall & F1 Score: Interpretation of Performance Measures». *Exsilio Blog*. Recuperado 5 de diciembre de 2022 (<https://blog.exsilio.com/all/accuracy-precision-recall-f1-score-interpretation-of-performance-measures/>).

Anexo

Para la construcción de las tablas de los tiempos y de los resultados de las métricas, además de los gráficos de la evolución del erro y la precisión durante la ejecución del entrenamiento, expuestas en la sección de la anterior. Se ha basado a partir de los resultados que se muestran en las siguientes imágenes.

Las imágenes van a estar mostradas para todos los modelos en el siguiente orden; primero la imagen de las métricas, segunda la imagen de los tiempos y resultados de las primeras 10 épocas de entrenamiento y por último la arquitectura del modelo con todas sus características

Modelo 1

```
Test loss: 0.5710439085960388
Test accuracy: 0.7058823704719543
10/10 [=====] - 0s 8ms/step
```

	precision	recall	f1-score	support
abnormal	0.57	0.77	0.65	104
normal	0.84	0.67	0.74	185
accuracy			0.71	289
macro avg	0.70	0.72	0.70	289
weighted avg	0.74	0.71	0.71	289

Ilustración 35. Resultados de las métricas modelo 1

```
Epoch 1/100
9/9 [=====] - 3s 245ms/step - loss: 10.7495 - accuracy: 0.4792 - val_loss: 2.11
97 - val_accuracy: 0.5398
Epoch 2/100
9/9 [=====] - 2s 181ms/step - loss: 5.8921 - accuracy: 0.5295 - val_loss: 4.027
4 - val_accuracy: 0.5121
Epoch 3/100
9/9 [=====] - 2s 204ms/step - loss: 5.3196 - accuracy: 0.5278 - val_loss: 1.987
5 - val_accuracy: 0.5640
Epoch 4/100
9/9 [=====] - 2s 190ms/step - loss: 4.5911 - accuracy: 0.5217 - val_loss: 1.489
0 - val_accuracy: 0.5986
Epoch 5/100
9/9 [=====] - 2s 169ms/step - loss: 4.4557 - accuracy: 0.4983 - val_loss: 2.078
8 - val_accuracy: 0.5744
Epoch 6/100
9/9 [=====] - 1s 155ms/step - loss: 4.7250 - accuracy: 0.5095 - val_loss: 1.767
3 - val_accuracy: 0.5952
Epoch 7/100
9/9 [=====] - 1s 162ms/step - loss: 3.7596 - accuracy: 0.5556 - val_loss: 1.522
2 - val_accuracy: 0.6505
Epoch 8/100
9/9 [=====] - 1s 149ms/step - loss: 3.5560 - accuracy: 0.5495 - val_loss: 1.620
3 - val_accuracy: 0.6436
Epoch 9/100
9/9 [=====] - 1s 149ms/step - loss: 3.4292 - accuracy: 0.5668 - val_loss: 1.568
2 - val_accuracy: 0.6471
Epoch 10/100
9/9 [=====] - 2s 201ms/step - loss: 3.2800 - accuracy: 0.5616 - val_loss: 1.454
7 - val_accuracy: 0.6505
```

Ilustración 36. Tiempos durante el entrenamiento modelo 1

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 39, 129, 16)	80
max_pooling2d (MaxPooling2D)	(None, 19, 64, 16)	0
dropout (Dropout)	(None, 19, 64, 16)	0
conv2d_1 (Conv2D)	(None, 18, 63, 32)	2080
max_pooling2d_1 (MaxPooling2D)	(None, 9, 31, 32)	0
dropout_1 (Dropout)	(None, 9, 31, 32)	0
conv2d_2 (Conv2D)	(None, 8, 30, 64)	8256
max_pooling2d_2 (MaxPooling2D)	(None, 4, 15, 64)	0
dropout_2 (Dropout)	(None, 4, 15, 64)	0
conv2d_3 (Conv2D)	(None, 3, 14, 128)	32896
max_pooling2d_3 (MaxPooling2D)	(None, 1, 7, 128)	0
dropout_3 (Dropout)	(None, 1, 7, 128)	0
global_average_pooling2d (GlobalAveragePooling2D)	(None, 128)	0
dense (Dense)	(None, 2)	258
Total params: 43,570		
Trainable params: 43,570		
Non-trainable params: 0		

Ilustración 37. Arquitectura de la red del modelo 1

Modelo 2

```

Test loss: 0.31849807500839233
Test accuracy: 0.899653971195221
10/10 [=====] - 6s 578ms/step

```

	precision	recall	f1-score	support
abnormal	0.86	0.93	0.89	130
normal	0.94	0.87	0.91	159
accuracy			0.90	289
macro avg	0.90	0.90	0.90	289
weighted avg	0.90	0.90	0.90	289

Ilustración 38. Resultados de las métricas del modelo 2


```

Epoch 1/100
9/9 [=====] - 109s 12s/step - loss: 0.6874 - accuracy: 0.5399 - val_loss: 0.654
6 - val_accuracy: 0.6574
Epoch 2/100
9/9 [=====] - 113s 13s/step - loss: 0.6641 - accuracy: 0.6476 - val_loss: 0.640
3 - val_accuracy: 0.6471
Epoch 3/100
9/9 [=====] - 130s 15s/step - loss: 0.6556 - accuracy: 0.6068 - val_loss: 0.636
8 - val_accuracy: 0.7059
Epoch 4/100
9/9 [=====] - 107s 12s/step - loss: 0.6104 - accuracy: 0.7057 - val_loss: 0.572
5 - val_accuracy: 0.7405
Epoch 5/100
9/9 [=====] - 102s 11s/step - loss: 0.5743 - accuracy: 0.7214 - val_loss: 0.546
8 - val_accuracy: 0.7370
Epoch 6/100
9/9 [=====] - 111s 13s/step - loss: 0.5391 - accuracy: 0.7335 - val_loss: 0.506
8 - val_accuracy: 0.7647
Epoch 7/100
9/9 [=====] - 112s 12s/step - loss: 0.5121 - accuracy: 0.7517 - val_loss: 0.572
6 - val_accuracy: 0.7301
Epoch 8/100
9/9 [=====] - 100s 11s/step - loss: 0.5762 - accuracy: 0.7005 - val_loss: 0.550
1 - val_accuracy: 0.7405
Epoch 9/100
9/9 [=====] - 100s 11s/step - loss: 0.5261 - accuracy: 0.7526 - val_loss: 0.521
5 - val_accuracy: 0.7336
Epoch 10/100
9/9 [=====] - 105s 12s/step - loss: 0.4969 - accuracy: 0.7665 - val_loss: 0.492
6 - val_accuracy: 0.7578

```

Ilustración 39. Tiempos durante el entrenamiento del modelo 2

Layer (type)	Output Shape	Param #
=====		
input_2 (InputLayer)	[(None, 40, 130, 1)]	0
vgg16 (Functional)	(None, 1, 4, 512)	14713536
global_average_pooling2d (GlobalAveragePooling2D)	(None, 512)	0
dropout (Dropout)	(None, 512)	0
dense_1 (Dense)	(None, 300)	153900
dropout_1 (Dropout)	(None, 300)	0
dense (Dense)	(None, 2)	602
=====		
Total params: 14,868,038		
Trainable params: 14,868,038		
Non-trainable params: 0		

Ilustración 40. Arquitectura del modelo 2

Modelo 3

```
Test loss: 0.47466713190078735
Test accuracy: 0.7647058963775635
10/10 [=====] - 6s 495ms/step
      precision    recall  f1-score   support

 abnormal      0.65      0.83      0.73      111
   normal      0.87      0.72      0.79      178

 accuracy              0.76      289
 macro avg      0.76      0.78      0.76      289
weighted avg      0.79      0.76      0.77      289
```

Ilustración 41. Resultados de las métricas del modelo 3

```
Epoch 1/100
9/9 [=====] - 146s 15s/step - loss: 0.6291 - accuracy: 0.6701 - val_loss: 0.556
7 - val_accuracy: 0.6782
Epoch 2/100
9/9 [=====] - 141s 15s/step - loss: 0.4795 - accuracy: 0.7769 - val_loss: 0.488
9 - val_accuracy: 0.7370
Epoch 3/100
9/9 [=====] - 124s 14s/step - loss: 0.3413 - accuracy: 0.8568 - val_loss: 0.459
9 - val_accuracy: 0.7682
Epoch 4/100
9/9 [=====] - 121s 14s/step - loss: 0.1750 - accuracy: 0.9505 - val_loss: 0.500
6 - val_accuracy: 0.7751
Epoch 5/100
9/9 [=====] - 99s 11s/step - loss: 0.0595 - accuracy: 0.9870 - val_loss: 0.5992
- val_accuracy: 0.7924
Epoch 6/100
9/9 [=====] - 101s 11s/step - loss: 0.0236 - accuracy: 0.9957 - val_loss: 1.030
7 - val_accuracy: 0.7543
Epoch 7/100
9/9 [=====] - 105s 12s/step - loss: 0.0241 - accuracy: 0.9922 - val_loss: 0.653
1 - val_accuracy: 0.7820
Epoch 8/100
9/9 [=====] - 100s 11s/step - loss: 0.0239 - accuracy: 0.9965 - val_loss: 0.758
3 - val_accuracy: 0.7785
Epoch 9/100
9/9 [=====] - 97s 11s/step - loss: 0.0200 - accuracy: 0.9948 - val_loss: 0.6898
- val_accuracy: 0.7889
Epoch 10/100
9/9 [=====] - 99s 11s/step - loss: 0.0079 - accuracy: 0.9991 - val_loss: 0.8580
- val_accuracy: 0.8131
```

Ilustración 42. Tiempos durante el entrenamiento del modelo 3

Layer (type)	Output Shape	Param #
input_2 (InputLayer)	[(None, 71, 130, 1)]	0
xception (Functional)	(None, 3, 4, 2048)	20860904
global_average_pooling2d (GlobalAveragePooling2D)	(None, 2048)	0
dropout (Dropout)	(None, 2048)	0
dense_1 (Dense)	(None, 300)	614700
dropout_1 (Dropout)	(None, 300)	0
dense (Dense)	(None, 2)	602
Total params: 21,476,206		
Trainable params: 21,421,678		
Non-trainable params: 54,528		

Ilustración 43. Arquitectura modelo 3

Modelo 4

```

Test loss: 0.5774462819099426
Test accuracy: 0.7231833934783936
10/10 [=====] - 8s 506ms/step

```

	precision	recall	f1-score	support
abnormal	0.72	0.72	0.72	141
normal	0.73	0.73	0.73	148
accuracy			0.72	289
macro avg	0.72	0.72	0.72	289
weighted avg	0.72	0.72	0.72	289

Ilustración 44. Resultados de las métricas del modelo 4

```

Epoch 1/100
9/9 [=====] - 98s 9s/step - loss: 0.7773 - accuracy: 0.5816 - val_loss: 0.6793
- val_accuracy: 0.6125
Epoch 2/100
9/9 [=====] - 91s 10s/step - loss: 0.6669 - accuracy: 0.6389 - val_loss: 0.6232
- val_accuracy: 0.6747
Epoch 3/100
9/9 [=====] - 88s 10s/step - loss: 0.5637 - accuracy: 0.7257 - val_loss: 0.5894
- val_accuracy: 0.7093
Epoch 4/100
9/9 [=====] - 94s 11s/step - loss: 0.4607 - accuracy: 0.7856 - val_loss: 0.5409
- val_accuracy: 0.7163
Epoch 5/100
9/9 [=====] - 77s 9s/step - loss: 0.3423 - accuracy: 0.8542 - val_loss: 0.5756
- val_accuracy: 0.7336
Epoch 6/100
9/9 [=====] - 77s 9s/step - loss: 0.2185 - accuracy: 0.9141 - val_loss: 0.6402
- val_accuracy: 0.7543
Epoch 7/100
9/9 [=====] - 84s 9s/step - loss: 0.0930 - accuracy: 0.9635 - val_loss: 0.9494
- val_accuracy: 0.7197
Epoch 8/100
9/9 [=====] - 83s 9s/step - loss: 0.1544 - accuracy: 0.9427 - val_loss: 1.0825
- val_accuracy: 0.7336
Epoch 9/100
9/9 [=====] - 87s 10s/step - loss: 0.3057 - accuracy: 0.8906 - val_loss: 1.1144
- val_accuracy: 0.6713
Epoch 10/100
9/9 [=====] - 87s 10s/step - loss: 0.3231 - accuracy: 0.8715 - val_loss: 0.7069
- val_accuracy: 0.6713

```

Ilustración 45. Tiempos durante el entrenamiento del modelo 4

Layer (type)	Output Shape	Param #
input_2 (InputLayer)	[(None, 40, 130, 1)]	0
resnet101v2 (Functional)	(None, 2, 5, 2048)	42620288
global_average_pooling2d (GlobalAveragePooling2D)	(None, 2048)	0
dropout (Dropout)	(None, 2048)	0
dense_1 (Dense)	(None, 300)	614700
dropout_1 (Dropout)	(None, 300)	0
dense (Dense)	(None, 2)	602
Total params: 43,235,590		
Trainable params: 43,137,926		
Non-trainable params: 97,664		

Ilustración 46. Arquitectura del modelo 4