



COMILLAS

UNIVERSIDAD PONTIFICIA

ICAI

GRADO EN INGENIERÍA EN TECNOLOGÍAS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO
Music Detecting Light System

Autor: Alfredo Sánchez Sánchez

Director: Javier Matanza Domingo

Madrid

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título

Music Detecting Light System

en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el

curso académico 2019/20 es de mi autoría, original e inédito y

no ha sido presentado con anterioridad a otros efectos.

El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido

tomada de otros documentos está debidamente referenciada.

Fdo.: Alfredo Sánchez Sánchez

Fecha: ..11../ ..07../ ..20.

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO

Firmado por
MATANZA DOMINGO
JAVIER -
48543978V el día

Fdo.: Javier Matanza Domingo

Fecha: ..11../ ..07../ ..2020



COMILLAS

UNIVERSIDAD PONTIFICIA

ICAI

GRADO EN INGENIERÍA EN TECNOLOGÍAS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO
Music Detecting Light System

Autor: Alfredo Sánchez Sánchez

Director: Javier Matanza Domingo

Madrid

Agradecimientos

Para empezar, me gustaría agradecer a mi tutor del proyecto Javier Matanza Domingo y a la Escuela de ICAI por su trabajo para asistirme y por su comprensión en estos tiempos de pandemia.

También he querido destacar y agradecer la labor de mis compañeros de ECE402-Senior Design Project de la Universidad en la que he estado de intercambio, University of Illinois at Urbana-Champaign, Francis Mui y Rang Wang porque junto a ellos empecé a gestar la idea del proyecto. Además, para mí fue también decisivo, la ayuda, en los comienzos del proyecto, el TA (Teacher Assistant) del curso de Urbana, Ruhao Xia.

Por último, me gustaría agradecer su ayuda a mi primo Ricardo Bravo, por su colaboración y ayuda tanto a la hora de crear el modelo del clasificador como en la redacción de la parte del clasificador de esta memoria. También agradecer su ayuda a la hora de ofrecer recursos como el curso Machine Learning de Andrew Ng ofrecido por la Universidad de Stanford a través de Coursera que me ha ayudado mucho a la hora de entender muchos de los conceptos necesarios para el proyecto.

MUSIC DETECTING LIGHT SYSTEM

Autor: Sánchez Sánchez, Alfredo.

Supervisor: Matanza Domingo, Javier.

Entidad Colaboradora: Universidad Pontificia Comillas

RESUMEN DEL PROYECTO

El objetivo del sistema es el de crear un sistema que mezcle luces y música. Para ello se capturará el audio con un micrófono, el audio capturado se analizará y clasificará según su género, para así hacer que unas luces leds brillen con distintos colores y figuras según el género que se haya clasificado al ritmo de la música.

Palabras clave: Machine Learning, Raspberri Pi, Leds.

1. Introducción

Sistema constará de varias partes en forma de “Ceiling Analysis” cuyo esquema a modo resumen, se muestra en la Ilustración 1.

La primera parte del proyecto consistirá en obtener el audio capturado con el micrófono, el cual será analizado de dos formas, una que detectarán las pulsaciones por minuto (beats per minute, o BPM) a tiempo real y otra que analice el audio y clasifique la canción que este sonando según el género.

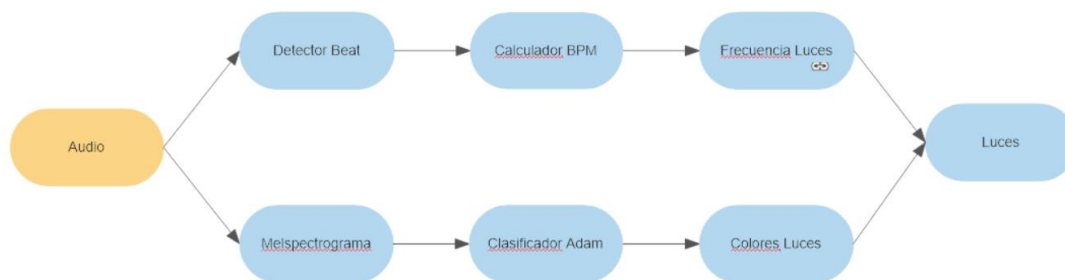


Ilustración 1- Esquema de Ceiling Analysis

2. Definición del Proyecto

Para realizar el proyecto el uso de una Raspberry Pi será fundamental, en la Raspberry Pi, se instalará el sistema operativo basado en la distribución de GNU/Linux llamado Raspbian. El código será implementado con el lenguaje de programación Python versión 3.7. El proyecto constará de dos partes principales, un clasificador de música por géneros y un detector de beats.

3. Descripción del Sistema

El Sistema, para su mejor explicación y análisis, se dividirá en cuatro partes destacadas, entre las que se encontrarán la captura de audio, el clasificador, el detector de beats y la salida de luces.

El programa de captura de audio grabará el audio el tiempo que se indique creando un vector con los valores captados y guardándolo posteriormente en un archivo de audio .WAV. Para el proyecto se precisará que el micrófono tenga incorporado un conversor analógico/digital integrado. En la parte de control de la Raspberry Pi, se implementarán los programas que llamen al modelo del clasificador y analicen el audio anteriormente capturado y guardado en el archivo en formato .WAV, una vez se haya clasificado el audio grabado, se seleccionará el color y figura que se desplegará en las luces, cada género tendrá un color y figuras únicos y distintos.

Una vez se tengan todos los parámetros necesarios para el inicio del encendido de luces, el detector de beats se correrá indefinidamente dentro de un bucle infinito, haciendo que se enciendan y apaguen las luces de la placa según se vayan detectando los beats.

El programa de detector de beats, también tendrá la capacidad de detectar, cuando no se detecte audio si se ha parado la canción y ha empezado una nueva, cuando esto ocurra, el proceso se reiniciará desde el principio.

Para el manejo de la matriz de leds, la placa se puede manejar en cuatro modos, pero en este caso el único modo que va a interesar va a ser el que controle cada led por separada para que se puedan dibujar las figuras en la matriz de mejor manera. En la Ilustración 2, se muestra un esquema del sistema con un resumen de lo que harán y las características de los componentes hardware que usará el sistema.

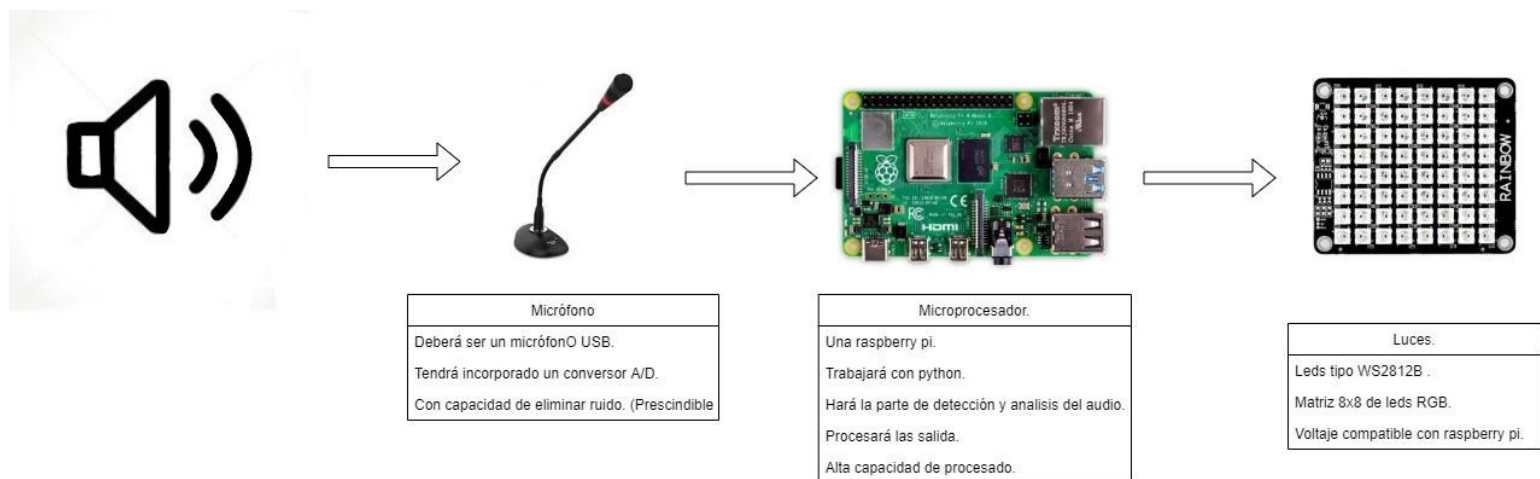


Ilustración 2– Esquema del Sistema.

4. Resultados

A la hora de analizar los resultados, se analizarán con respecto a los objetivos que se desean obtener y que son la latencia, la precisión y capacidad de adaptación, se analizarán los objetivos de las dos partes fundamentales del sistema por separado. En el caso del

detector de beats el análisis se quedará en el análisis de un solo objetivo, puesto que los objetivos de latencia y precisión están muy relacionados.

Para el objetivo de capacidad de adaptación, no se precisa de un amplio análisis, gracias al uso de la Raspberry Pi y como está diseñado el proyecto, se considerará que cumple el objetivo si el programa en general funciona correctamente.

En el caso del objetivo de latencia del clasificador, se centrará en el análisis de la optimización del tiempo de grabación, haciendo que este tiempo se reduzca y que la precisión sea la máxima posible. Para analizar dicho objetivo se grabarán varias canciones a distintos tiempos y conforme a un sistema de puntuaciones se analizará la precisión en la clasificación de las grabaciones. En la Ilustración 3, se muestra una gráfica con las puntuaciones otorgadas según el tiempo, será la gráfica que se use en el análisis de la latencia del clasificador.

A la hora de evaluar la precisión del clasificador, se crearán dos modelos uno con el dataset Gtzan Genre Collection integro y otro con el dataset con un número de canciones añadidas. Se compararán ambos modelos usando los valores de exactitud, pérdida y las matrices de confusión junto con los parámetros que se pueden obtener de ella.

Para realizar el análisis de la latencia y precisión del detector de beats, se recurrirá a la plataforma Tunebat, una página web con un número elevado de canciones que entre otras características incluye su BPM, dicho valor que se obtenga de la página web se comparará con el resultado del programa. (Tunebat)

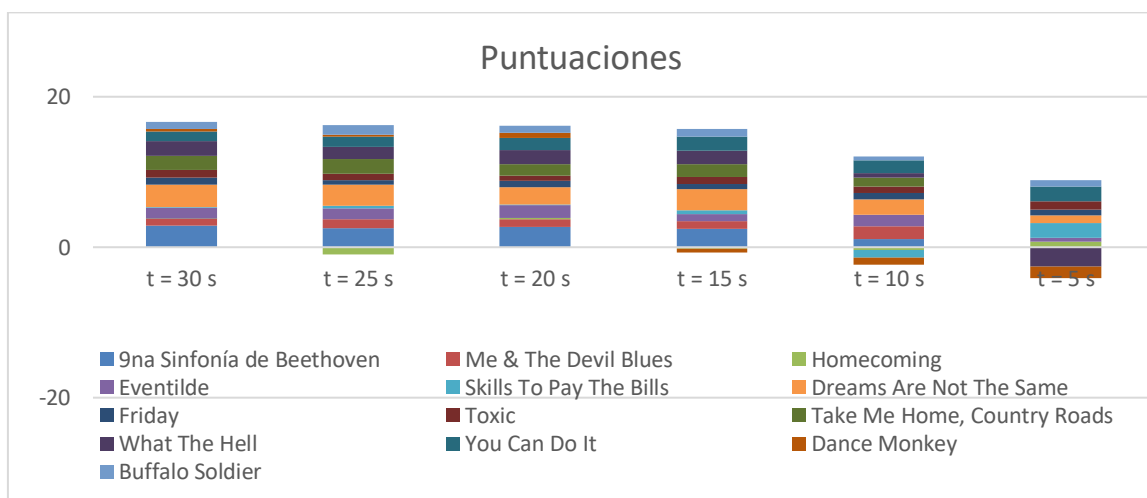


Ilustración 3- Gráfica de puntuaciones según el tiempo

5. Conclusiones

Tras el análisis de los resultados, se concluye finalmente que, con respecto a la latencia del clasificador, el mejor tiempo de grabación, es decir, el tiempo que mejor optimiza el tiempo de grabación y la precisión es de 15 segundos, dejando una latencia de unos 22 segundos con respecto a la clasificación. Se usará el modelo creado con el dataset truncado, al obtener resultados más precisos y acordes a la música con la que el sistema más se va a utilizar. A la vista de los resultados de los análisis se puede extraer que, para el rendimiento del clasificador, se podría hacer mediante un aumento considerable del dataset a la vez que se aumenten los hiperparámetros del modelo.

Con respecto al detector de beats, se puede concluir que el funcionamiento, en líneas generales es el idóneo y correcto, siempre y cuando el micrófono tenga una señal clara y limpia del audio.

MUSIC DETECTING LIGHT SYSTEM

Author: Sánchez Sánchez, Alfredo.

Supervisor: Matanza Domingo, Javier.

Collaborating Entity: Universidad Pontificia Comillas

ABSTRACT

The system objective is for the light patterns to change based on the music detected. In order to accomplish it, the audio sound will be captured using a microphone, once the sound is recorded, it will be analyzed and classified by its genre, so as to make the light pattern change depending on the genre the music has been classified.

Keywords: Machine Learning, Raspberry Pi, Leds.

1. Introduction

The system will be divided into different parts using the “Ceiling Analysis” method, the scheme is represented in Figure 1. To begin with, the first thing that must be done will be taking the audio by capturing using a microphone, the audio must be analyzed in two different ways. The first way will be a real time beat per minute or BPM detector and the second way will be a Machine Learning classifier that will classify the sound by its genre.

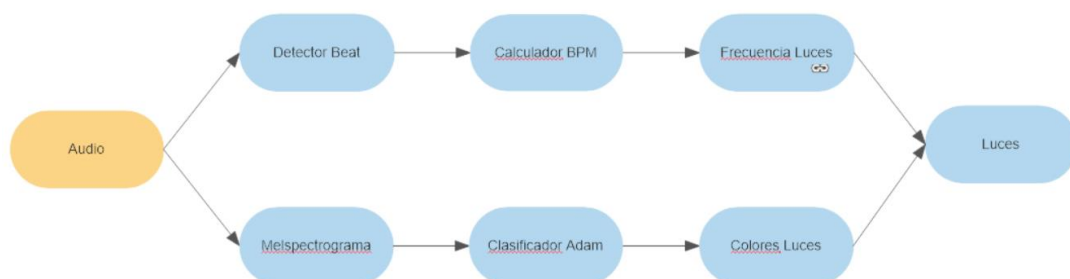


Figure 1- Ceiling Analysis scheme

2. Project Definition

A Raspberry Pi will be the main component for the control part of the project, in the Raspberry Pi a GNU/Linux Raspbian operating system will be installed. Python 3.7 will be the main programming language for the entire project. The project will have two main parts, that were introduced in the Introduction part and that are the musical genre classifier and the beat detector.

3. Description of the system

For a better synthesis and analysis of the system, it will be divided into four different parts. The four parts will be the audio capture, the classifier, the beat detector and the lights output.

The capturing audio program will record the time that the user wants to, using an array with the values captured by the microphone and those values will be saved into a .WAV file. A microphone with an analog-to-digital converter will be needed. In the control part that will be using the Pi include the program that will load the model of the classifier and that will analyze the audio saved in the .WAV file. Once the file is classified a color and figure will be selected for the lights pattern to show, each genre will have a unique representative figure and color.

Once all the parameters are ready for the lights to start shining, the beat detector will be recurrently running in an infinitive while loop, so as that the lights will be clicking at the same rhythm as the program detects beats.

The beat detector program will also have the ability to detect if a song has stopped and if that is the case, the whole program will restart himself again.

For the UnicornHat matrix it has 4 modes to control the leds, but we will only be interested in the mode in which we control each led of the matrix separately, that will give the ability to use figures along the matrix.

In Figure 2, a system scheme of the system is shown with a summary of the hardware components characteristics and their value in the project.

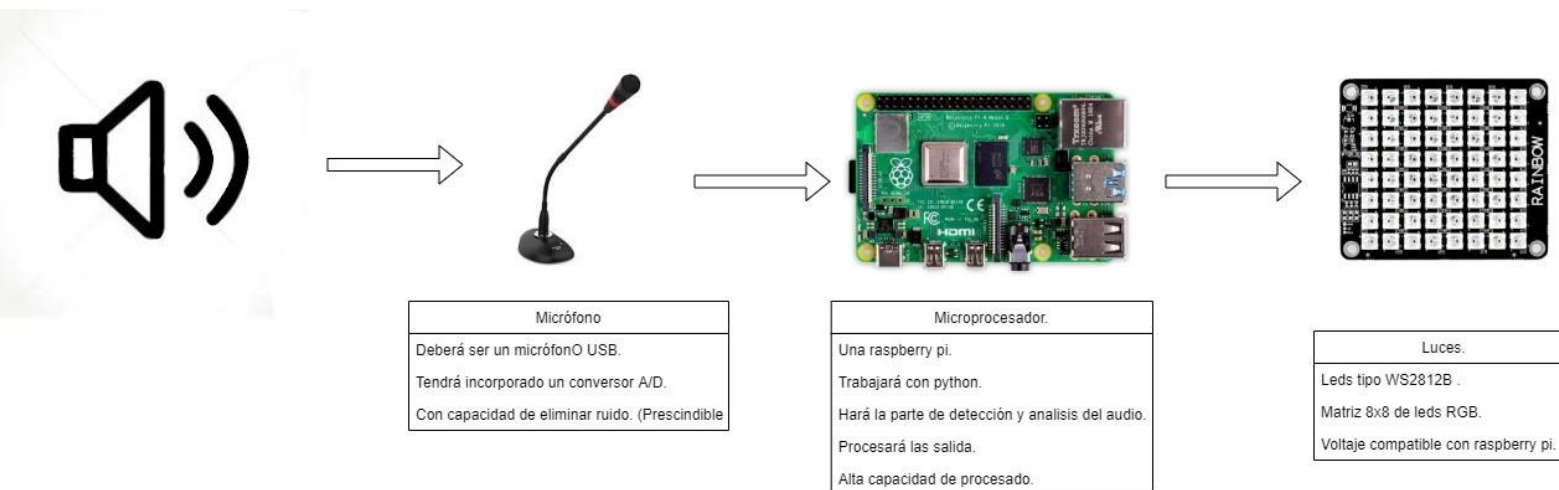


Figure 2- System scheme

4. Results

At the time of analyzing the results, it will be analyzed with respect to the objectives that are willing to be obtained and that are latency, precision and the adaptative capacity, those objectives will be analyzed of the two important parts of the system separately. In the beat detector analysis will only be needed one objective analysis, because the latency and precision objectives are totally related.

For the adaptative capacity objective, an analysis will not be needed, thanks to the Pi use and how the project is designed, the objective will be accomplished if the program works as it should be in a general way.

For classifier latency, the analysis will be focus on optimizing the recording time, reducing the time and maximizing the precision. In order to analysis the objective some songs will be recorded with different recording times and by using a ranking system the

precision will be analyzed. In the fafsd, a graphic of the rankings of the songs recorded with the time, it will be graph used in the latency analysis of the classifier.

In the precision analysis of the classifier, two different classification models will be created one with the full sdadf dataset and another with the dataset with some songs added to the dataset. Both models will be compared using the accuracy, the loss and the confusion matrix with the parameters that are easily obtainable from the confusion matrix.

For the latency and precision analysis of the beat detector, the platform Tunebat will be used, a web page with a huge number of song characteristic such as the BPM, the web page value will be compared to the result of the program.

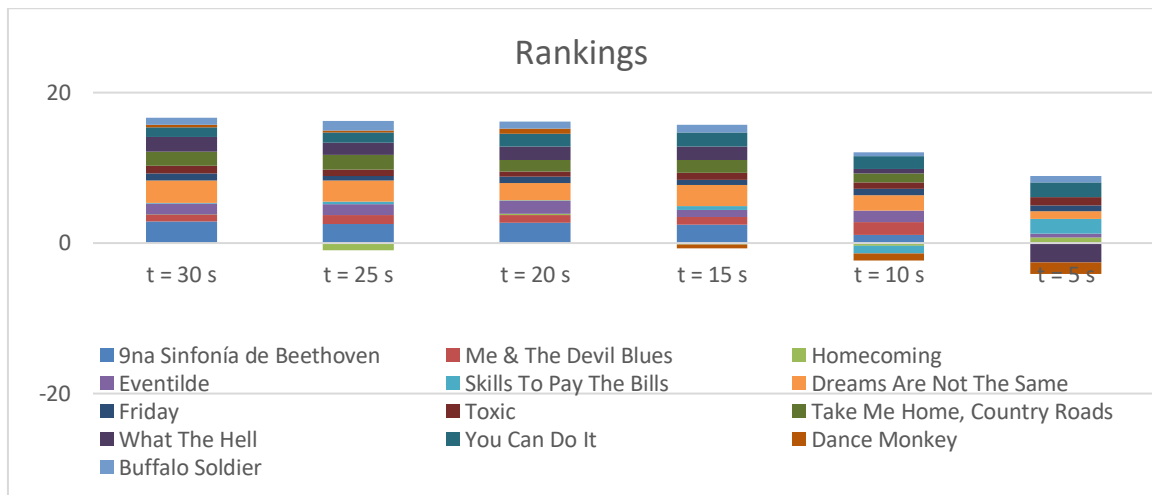


Figure 3- Graphic of rankings among time

6. Conclusions

After the analysis, a conclusion can be made, saying that the best time for maximizing the precision and minimizing the time that the classifier takes will be 15 seconds, making the whole classifier latency to be 22 seconds. Another conclusion obtained can be that the model that best fits the results would be the model created with the altered classifier.

We can also say that the best way to improve the classifier performance will be to add more songs to the dataset and augmenting the hyper parameters of the model as well. The beat detector program works as expected, but the microphone must obtain a clean audio signal, without noise.

Índice de la memoria

Capítulo 1. Introducción	I
1.1 Motivación del proyecto	I
Capítulo 2. Descripción de las Tecnologías	II
2.1 Raspberry Pi.....	II
2.2 TensorFlow	II
2.3 Unicorn Hat.....	II
2.4 Espectrograma.....	III
2.5 Escala Mel.....	IV
2.6 Melspectrograma	IV
2.7 Gtzan Genre Collection	V
2.8 Redes neuronales	VI
Capítulo 3. Estado de la Cuestión	VII
Capítulo 4. Definición del Trabajo	VIII
4.1 Justificación	VIII
4.1.1 Capacidad de Análisis.....	VIII
4.1.2 Automatización	VIII
4.1.3 Transportabilidad	VIII
4.2 Objetivos.....	IX
4.2.1 Latencia.....	IX
4.2.2 Precisión	IX
4.2.3 Capacidad de adaptación.....	X
4.3 Metodología	X
4.3.1 Clasificador.....	X
4.3.2 Detector de Beats.....	XI
4.4 Planificación y Estimación Económica	XII
4.4.1 Planificación.....	XII
4.4.2 Estimación económica	XV
Capítulo 5. Diccionario de términos	XVIII
Capítulo 6. Sistema/Modelo Desarrollado	XXIII

6.1	Captura de datos	XXIII
6.1.1	Fundamentos del sistema	XXIII
6.1.2	Diseño	XXIV
6.1.3	Implementación.....	XXV
6.2	Clasificador	XXVI
6.2.1	Fundamentos del sistema	XXVI
6.2.2	Diseño	XXX
6.2.3	Implementación.....	XXXIII
6.3	Detector de beats	XXXV
6.3.1	Fundamentos del sistema	XXXV
6.3.2	Diseño	XXXVI
6.3.3	Implementación.....	XXXIX
6.4	Salida de las luces.....	XLI
6.4.1	Fundamentos del sistema	XLI
6.4.2	Diseño	XLI
6.4.3	Implementación.....	XLIV
Capítulo 7. Análisis de resultados		XLV
7.1	Clasificador	XLV
7.1.1	Latencia.....	XLV
7.1.2	Precisión	LII
7.1	Detector de beats	LXII
Capítulo 8. Conclusiones y Trabajos Futuros		LXVIII
Capítulo 9. Bibliografía.....		LXXI
ANEXO I: ALINEACIÓN DEL PROYECTO CON LOS ODS.....		LXXVI
ANEXO II		LXXVIII

Índice de figuras

Ilustración 1- Esquema de Ceiling Analysis	8
Ilustración 2- Esquema del Sistema.	9
Ilustración 3- Gráfica de puntuaciones según el tiempo	10
Ilustración 4- Ejemplo de Espectrograma.	III
Ilustración 5- Ejemplo de Melspectrograma	V
Ilustración 6- Esquema captura de audio	XXIV
Ilustración 7- Melspectrogramas por géneros	XXVIII
Ilustración 8- Diagrama de los conjuntos de datos	XXXI
Ilustración 9- Resumen del modelo	XXXII
Ilustración 10- Logos de las principales librerías usadas	XXXIV
Ilustración 11- Posiciones de probabilidades con géneros.....	XXXV
Ilustración 12- Calculo variables detector de beats	XXXVII
Ilustración 13- Árbol de decisión de detector de beats	XXXVIII
Ilustración 14- Resumen vector bpm	XXXIX
Ilustración 15- Esquema de salida de luces.....	XLI
Ilustración 16- Gráficas de puntuaciones	LI
Ilustración 17- Tasa aprendizaje de exactitud creado con el modelo íntegro	LIII
Ilustración 18- Tasa aprendizaje de exactitud creado con el modelo truncado.....	LIII
Ilustración 19- Tasa aprendizaje de pérdida creado con el modelo íntegro	LV
Ilustración 20- Tasa aprendizaje de pérdida creado con el modelo truncado	LV
Ilustración 21- Diagrama de comparación de los modelos	LXI

Índice de tablas

Tabla 1- Planificación mes 1	XII
Tabla 2- Planificación mes 2	XIII
Tabla 3- Planificación mes 3	XIV
Tabla 4- Planificación mes 4	XIV
Tabla 5- Coste de componentes Hardware.....	XVI
Tabla 6- Descripción de componentes Hardware.....	XVII
Tabla 7- Pistas añadidas a cada género	XXVII
Tabla 8- Relación Melspectrogramas con géneros.....	XXIX
Tabla 9- Relación de géneros y figuras en la salida	XLII
Tabla 10- Relación de géneros y colores en la salida	XLIII
Tabla 11- Tiempos totales.....	XLVIII
Tabla 12- Puntuaciones de géneros	XLIX
Tabla 13- Puntuaciones obtenidas	LI
Tabla 14- Matriz de confusión del modelo creado con el dataset íntegro	LVI
Tabla 15- Matriz de confusión del modelo creado con el dataset truncado.....	LVII
Tabla 16- Parámetros del modelo creado con el dataset íntegro	LX
Tabla 17- Parámetros del modelo creado con el dataset truncado.....	LX
Tabla 18- Comparación de modelos	LXII
Tabla 19- Tabla de canciones para test de detector de beats.....	LXIV
Tabla 20- Cálculos de medias BPM.....	LXV
Tabla 21- Comparación de BPMs.....	LXVI

Capítulo 1. INTRODUCCIÓN

1.1 MOTIVACIÓN DEL PROYECTO

La música es un lenguaje que se compone de algo más que únicamente sonido. La vibración de una tecla de piano al ser tocada, el rasgar de las cuerdas de una guitarra o el simple temblor del repiqueteo de una batería forma parte también de la música. La luz, por otro lado, puede servir para interpretar otras realidades, realidades que no siempre es posible captar con nuestros sentidos y que permite crear lazos entre personas.

Luces y música son dos instrumentos que separados son básicos a la hora de proporcionar entretenimiento. Se puede utilizar tanto luces como música para crear un entorno relajado y evocador para descansar, también se puede crear el efecto contrario de vitalidad y energía utilizando cualquiera de las dos herramientas, pues se adaptan perfectamente a cualquier estado de ánimo o situación. Son dos instrumentos útiles a la hora de crear y cambiar estados de ánimo, pero no por ello incompatibles, el objetivo de este proyecto es el de combinar ambas características para así capturar lo que se esté escuchando y traducirlo en una receta de luz “inmediata”, el objetivo es crear una conexión profunda entre la música, las luces y el usuario.

Capítulo 2. DESCRIPCIÓN DE LAS TECNOLOGÍAS

Para el proyecto se utilizarán las siguientes tecnologías:

2.1 RASPBERRY PI

Se utilizará una Raspberry Pi 4 Computer Model B 4GB RAM. Se trata de una placa que soporta varios componentes necesarios en un ordenador común, en ella se instanciará un programa clasificador, que se necesitará que corra lo más rápido posible, además de la necesidad de realizar muchas operaciones costosas para una simple placa. Los 4GB de RAM son necesarios para abaratar costes computacionales. (ABC Tecnología, 2013)

2.2 TENSORFLOW

Es una biblioteca de código abierto desarrollada por Google para llevar a cabo proyectos de Machine Learning. TensorFlow fue creada por el equipo de Google Brain y sacada en 2015 bajo la licencia Apache 2.0. Es una herramienta muy usada y empleada principalmente para la construcción de redes neuronales. La librería se usará en el lenguaje de programación Python 3.7. en la Raspberry Pi. (Equipo Paradigma Digital, 2020)

2.3 UNICORN HAT

Se trata de una matriz 8x8 de leds RGB inteligentes conocidos como ws2812s. Cada led ws2812 es capaz de entender señales como parecidas al código morse. La señal indica a un led que valores de Rojo, Verde y Azul lucirá y el led, una vez que conoce el color, se lo pasará a los vecinos cercanos. Esta transmisión ocurre tan rápido que no puede ser captado por un ser humano. (Howard, s.f.)

2.4 ESPECTROGRAMA

Se trata de una representación gráfica del espectro de frecuencias de una emisión sonora. Para obtenerlo se divide la señal en ventanas superpuestas y se calculará la FFT dichas ventanas en las que se ha dividido a la señal. El espectrograma tendrá la frecuencia en el eje y, el tiempo en el eje X y el color indicará la amplitud en decibelios. En la Ilustración 4 se muestra un ejemplo de espectrograma con la canción “*American Idiot*” de “*Green Day*”. (iZotope Education Team, 2020)

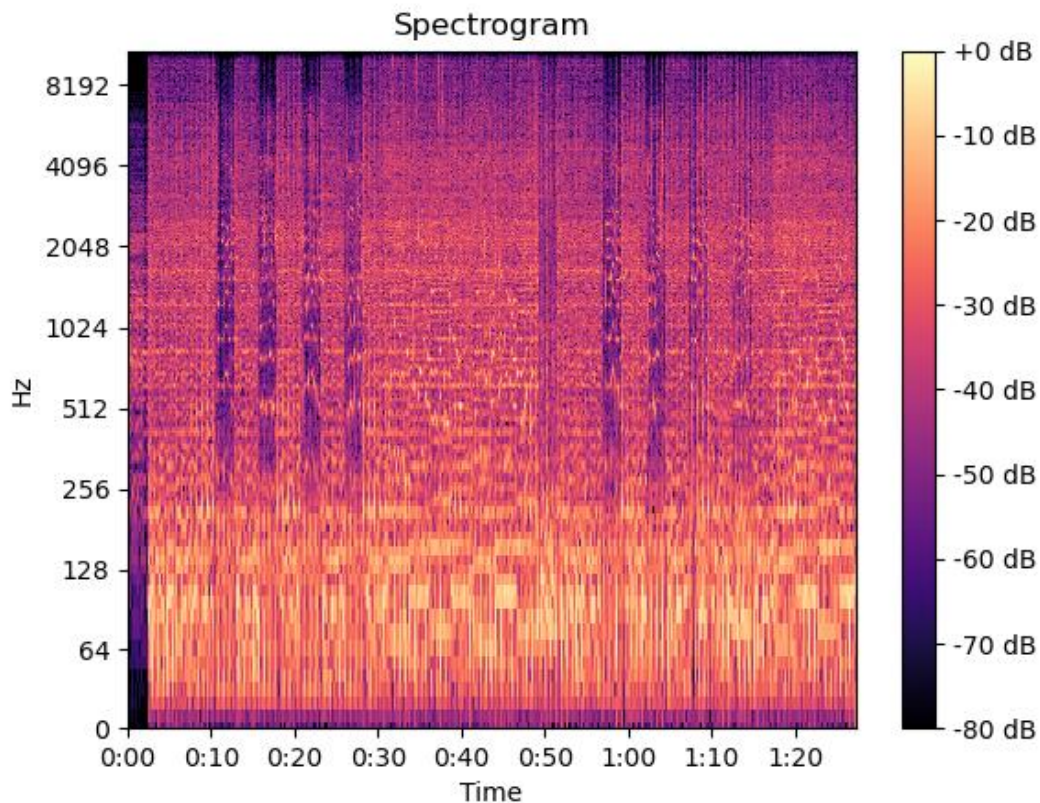


Ilustración 4- Ejemplo de Espectrograma.

2.5 ESCALA MEL

Se trata de una escala psico acústica que se construyó a partir de tests psico acústicos con dos tonos, donde las personas percibían la distancia entre esos tonos y la evaluaban como un salto de distancia equivalente de otros. Dicha escala parte de mil hercios a los que se asignan un valor de mil mels, es decir, se ancla el mapeo justo en un kilohercio y a partir de ahí se va aproximando con la Ecuación 1. Básicamente consiste en escalar las frecuencias a las que los sonidos son mejor reconocidos por los humanos. (Monfort, 2015)

$$m = 2595 \log_{10} \left(1 + \frac{f}{700} \right)$$

Ecuación 1- Fórmula de Escala mel

2.6 MELSPECTROGRAMA

Se trata de un espectrograma donde las frecuencias se convierten a la escala mel. Los melspectrogramas sirven para analizar la sonoridad, la duración, la estructura de los formantes, la intensidad, las pausas e incluso el ritmo en una misma gráfica y conforma la entrada perfecta para el clasificador de géneros. En la Ilustración 5, se puede observar el melspectrograma de la misma canción usada en la Ilustración 4. (Gartzman, 2019)

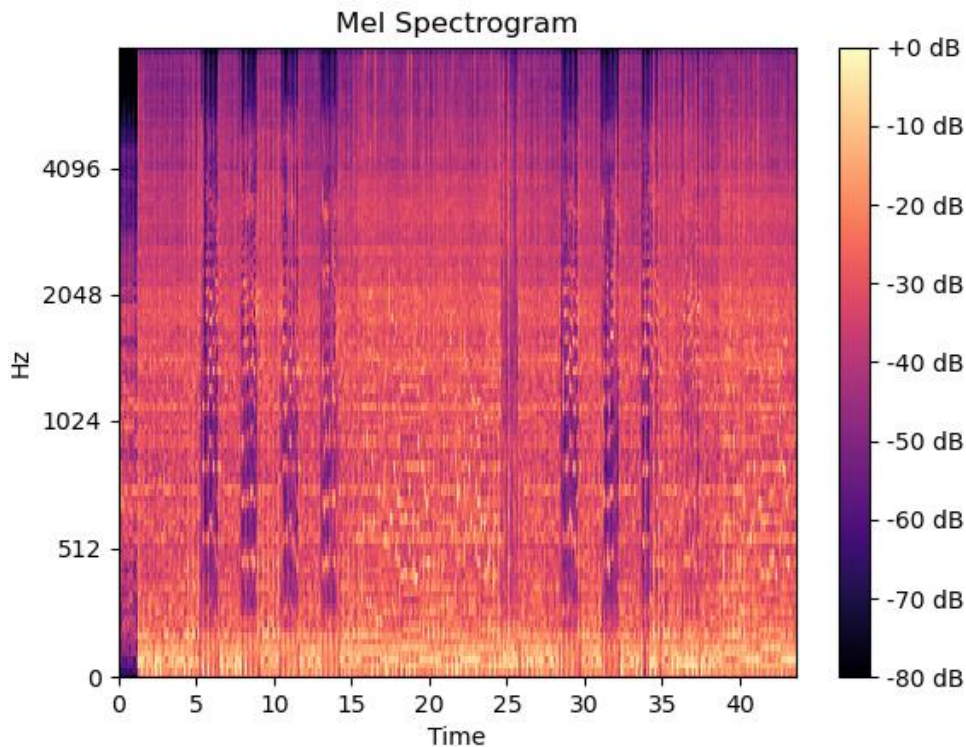


Ilustración 5- Ejemplo de Melspectrograma

2.7 GTZAN GENRE COLLECTION

Se trata de un dataset creado por George Tzantakis y Perry Cook en su trabajo “Music genre classification of audio signals”, uno de los primeros trabajos en clasificación de música en géneros. El dataset consiste en 1000 fragmentos de canciones de 30 segundos cada uno divididos en 10 géneros: Blues, Clásica, Country, Disco, Hip-hop, Jazz, Metal, Pop, Reggae y Rock. Los fragmentos fueron recogidos de distintas fuentes, como la radio o CDs, haciendo que exista una variación en las condiciones de grabación. En el proyecto se añadirán más canciones al dataset, de forma que se corrijan algunos defectos para el clasificador. (González, 2019)

2.8 REDES NEURONALES

Se trata de una red neuronal artificial con aprendizaje supervisado, que contiene varias capas ocultas especializadas y con jerarquía, de forma que las primeras capas pueden detectar elementos simples y se van especializando para detectar finalmente elementos más complejos. Finalmente, la última capa, conocida como la capa de salida, dará las predicciones finales que en este caso serán los géneros. (Equipo Paradigma Digital, 2020)

Dentro de las redes neuronales se pueden encontrar algunas variantes entre las que destacan las redes CNN (redes neuronales convolucionales) y las RNN (redes neuronales recurrentes).

Las redes neuronales recurrentes son una variante de redes neuronales muy utilizadas para el procesamiento del lenguaje. En este tipo de redes, una entrada se procesa a través de varias capas y se produce una salida, suponiendo que dos entradas sucesivas son independientes entre sí. Se le llama recurrentes porque realizan la misma tarea para cada elemento de una secuencia, y la salida depende de los cálculos anteriores, Otra forma de verse este tipo de redes, es que poseen “memoria” que guarda información sobre lo calculado anteriormente.

Las redes neuronales convolucionales es una variante de redes neuronales que se utilizan en el campo de la visión artificial, con el manejo e identificación de imágenes. Su nombre viene de que el tipo de capas en que se componen consiste en capas convolucionales, capas agrupadas, capas totalmente conectadas y capas de normalización. Aquí significa solo que en lugar de usar funciones de activación normales, las funciones de convolución y agrupación se utilizan como funciones de activación. El clasificador creado en el proyecto será de este tipo, pues al final lo que se pretende es comparar los audios en forma de Melspectrograma. (Cornieles, 2019)

Capítulo 3. ESTADO DE LA CUESTIÓN

Actualmente, lo que se puede encontrar son tiras de leds que se encuentran conectadas en paralelo a la bobina de unos altavoces, de forma que se iluminan los leds al igual que el sonido sale del altavoz. No obstante, esta solución es una solución vaga que no lleva a cabo un análisis de la música o la canción que se está escuchando. Y lo único que tiene en cuenta es el volumen de la canción y los cambios de este. (MCMCHRIS, 2018)

También se pueden encontrar tiras de leds que tienen preinstaladas juegos de luces de manera automática dependiendo del tempo y el color que el propio usuario elija. Así, el usuario selecciona que juego le viene mejor a que canción, pero no haciéndolo de forma automática, sin la intervención de ningún usuario que de manera subjetiva elija el set de luces. (Amazon, s.f.)

Como reconocimientos de canciones, se pueden encontrar aplicaciones como Shazam, cuyo algoritmo consiste en identificar unos puntos comunes en las ondas del espectrograma y cuyos resultados son comparados con los existentes y ya guardados en la base de datos de los servidores. Algo parecido se implementará en el sistema, lo único que, en este caso las canciones se clasificarán según el género y no se pretenderá comparar cada audio o pista de audio grabada con las guardadas en una base de datos. (Jovanovic, 2015)

Capítulo 4. DEFINICIÓN DEL TRABAJO

4.1 JUSTIFICACIÓN

4.1.1 CAPACIDAD DE ANÁLISIS

El sistema tendrá la capacidad de analizar la música que este sonando de manera que el usuario, además de hacer que el usuario disfrute de las luces brillando al ritmo de la música, podrá conocer, se le interesa y lo desea, el género aproximado de ésta y el ritmo al que está sonando, pues los leds brillaran de acorde al ritmo. De esta forma, no se utiliza la señal de la música directamente como salida, como hacen las tiras de leds conectadas en paralelo a la bobina del altavoz, sino que se obtendrá una señal propiamente analizada y no obteniendo directamente luces aleatorias que brillan según las vibraciones de los altavoces.

4.1.2 AUTOMATIZACIÓN

El sistema, tampoco será un juego preinstalado, pues el objetivo es que el sistema sea cuanto más dinámico posible y que pueda adaptarse a la mayoría de las situaciones en que se use, incluso con canciones poco conocidas o que sean difícilmente analizables por los usuarios. El sistema interpretará el sonido (su tono, el ritmo, el volumen) en tiempo real y los convierte en luz, a través de tecnología LED, con variantes como el color, luminosidad y movimiento. De esta forma, al usuario se le quita la molestia y la responsabilidad de seleccionar y elegir como lucirán las luces y se realiza de forma automática y autónoma.

4.1.3 TRANSPORTABILIDAD.

Se pretende crear un sistema fácilmente transportable y que tenga una gran capacidad de adaptación, de forma que pueda servir tanto para música ya grabada anteriormente y reproducida desde cualquier dispositivo de reproducción musical, como también para música improvisada o que se esté tocando en directo. Así se crea una experiencia multisensorial que puede ser aplicada a festivales, museos de arte, eventos culturales,

espectáculos; en fin, en cualquier lugar en que se desee que luz y sonido interactúen. Esta especial característica hace que el sistema que se implementara en este proyecto sea distinto y más completo que los descritos anteriormente.

4.2 OBJETIVOS

El objetivo, como se ha explicado en la Introducción, es el de crear un sistema que mezcle luces y música. Para ello, se analizará la música que este sonando y se harán encender leds al ritmo de esta de forma automática.

A la hora de clasificar y detectar las pulsaciones, se deberán atender tres objetivos principales, estos objetivos que se pretenden alcanzar son también los que ayuden a comprobar que el sistema funciona de forma esperada y ayudará a la hora de comprobarlos mediante test. (Mccrea, 2014)

4.2.1 LATENCIA

Se deberá de controlar el tiempo que tarda el sistema en analizar la canción y hacer que las luces brillen con todo el material analizado. Para comprobar este requisito a la hora de testear el programa, se creará una variable que calcule el tiempo que transcurre entre que entra el input y se obtiene el output, esto se hará en todos los apartados del sistema. El mayor problema con este requisito se encuentra con el clasificador, pues es la parte en la que más tiempo se puede perder. El detector de beats, no supondrá tanto problema en este aspecto, pero también se pretenderá que no tenga un retraso de mucho más de 0.5 segundos.

4.2.2 PRECISIÓN

Requerimos que el sistema sea preciso, pero ya que se trata de una precisión subjetiva, esta consistirá en dos cosas, dependiendo de lo que queramos medir:

Para medir la precisión del clasificador por géneros, cuyo resultado se verá en el color en que luzcan los leds, se considerará una salida correcta, cualquier color que a una persona

normal y corriente, sin necesidad de ser experto en música, no le extrañe, conociendo a que color corresponde cada género.

Cuando hablamos del detector de beats, cuyo resultado se verá en la frecuencia con la que brillen los leds, se considerará algo preciso si es capaz, de dar un resultado en canciones donde el ritmo este terriblemente marcado, que es lo que se usará a la hora de contrastar los resultados.

4.2.3 CAPACIDAD DE ADAPTACIÓN

Se pretende que el sistema pueda y tenga la capacidad de adaptarse a varias situaciones o entornos. Es decir, que no solo funcione en casa, sino que sea fácilmente transportable y pueda funcionar en la mayor parte de ambientes donde pueda haber música. Necesitamos un sistema que no sea muy grande, como los típicos equipos de música, para que así sea más manejable y fácil de transportar. Esto explica el uso de la Raspberry Pi, puesto que es un equipo pequeño y a la vez con alta capacidad de computación que será fundamental.

4.3 METODOLOGÍA

Para realizar el proyecto, como ya se ha mencionado antes, será necesaria una Raspberry pi, donde se instalará Raspbian y Python. Gracias al uso de la Raspberry Pi, el objetivo de Capacidad de adaptación será más fácil de cumplir. El proyecto como se ha introducido en los apartados anteriores constará de dos partes principales, un clasificador de música por géneros y un detector de beats para tener el ritmo de la canción para así hacer que los leds luzcan.

4.3.1 CLASIFICADOR

El clasificador actuará sobre un audio en formato “.wav” (*Waveform Audio Format*), obtenido gracias a un programa de Captura de datos cuyo archivo será usado por la clasificación del audio, para ello, se deberá construir un modelo clasificador. Se usará un clasificador que utilizará redes neurales convolucionales (*Convolutional Neural Networks*, o CNN), que es un tipo de red neuronal artificial con aprendizaje supervisado, que contiene

varias capas ocultas especializadas y con jerarquía, de forma que las primeras capas pueden detectar elementos simples y se van especializando para detectar finalmente elementos más complejos (Equipo Paradigma Digital, 2020). Se trata de la forma de aprendizaje supervisado que más se asemeja a la forma en que piensan los humanos, esta es una forma de asegurar que el objetivo de Precisión se pueda cumplir. Se debe exigir que el clasificador tenga una precisión entre el 70- 80 % tanto cuando se evalué el train set como cuando se haga lo propio con el test set.

Se creará el modelo y se entrenará con el dataset Gtzan Genre Collection. Una vez tengamos el modelo con la precisión especificada entonces, se instanciará un programa, que será el que finalmente use la Raspberry pi que leerá el modelo generado y mediante la comparación del Melspectrograma, el modelo dará el género en que lo clasifique. Se obtendrá el género de la canción o los géneros que sean más probables, junto con su porcentaje de fiabilidad. Con el género que obtengamos, se elegirá tanto el color como las figuras con en el que las luces lucirán.

A la hora de controlar la latencia del clasificador y para testearla, se creará una variable que cuente el tiempo desde que se inicia el programa hasta que se clasifica la canción. Se debe prestar la máxima atención de forma que se maximice la precisión de los resultados minimizando el tiempo, puesto que la relación de ambos es directamente proporcional y cuanto más tiempo de grabación se use el clasificador, mayor será la precisión y viceversa.

4.3.2 DETECTOR DE BEATS

Para obtener el ritmo a tiempo real de las canciones y por lo tanto el ritmo con el lucirán los leds, será necesario un programa que con el audio a tiempo real encuentre los beats y que pueda transformar esos beats encontrados a señales de luz para los leds, el programa además tendrá la capacidad de calcular los BPM de la señal y el tiempo medio e instantáneo entre beat y beat.

Puesto que se trata de un detector a tiempo real, el tiempo de latencia debe ser mínimo y prácticamente instantáneo, la precisión de este se hará de forma objetiva, ya que puede ser difícil en ciertas canciones realizar este trabajo para la mayoría de las personas.

4.4 PLANIFICACIÓN Y ESTIMACIÓN ECONÓMICA

El proyecto tiene una duración aproximada de cuatro meses en total, desde el inicio de este a la finalización incluido la redacción del presente informe. Por este motivo, la planificación se dividirá en los cuatro meses.

4.4.1 PLANIFICACIÓN

<i>Mes 1</i>	
Semana 1	Compra de materiales / Instalación de Software
Semana 2	Grabación / Procesado de audio
Semana 3	Creación de modelo del clasificador
Semana 4	Creación de modelo del clasificador / Creación del programa

Tabla 1- Planificación mes 1

<i>Mes 2</i>	
Semana 1	Testeo de clasificador / Posibles ajustes
Semana 2	Posible aumento dataset / Truncado de dataset
Semana 3	Creación de detector de beats
Semana 4	Testeo de detector de beats

Tabla 2- Planificación mes 2

Mes 3

Semana 1	Prueba de Unicorn Hat
Semana 2	Creación de programa
Semana 3	Testeo del programa
Semana 4	Posibles reajustes

Tabla 3- Planificación mes 3

<i>Mes 4</i>	<i>Redacción de informe / Testeo / Posibles reajustes</i>
--------------	---

Tabla 4- Planificación mes 4

La Tabla 1 define la planificación del primer mes, la primera semana será destinada a la compra de materiales que se utilizarán y que se detallarán en las tablas Tabla 5 y Tabla 6. Durante esa semana también se instalará los softwares que sean necesarios, en concreto en la Raspberry Pi Raspbian y Python, también se incluye la familiarización con ellos. En la segunda semana del primer mes se realizará el programa de grabación del audio, así como la familiarización de las herramientas y librerías de procesado de audio. La semana 3 y 4 se destinarán a la creación del modelo del clasificador, así como el estudio de clasificadores Machine Learning y los conceptos que se deberán tener en cuenta para su testeo, se creará al final de la semana el programa que use el modelo del clasificador para detectar el género de una nueva canción.

La Tabla 2 detalla la planificación del segundo mes, durante este mes se apurará y mejorará el clasificador y se creará el detector de beats. Durante la primera semana del mes, se testeará el clasificador creado el mes anterior y se harán posibles ajustes y mejoras para aumentar la precisión de este. En la segunda semana se manipulará el Gtzan Genre Collection, de modo que se trunquen y corrijan posibles fallos a la hora de clasificar y ajustarlo a la aplicación que se dará en el proyecto. Las últimas dos semanas del mes, se creará el detector de beats

y se testeará propiamente, de forma que sea lo más ajustado al tiempo real posible y lo más preciso posible.

En la Tabla 3, se resume la planificación del tercer mes, en el que durante la primera semana se probará el hardware de luces Unicorn Hat creando programas que hagan que se creen distintos patrones de luces para aprender a usarlo. En la segunda semana, se juntarán todos los programas anteriores (grabador de audio, detector de beats, clasificador y juego de luces) y se unificarán para crear el programa prototipo final, de forma que las dos últimas semanas del tercer mes se guarden para el testeo del programa final y los posibles ajustes que puedan surgir.

El último mes, detallado en la Tabla 4, se reservará para la redacción del presente documento, así como a introducir mejoras al proyecto o como tiempo de reserva ante posibles contratiempos que puedan surgir a lo largo de la instanciación o testeo del proyecto.

4.4.2 ESTIMACIÓN ECONÓMICA

A continuación, se detallará la estimación económica incluyendo tanto la lista de los componentes hardware, como una estimación de la mano de obra que se necesite (trabajo del alumno) que se necesitarán a lo largo del proyecto.

<i>Componente</i>	<i>Fabricante</i>	<i>Cantidad</i>	<i>Coste (€)</i>
Raspberry Pi	Raspberry	1	64.99
Micrófono USB	Gyvazla	1	21.99
Unicorn Hat	Pimorini	1	28.03
Componentes Extra para Raspberry Pi	Bruphny	1	20.99

MicroUSB	Netac	1	6
Total			142

Tabla 5- Coste de componentes Hardware

<i>Componente</i>	<i>Descripción</i>
Raspberry Pi	Tipo de componente: Placa Base Memoria RAM: 4GB Tipo de memoria: LPDDR4 Zócalo de procesador: Broadcom BCM2711B0 Conexiones internas: 2 puertos USB 3.0; 2 puertos USB 2.0
Micrófono USB	Sensibilidad: -30dB±3dB Patente Polar: Omnidireccional SNR: 84 dB Frecuencia de respuesta: 20 Hz-16KHz
Unicorn Hat	8x8 WS2812B Leds RGB Pinout compatible con Raspberry Pi Datos enviados por DMA, no por PWM Cable Micro HDMI a HDMI y adaptador

Componentes Extra para Raspberry Pi	Cable alimentación USB tipo C de 3A Ventilador y disipadores de calor
-------------------------------------	--

Tabla 6- Descripción de componentes Hardware

En la Tabla 5 se muestra el coste de los componentes hardware que se utilizarán, además de la suma como coste total del coste de todos los componentes y en la Tabla 6 se muestra una descripción del hardware que se listo anteriormente. Además del coste de los componentes hardware, también se puede incluir el trabajo en forma de mano obra. Estimando que la mano de obra puede ser de 35 €/ hora y que se puede trabajar unas 15 horas/semana el cálculo sale:

$$35\text{€/hora} * 10 \frac{\text{horas}}{\text{semana}} * 16 \text{ semanas} = 5600\text{€}$$

Capítulo 5. DICCIONARIO DE TÉRMINOS

En este apartado se desarrollarán y explicarán conceptos o términos que se mencionarán y utilizarán en los capítulos posteriores de Sistema/Modelo Desarrollado y Análisis de resultados.

Activition: La activación se realizará con la función ReLU (Rectified Linear Unit). La función ReLU es una función muy utilizada en las capas intermedias de redes neuronales. ReLU funciona de forma que aplanar la respuesta a todos los valores negativos a 0, mientras deja todo sin cambios para valores iguales o mayores que 0. Se trata de una función simple y eso ayuda muy bien para las capas intermedias, ya que en estas capas la cantidad de pasajes y cálculos es muy grande. (Equipo Paradigma Digital, 2020)

Aprendizaje autónomo: Al enfrentarse a un problema de aprendizaje automático, al final estamos intentando modelar una función o una respuesta de una variable que denominamos “objetivo” en base otra serie de variables que llamamos “atributos”. Esto es lo que ocurre en el caso del clasificador, en el que los atributos son las imágenes y los colores de estas en los espectrogramas y el objetivo es clasificarla según el parecido de los atributos al género que corresponda. (Equipo Paradigma Digital, 2020)

Batch: Se entiende como batch o lote al grupo de muestras de entrenamiento, una vez que se inicia el entrenamiento, para poder ir actualizando el cálculo de pesos sin tener que esperar a que el modelo haya terminado el entrenamiento, se toma, una muestra de cada clase menor al número total, pero aleatoria. Se va entrenando el modelo con este lote, se van actualizando los pesos y una vez hecho, se pasa al siguiente lote (batch) reiteradamente hasta que se haya procesado todo el conjunto de entrenamiento. (Equipo Paradigma Digital, 2020)

Conjunto de entrenamiento: Se trata de una división del dataset que se usará para que los algoritmos se autoajusten para acercarse a su objetivo. Suele destinarse una mayoría del dataset para el conjunto de entrenamiento. (Equipo Paradigma Digital, 2020)

Conjunto de test: El conjunto de prueba o conjunto de test es otra división del dataset que se usará para que, una vez entrenado el algoritmo, se compruebe el funcionamiento de este. Este conjunto del dataset, no será utilizado por el algoritmo antes de que ya se haya entrenado. (Equipo Paradigma Digital, 2020)

Conjunto de validación: El conjunto de validación se trata de una parte del conjunto de test utilizado si existe más de un algoritmo a evaluar o hay que hacer ajustes del conjunto de test, de forma que los hiperparámetros se ajustan con una sección de entrenamiento y las distintas opciones se compraran entre sí usando este conjunto. (Equipo Paradigma Digital, 2020)

Conv2d: Se trata de una instrucción que introduce una capa convolucional para representaciones de dos dimensiones, que será activado con la función que se indique en Activation. (Burgal, 2018)

Dataset: Un dataset o un conjunto de datos, es una representación de datos residente en memoria que proporciona un modelo de programación relacional coherente independiente del origen de datos que contiene. Se trata básicamente de una agrupación de registros, donde cada registro a su vez es una agrupación de pares clave valor que definen a un dato individual. (Balagueró, 2018)

Dense: Se trata de instrucción con la que se añaden las capas normales, la primera capa será una capa oculta que tendrá 512 nodos y usará ReLU como activación, mientras que la última capa, será la capa de salida. La capa de salida tendrá 10 nodos correspondientes a los 10 géneros que se quieren predecir. Esta última capa de salida usará como activación la función softmax, la cual calcula la distribución de probabilidades del evento en “n” diferentes eventos, en el sistema serán 10 diferentes eventos. La función, calculará las probabilidades del intervalo (0,1) de cada clase objetivo, se seleccionará la clase con el valor más alto de probabilidad. (Sharma, 2017)

Dropout: Se trata de un método de desactivación que se usará para la MaxPooling2D, su función es la de desactivar un número de neuronas de una red neuronal de forma aleatoria. En cada iteración, esta capa desactivará diferentes neuronas que no se tomarán en cuenta ni

para el forward propagación ni para el backward propagación lo que hace que las neuronas cercanas no dependan tanto de las neuronas desactivadas. Ayuda, al igual que en la MaxPooling2D, ayuda a reducir el sobre entrenamiento ya que las neuronas cercanas suelen aprender patrones que se relacionan y estas relaciones pueden llegar a formar un patrón muy específico con los datos de entrenamiento. Dropout hace que la dependencia sea menor, así que las neuronas necesitaran trabajar mejor solitariamente y no depender tanto de las relaciones con las neuronas vecinas. (Rodríguez, 2018)

Especificidad: La especificidad o la tasa de Verdaderos Negativos, se trata de los casos negativos que el clasificador ha detectado correctamente. Básicamente consiste en expresar cuan bien puede detectar el modelo los negativos. (Arce, 2019)

Exactitud: Se trata de una métrica que indica cómo de preciso es un modelo de Machine Learning a la hora de hacer predicciones. Se define como la ratio entre las predicciones correctas respecto al número total de ejemplos del dataset. Cuanto mayor sea la exactitud, más aciertos obtendrá el modelo y por tanto será más preciso. (Arce, 2019)

F1 Score: Se trata de un parámetro que combina la precisión y la sensibilidad en una sola métrica, es por esto por lo que es de gran utilidad cuando la distribución de las clases es desigual. Permite distinguir modelos que tengan alta precisión y sensibilidad de modelos que tenga cualquiera o ambos parámetros bajos. (Arce, 2019)

Feature: Al enfrentarse a un problema de aprendizaje automático, al final estamos intentando modelar una función o una respuesta de una variable llamada “objetivo” en base otra serie de variables denominadas “atributos”, los atributos son lo que se denominan features. (Equipo Paradigma Digital, 2020)

FFT (Fast Fourier Transform): La FFT o Transformada Rápida de Fourier es una herramienta fundamental en el procesado digitales de señales, se trata de un algoritmo para el cálculo de la Transformada Discreta de Fourier, es muy útil por eliminar una gran parte de los cálculos repetitivos a que está sometida la transformada. (Sastrón, 2017)

Flatten: Se trata de una instrucción que convierte los elementos de la matriz de imágenes de entrada en un array plano, se suele utilizar antes de usar la instrucción Dense.

Frames: Se conoce como frames o data frames a estructuras de datos de dos dimensiones que pueden contener datos de diferentes tipos, por lo tanto, se pueden considerar estructuras heterogéneas. Es una estructura muy usada para el análisis de datos y para el tratamiento de paquetes estadísticos. (Peng, 2016)

Epoch: Se conoce como epoch o época a un hiperparámetro que define el número de veces que el algoritmo de aprendizaje funcionará en todo el conjunto de datos de entrenamiento. Una época significa que cada muestra en el conjunto de datos de entrenamiento ha tenido la oportunidad de actualizar los parámetros internos del modelo. Cada época se compone de uno o más lotes. El número de épocas debe ser el suficiente para que el error del modelo se haya minimizado lo suficiente. (Equipo Paradigma Digital, 2020)

InputLayer: Se trata de la capa de entrada, es la capa donde los atributos de entrada serán procesados tal cuál llegan después de haberse computado los melspectrogramas. En el modelo del clasificador, constará de 128x129 neuronas de entrada, correspondiente a las 128 escalas de mel x 129 escalas de tiempo.

Hiperparámetros: Los hiperparámetro de un modelo de aprendizaje forman una configuración cuyo valor no puede ser estimado por los datos y se debe establecer antes del proceso de entrenamiento. En los hiperparámetros, se incluye el número de iteraciones, el número de capas y neuronas de una red neuronal o la participación de los datos en los conjuntos de entrenamiento, validación y prueba. (Equipo Paradigma Digital, 2020)

Matriz de confusión: Se trata de una tabla que a menudo se usa para describir el rendimiento de un modelo de clasificación o clasificador en un conjunto de datos de test para los que se conocen los valores verdaderos. De la matriz de confusión se pueden obtener métricas como la exactitud, la precisión, la sensibilidad, la especificidad y por último la F1 Score. (Arce, 2019)

Maxpooling2D: Se trata de una instrucción en la que se introduce la capa de reducción el número de neuronas después de las convoluciones, de forma que el número de neuronas de las capas posteriores no se hagan muy grandes. (Na8, 2018)

Pérdida: Se trata de una función que se aplica sobre un conjunto de variables para dar lugar a un valor numérico indicativo de la calidad del ajuste del modelo usado, con respecto a los valores reales cuyo comportamiento es el que se pretende modelar.

Se busca minimizar la función de pérdida de forma gradual, en distintas iteraciones, hasta conseguir el mínimo error dentro de unos determinados márgenes que permitan que el modelo se generalice (evitando el sobre entrenamiento). (Equipo Paradigma Digital, 2020)

Precisión: Se refiere a la dispersión del conjunto de valores obtenidos a partir de medidas repetidas de una magnitud. Cuanto menor sea la dispersión mayor será la precisión. Se representa por la razón entre el número de predicciones correctas (tanto positivas como negativas) y el total de predicciones. (Arce, 2019)

Tasa de entrenamiento: La tasa de entrenamiento o en inglés, learning rate, es un hiperparámetro que se encarga de actualizar los modelos en el proceso de entrenamiento, se trata de un valor escalar positivo que oscila en 0 y 1, usado de forma que, durante cada iteración del proceso de entrenamiento, actualiza el valor de los pesos y sesgos en la dirección que minimiza la pérdida. (Equipo Paradigma Digital, 2020)

Sensibilidad: La sensibilidad o la Tasa de Verdaderos Positivos, es la proporción de casos positivos que fueron acertados por el algoritmo. (Arce, 2019)

Capítulo 6. SISTEMA/MODELO DESARROLLADO

Esta sección describe el desarrollo del proyecto siguiendo una estructuración en fases, permitiendo la división del proyecto en subconjuntos lógicos para facilitar la explicación. El desarrollo se dividirá en varias fases, cada fase consta de diferentes apartados: Fundamentos del sistema, Diseño e Implementación. (García, 2014)

En Fundamentos del sistema, se definirán una serie de conceptos previos que deben establecerse como paso previo a las fases siguientes.

En Diseño, se realizará una esquematización para facilitar el proceso de desarrollo de la aplicación, a la vez que sirve para entender el funcionamiento del sistema.

En Implementación, esta parte se dividirá en dos una parte conocida como estado del arte, donde se describirán las herramientas y librerías que se usarán y otra parte conocida como pasos a seguir, donde se describen los pasos seguidos en el desarrollo de la fase que corresponda en ese momento. (Rubén Gómez Fuentes, Daniel Lago Agudo, 2016)

6.1 CAPTURA DE DATOS

6.1.1 FUNDAMENTOS DEL SISTEMA

Un micrófono, es un dispositivo capaz de detectar la magnitud física del sonido, el sonido es una vibración que se programa como una honda mecánica de presión que se desplaza a través de un medio físico como puede ser el aire o el agua. Los dispositivos de grabación imitan el proceso con el que los seres humanos recibimos y analizamos los sonidos, convirtiéndola en una señal eléctrica.

En un micrófono, el primer componente eléctrico al encontrar esta señal se traduce en una señal de tensión analógica, la cuál es continua. La señal continua no es tan útil en el mundo

digital, así que antes de ser procesada, debe ser traducida en una señal discreta que se pueda almacenar digitalmente.

Para convertir la señal analógica a la señal digital, será necesario el uso del Teorema de Nyquist-Shannon, esta teoría dice que la tasa de muestro que se debe usar para no notar saltos en la continuidad y no perder frecuencias para reconstruir la señal original debe ser del doble de la frecuencia máxima que se puede percibir. (Jovanovic, 2015)

El oído humano puede percibir hasta 20- 22 KHz de frecuencia. Para poder captar todas las frecuencias que un ser humano puede oír en una señal de audio, se va a seleccionar una velocidad de muestreo de 44100Hz. (Refraction Productions)

Este apartado de Captura de datos será utilizado en dos apartados posteriores y de forma distinta. Para el Clasificador, se grabará un audio de una duración de segundos determinada y se guardará en un archivo “.wav”, en este apartado se describirá este uso puesto que es más sencillo. Para el Detector de beats, la Captura de datos se usará de forma que capte pequeños cambios en el audio, de esta forma el micrófono grabará pequeños audios de Diseño segundos y que además se guarden de forma que se tengan en cuenta más adelante. Se detallará más en el apartado de Detector de beats.

6.1.2 DISEÑO

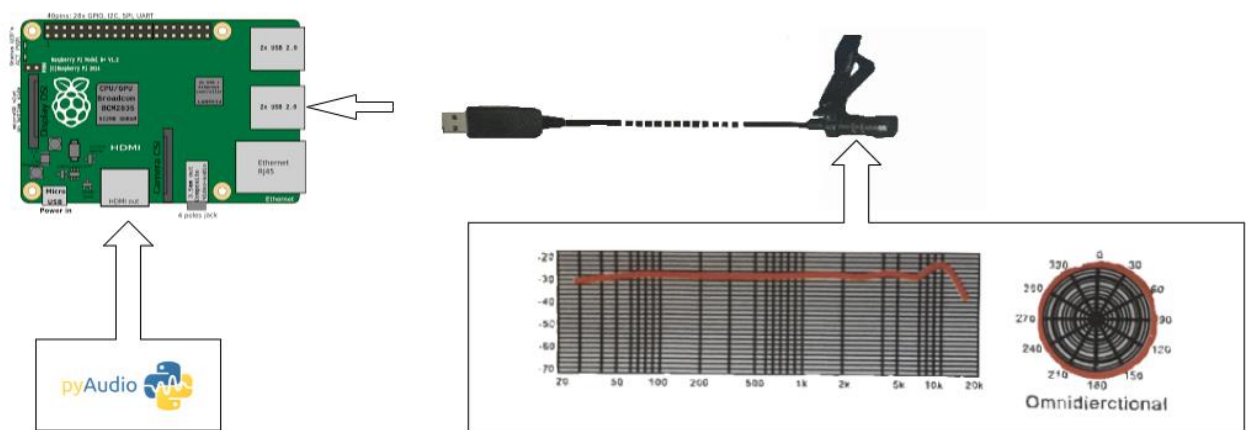


Ilustración 6- Esquema captura de audio

En la Ilustración 6, se puede apreciar un esquema de lo que será la fase de captura de datos, junto con el rango de frecuencias que permite grabar el micrófono intrínseco en las características de este. Además, es destacable que el micrófono permite las velocidades de muestreo de 8 KHz, 11.0592 KHz, 22.05 KHz, 44.1 KHz Y 48 KHz, entre las que se incluye la frecuencia y mencionado anteriormente de 44.1 KHz y ya se ha detallado en el apartado anterior de Fundamentos del sistema. También se puede observar en la Ilustración 6 que el micrófono es un micrófono con conexión USB.

Puesto que el micrófono que se ya tiene incorporado un convertidor analógico-digital se debe seleccionar únicamente la frecuencia de muestreo, el número de canales (mono o estéreo) y el tamaño de la muestra. En el caso del proyecto tal y como se ha indicado en el apartado de Fundamentos del sistema la frecuencia de muestreo será de 44100 Hz, el audio se grabará en mono por las características del micrófono y el tamaño de las muestras serán de 16 bits. Después se guardarán las muestras en un archivo “.wav” como una matriz de bytes.

6.1.3 IMPLEMENTACIÓN

6.1.3.1 Estado arte

Se utilizarán las librerías Python pyaudio y wave para la implementación del programa. La librería pyaudio es una librería de audio I/O multi-plataforma y open source, utilizada para capturar señales de audio a través de micrófonos, las señales capturadas se pueden visualizar en su representación temporal y su representación frecuencial. La librería wave es una librería estándar de python muy usadas para grabar audios en formatos “.wav”. Las funciones en este módulo pueden escribir datos de audio en formato raw en un archivo y permite también atributos en un archivo “.wav”.

6.1.3.2 Pasos a seguir

1. Se recogerán los datos y las variables indicados en el apartado de Diseño, dichas variables serán el número de canales, la velocidad de muestreo, el formato de resolución, las muestras por buffer y los segundos que se grabarán.
 2. Una vez creadas e inicializadas las variables que van a ser necesarias, se creará una
-

variable PyAudio(), de la librería pyaudio referida en la sección Estado arte y se abrirán introduciéndole las variables formato, velocidad de muestreo, número de canales, las muestras por buffer y que es un audio de entrada.

3. Se creará un array de frames en el cual se guardarán los datos que pyaudio vaya leyendo, se agregarán los datos recogidos hasta los segundos que se indicaron.
4. Una vez se termine se cerrarán las variables y se guardará el array frames en el archivo “.wav” que se desee gracias al uso de la librería wave, se indicará el número de canales, el formato y la velocidad de muestreo usado.

6.2 CLASIFICADOR

6.2.1 FUNDAMENTOS DEL SISTEMA

En este apartado, se creará el modelo del clasificador, así como el programa que use el modelo creado para clasificar las canciones capturadas gracias al apartado de Captura de datos. Para crear el modelo del clasificador se usará, como ya se ha mencionado anteriormente el data set Gtzan Genre Collection, dicho dataset, entre otras cosas se escoge porque los fragmentos fueron recogidos de distintas fuentes, haciendo que exista una variación en las condiciones de grabación de cada una.

El dataset, sin embargo, no es el original, se le han añadido una serie de canciones, en las mismas condiciones en las que posteriormente se clasificará, de forma que se favorezca a los géneros con los que el clasificador tiene más problemas, se penalice a los géneros que dan más falsos positivos y se actualice el dataset. Para grabar las nuevas pistas, se ha utilizado el programa de Captura de datos descrito en el apartado anterior. El número de pistas añadidas a cada género se encuentra detallado en la Tabla 7.

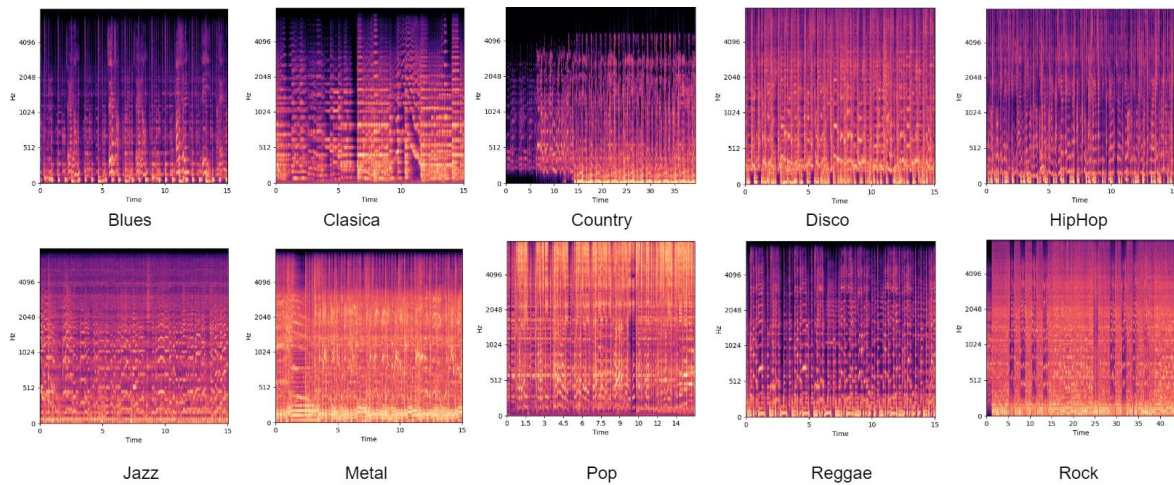
<i>Género</i>	<i>Pistas Añadidas</i>
Pop	50

Rock	40
Reggae	20
Metal	10
Jazz	30
Hip Hop	40
Dance	40
Country	10
Clasica	34
Blues	10

Tabla 7- Pistas añadidas a cada género

A la hora de crear el clasificador, aparece un problema grande a abordar, y este es que *feature* se va a usar, es decir, que atributo o característica se usará. En el caso que atañe, un atributo, que viene perfecto es el Melspectrograma, desarrollado en el capítulo de Descripción de las Tecnologías.

El Melspectrograma se puede ver como una representación visual de una señal de audio, en concreto, representa el espectro de frecuencias a lo largo del tiempo, por tanto, engloba todas las características del audio en una imagen.



MELSPECTROGRAMAS POR GÉNEROS

Ilustración 7- Melspectrogramas por géneros

<i>Género</i>	<i>Título</i>	<i>Autor</i>	<i>Tiempo (s)</i>
Blues	It Serves You Right To Suffer	John Lee Hooker	30
Clásica	Symphony No.39 In EFlat Major, k543:III Menuetto (Allegro)	Mozart	30
Country	Old Town Road	Lil Nas X ft. Billy Ray Cyrus	78
Disco	Playboy (Be me)	La Toya Jackson	30
Hip-hop	Stressed Out	A Tribe Called Quest & Faith Evans	30
Jazz	Forth Worth	Joe Lovano	30
Metal	Gypsy	Dio	30
Pop	A Dónde Vamos	Morat	30

Reggae	Roots, Rock, Reggae	Bob Marley & The Wailers	30
Rock	American Idiot	Green Day	86

Tabla 8- Relación Melspectrogramas con géneros.

En la Ilustración 7, se muestran los melspectrogramas de los diferentes géneros, se puede apreciar como varían las representaciones según el género que se esté tratando y cómo algunos géneros tienen más similitudes que otros, esos géneros serán los que el clasificador tenga mayor dificultad a la hora de clasificar. En la Tabla 8, se muestran las canciones, el autor y el tiempo de la pista utilizadas para crear los melspectrogramas de la Ilustración 7.

Básicamente, lo que se hará es transformar el problema en un problema de clasificación de imágenes. Esto es perfecto porque encaja perfectamente con los modelos de redes convolucionales CNN.

Las redes neuronales convolucionales consisten en múltiples capas de filtros convolucionales de una o más dimensiones. Al poder trabajar con matrices bidimensionales son muy efectivas para tareas relacionadas con la visión artificial. El objetivo de la aplicación de un filtro convolucional es la de transformar los datos de forma que ciertas características de la imagen se vuelvan más predominantes, obteniendo como resultado un conjunto de datos menos sensibles a pequeñas modificaciones de los datos de entrada.

Antes de construir el modelo, hay que realizar una serie de pasos:

1. Escalar los valores de los melspectrogramas de manera que queden entre 0 y 1 para mejorar la eficiencia a la hora de computar los datos.
2. Se deberá introducir los valores haciendo que sean aleatorios, de forma que al introducir aleatoriedad al conjunto de datos, se evita el sobre entrenamiento del conjunto de entrenamiento.

6.2.2 DISEÑO

A la hora de crear el modelo, se requiere tomar una serie de decisiones, para empezar, se debe elegir cuál será el porcentaje de datos que va a usarse para el conjunto de entrenamiento y de prueba.

En Machine Learning, el conjunto de entrenamiento son los datos con los que los algoritmos se autoajustan para acercarse a su objetivo. Una vez entrenado el sistema, su funcionamiento se comprueba con otro conjunto de datos conocido como conjunto de prueba. Si hay más de un algoritmo a evaluar o hay que hacer ajustes de hiperparámetros, como es el caso, se puede usar una parte del conjunto de entrenamiento como conjunto de validación, de forma que los hiperparámetros se ajustan con una sección de entrenamiento y las distintas opciones de compraran entre sí usando el conjunto de validación.

En este caso, se usará el 30 % del conjunto de datos como conjunto de prueba, mientras que el conjunto de validación se irá ajustando, de forma que va a ir decreciendo a medida que el algoritmo se vaya haciendo más preciso. En la Ilustración 8, muestra un diagrama de como quedarán los conjuntos de datos seleccionados.

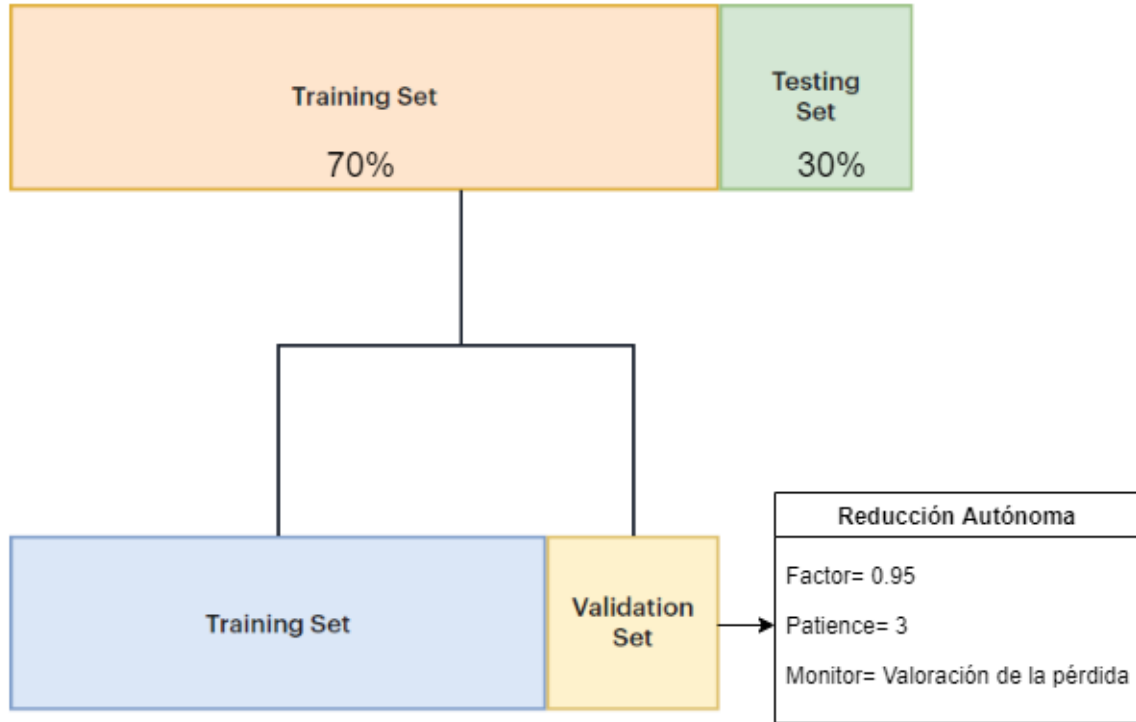


Ilustración 8- Diagrama de los conjuntos de datos

La arquitectura para el modelo CNN será la que se muestra en la Ilustración 9. Se puede apreciar el número de parámetros que tendrá cada capa del modelo. Se puede observar, además el número total de parámetros, calculado como la suma de los parámetros de cada capa, el cuál es de 2495114.

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 128, 129, 1)]	0
conv2d (Conv2D)	(None, 128, 129, 16)	160
activation (Activation)	(None, 128, 129, 16)	0
max_pooling2d (MaxPooling2D)	(None, 64, 64, 16)	0
dropout (Dropout)	(None, 64, 64, 16)	0
conv2d_1 (Conv2D)	(None, 64, 64, 32)	4640
activation_1 (Activation)	(None, 64, 64, 32)	0
max_pooling2d_1 (MaxPooling2D)	(None, 32, 32, 32)	0
dropout_1 (Dropout)	(None, 32, 32, 32)	0
conv2d_2 (Conv2D)	(None, 32, 32, 64)	18496
activation_2 (Activation)	(None, 32, 32, 64)	0
max_pooling2d_2 (MaxPooling2D)	(None, 16, 16, 64)	0
dropout_2 (Dropout)	(None, 16, 16, 64)	0
conv2d_3 (Conv2D)	(None, 16, 16, 128)	73856
activation_3 (Activation)	(None, 16, 16, 128)	0
max_pooling2d_3 (MaxPooling2D)	(None, 8, 8, 128)	0
dropout_3 (Dropout)	(None, 8, 8, 128)	0
conv2d_4 (Conv2D)	(None, 8, 8, 256)	295168
activation_4 (Activation)	(None, 8, 8, 256)	0
max_pooling2d_4 (MaxPooling2D)	(None, 4, 4, 256)	0
dropout_4 (Dropout)	(None, 4, 4, 256)	0
flatten (Flatten)	(None, 4096)	0
dropout_5 (Dropout)	(None, 4096)	0
dense (Dense)	(None, 512)	2097664
dropout_6 (Dropout)	(None, 512)	0
dense_1 (Dense)	(None, 10)	5130
Total params: 2,495,114		
Trainable params: 2,495,114		
Non-trainable params: 0		

Ilustración 9- Resumen del modelo

6.2.3 IMPLEMENTACIÓN

6.2.3.1 Estado del arte

Para implementar el clasificador, será necesario el uso de varias librerías de python. Para conseguir los melspectrogramas, se utilizará la librería librosa que es una librería para el análisis de audio y música. (Brian McFee, 2015) . También se usará la librería NumPy que es una librería utilizada para trabajar con vectores y matrices multi dimensionales, incluye un gran número de funciones matemáticas para operar con ellos. La librería librosa también utiliza NumPy a la hora de operar con los audios. (Johari, 2019) (Brian McFee, 2015)

Otra librería importante que se usará será sklearn, esta librería cuenta con algoritmos de clasificación, regresión, clustering y reducción de dimensionalidad. Además, presenta la compatibilidad con otras librerías de Python como NumPy, SciPy y matplotlib. En el programa será muy importante a la hora de dividir los conjuntos de entrenamiento y prueba, además de ser una herramienta fundamental para el análisis y modelado estadístico de los resultados. (Universidad de Alcalá)

Hay que destacar, que el modelo se guardará como un archivo “.h5”, por tanto, se deberá importar la librería h5py. El formato h5 (Hierarchical Data Format) se trata un formato de datos que contiene matrices multidimensionales de datos científicos, se usa fundamentalmente para almacenar datos numéricos, gráficos o de texto. (File Extension, s.f.)

Por último, se importará la librería más importante y la que más ayudará tanto a la hora de crear el modelo como a la hora de implementar el programa que detecte un nuevo género. Esta librería es TensorFlow y su explicación se encuentra detallada en el apartado de Descripción de las Tecnologías.

En la Ilustración 10 se muestra un resumen con los logos de las principales librerías que se usarán para el clasificador.



Ilustración 10- Logos de las principales librerías usadas

6.2.3.2 Pasos a seguir

En este apartado se describirá la implementación del programa final que clasificará una nueva canción usando el modelo creado, no se describirá el programa que cree el modelo por su dificultad y amplitud, sin embargo, en el anexo II y metido en un github con todo el proyecto, se mostrará si se desea obtener el código para más información.

1. Como ya se ha mencionado anteriormente, el programa recibirá un audio “.wav” en el que se encuentra grabado el audio que se pretende clasificar. Dicho audio se procesará usando la librería librosa, la cual cargará la canción usando load que es un método que carga el archivo “.wav” y devuelve la velocidad de muestreo y un vector con el audio en función del tiempo.
2. Una vez cargado y teniendo la señal en el vector, se procederá a dividir dicho vector para crear una matriz de forma que tenga la forma necesaria para que el melspectrograma final pueda ser comparado de la misma forma que los melspectrogramas que se usaron en el modelo (independientemente del tamaño de las pistas de audio que se empleen.

3. Una vez divididas las canciones y habiendo sido ajustadas, se calcula el melspectrograma con las funciones de melspectrogram de librosa, a la cual se le introduce la señal de la que se desea calcular el melspectrograma, la longitud de la ventana del espectro en frecuencia, que en este caso será de 1024 y una variable que es el número de muestras entre “frames” sucesivos, se seleccionarán 256 muestras.
4. Una vez que se tenga el melspectrograma listo, usando el método load_model() de la librería TensorFlow, se cargará el modelo que vendrá dado en el archivo “.h5” y que contendrá todos los parámetros ya entrenados.
5. Una vez cargado se llamará al método predict() de TensorFlow introduciendo el melspectrograma calculado anteriormente.
6. Las predicciones serán un vector que contendrá las probabilidades que se le da a cada género, para seleccionar se cogerá la posición del vector con el valor más alto, de esta forma se obtiene el género que más posibilidades el clasificador le ha dado.

La relación de las posiciones del vector de probabilidades devuelto al predecir con los géneros viene representada en la Ilustración 11.

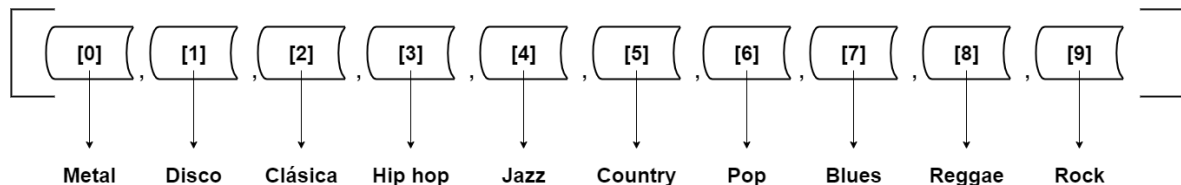


Ilustración 11- Posiciones de probabilidades con géneros

De esta forma si el vector selecciona que en la posición 4 tiene el valor más alto, es decir, la probabilidad más alta, el género seleccionado será jazz y así con el resto de los géneros.

6.3 DETECTOR DE BEATS

6.3.1 FUNDAMENTOS DEL SISTEMA

El tempo es el ritmo o compás de una acción. Si hablamos en el ámbito musical, se trata del ritmo de una canción o pieza musical. Este se normalmente se mide en BPM (*Beats per minute* o pulsos por minuto), se trata de una medida global en la canción que pretende cuando

se producirá el próximo beat, pero no es una medida local que te indique el momento exacto del beat. Nuestra aplicación se centra más en un análisis local para detectar el momento exacto en que se produce.

La detección de los beats en una pista de audio es un ejercicio muy sencillo para el oído humano, pero a la hora de formalizarlo se convierte en una tarea bastante complicada.

Una persona escucha un sonido y lo transforma en una señal eléctrica que el cerebro interpreta, esta señal tendrá más o menos energía en función del sonido escuchado. Una persona detectará un beat cuando se produzca un aumento significativo de la energía del sonido y por un periodo relativamente breve. De este modo, se podría decir que los beats dependen de un historial de cantidad de energía. Se puede decir que un beat sucede cuando el instante de energía supera en cierta cantidad a la media de energía recibida últimamente.

El detector de beats también tendrá la capacidad de detectar cuando una canción se haya parado, de forma que cuando el audio que se reciba sea prácticamente cero o muy cercano a cero, se tomará como que se ha parado el audio y que comenzará una nueva canción. Al detectar el inicio de una nueva canción o más bien el final de la canción que estaba sonando, se reiniciarán todos los valores y el programa comenzará de nuevo.

6.3.2 DISEÑO

Como se ha dicho en el apartado de Fundamentos del sistema, el detector de beats, lo que hará será encontrar el instante en el que el sonido supere en cierta cantidad a la media de energía recibida últimamente, para ello se utilizará la transformada de Fourier a cada pequeño instante de tiempo grabado cada 0.01 segundos y la pista se irá comparando con todos los 0.01 segundos grabados anteriormente.

Para ello será necesaria el cálculo la FFT de cada instante recogido por el micrófono, de la FFT se calculará la media de las amplitudes que salgan de la FFT, también las amplitudes que sean más bajas serán necesarias, calculando la media de esas amplitudes. Se irán acumulando de forma que cada pequeño audio detectado, tendrá las medias de las bajas

amplitudes de los espectros de los audios grabados antes, además de las medias de su FFT y de las bajas amplitudes de su FFT.

También se calculará una variable llamada media base de FFT, la cual se calculará cogiendo la mitad de los valores de las amplitudes del espectro cuando se han quitado los valores grandes, será una variable fundamental a la hora de comparar.

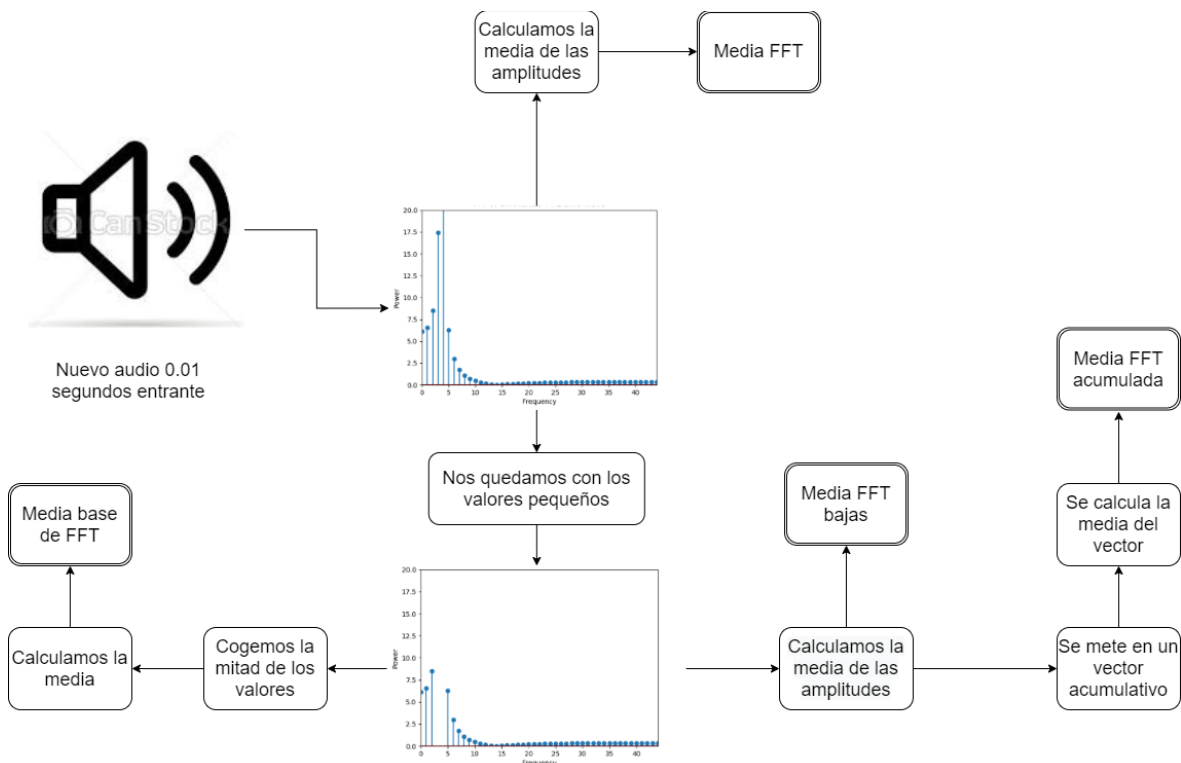


Ilustración 12- Cálculo variables detector de beats

En la Ilustración 12, se muestra un esquema de cómo se calcularán las variables que serán necesarias para tomar la decisión de que el audio que entre sea beat o no. Cuatro variables van a ser las que se deberán calcular:

1. La media de los valores del espectro de frecuencias o Media FFT.
2. La media de los valores del espectro quitando las frecuencias altas, se considerarán frecuencias altas a valores de más de 1000, que se llamarán Media FFT bajas

3. A la Media FFT bajas, se introducirá en un vector acumulativo de medias, donde estarán las medias de audio anteriores y la del nuevo. Se calculará la media del vector obteniendo de esta forma la Media FFT acumulada.
4. La variable Media base de FFT descrita anteriormente.

Estas cuatro variables descritas anteriormente serán las que se usarán para tomar las decisiones que se deseen. La toma de decisiones se resume en la Ilustración 13, donde se muestra el árbol de decisión de si es un beat o no tomando las variables calculadas con la Ilustración 12.

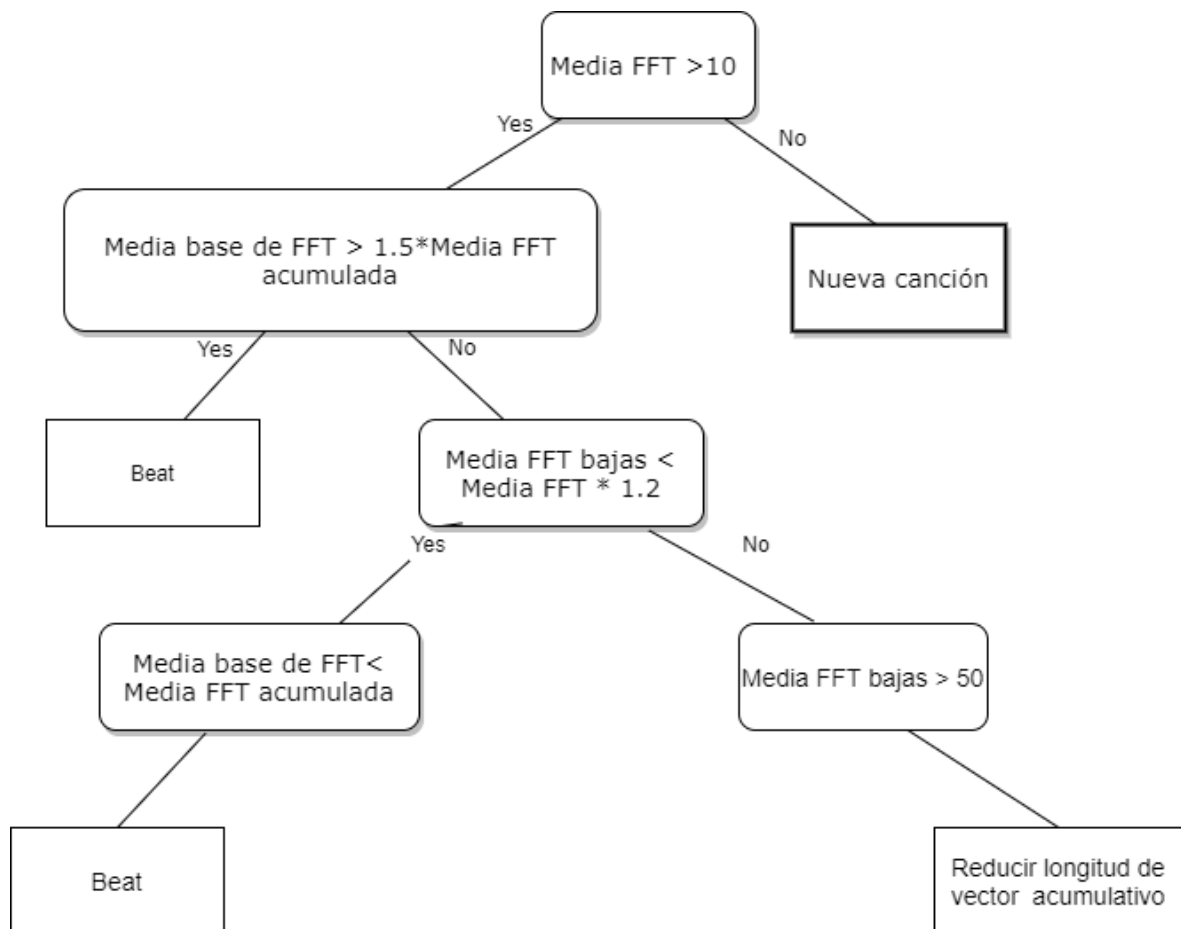


Ilustración 13- Árbol de decisión de detector de beats

Se detectará una nueva canción cuando la media del audio sea menor que 10, se considerará que se está en silencio y por tanto que la canción se ha parado o no está sonando.

Sin embargo, los cálculos para detectar el beat, todavía no se han completado del todo habrá que hacer más cálculos para afirmar que se tiene un beat y para calcular los beats por minuto.

Se tendrá en cuenta el tiempo que pasa entre que se detecta un beat y otro, de forma que cuando el tiempo sea demasiado pequeño, se descarte como beat (pues no tiene mucho sentido que se detecte un beat cada 0.01 segundos). Se creará también un vector con los beat o mejor dicho con la diferencia entre dos beats detectados, de esta forma, se introduce una segunda forma de controlar que los beats se detecten de forma más o menos constantes a lo largo de la canción.

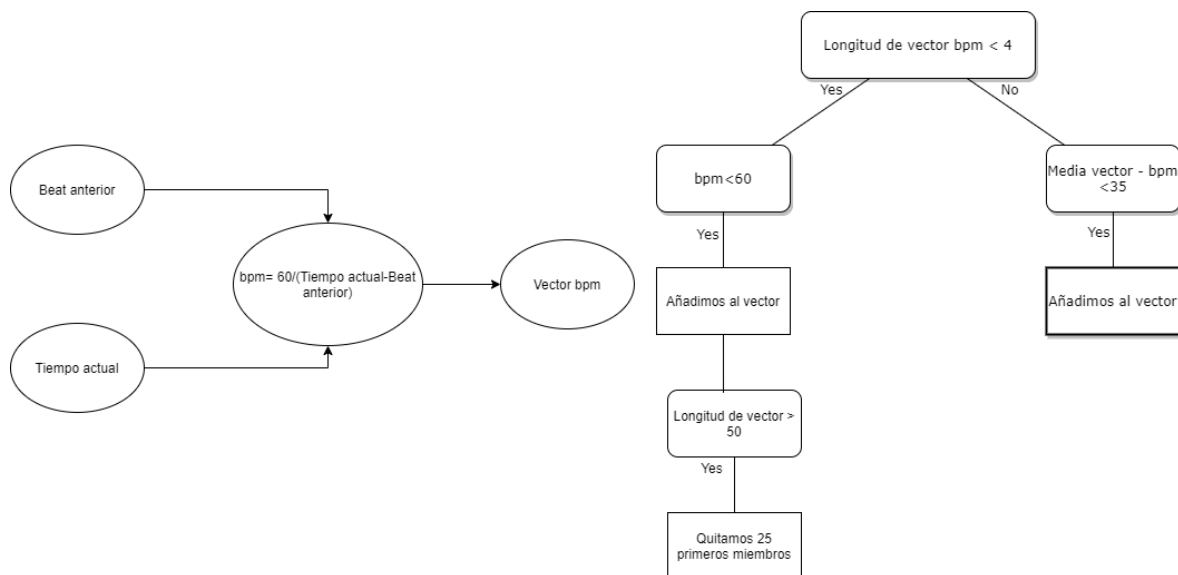


Ilustración 14- Resumen vector bpm

En la Ilustración 14, se muestra un esquema de cómo se irá calculado el vector bpm, para así poder tener en cuenta todas las variantes que puedan aparecer en el problema.

6.3.3 IMPLEMENTACIÓN

6.3.3.1 Estado arte

Como ya se ha visto la detección de beats, no es un cálculo trivial y se debe trabajar con vectores y tener especial cuidado con el trato de estos. También todos los cálculos deben ser prácticamente instantáneos.

Para implementar el programa, se usarán librerías que ya se han visto y analizado en las partes anteriores, como las que se incluyen numpy y pyaudio, que ayudarán en el trabajo con vectores y para grabar la música respectivamente.

En esta parte también se usarán librerías no usadas anteriormente como threading que es una librería para crear threads que será bastante útil para hacer que el programa pueda estar escuchando audios constantemente mientras que a la vez va haciendo los cálculos necesarios. También se usarán otras librerías como keyboard para parar el programa al pulsar el teclado o time para conseguir el tiempo actual, que es importante a la hora de calcular el vector bpm.

6.3.3.2 Pasos a seguir

A la hora de implementar el programa se deben calcular los valores que se han introducido en el apartado de Diseño para ello se seguirán los siguientes pasos:

1. Se crearán las variables iniciales, es decir, una clase creada en otro archivo llamada InputRecorder() y se activará, iniciando el thread que llama al método de grabación.
 2. Mientras que no se indique otra cosa, se irán detectando los beats, para lo que se crearán y calcularán las variables necesarias.
 3. Se calculará la fft de la pequeña señal de 0.01 segundos capturada el método de grabación usado con el thread.
 4. Se calculará la media de la fft obtenida anteriormente.
 5. Se obtendrán los valores menores de 1000 Hz, de las bajas frecuencias.
 6. Se calculará la media de los valores obtenidos en el paso anterior.
 7. Se añadirán los valores de la media de bajas frecuencias y la media de la fft a los vectores de media acumuladas y de media baja acumulada.
 8. Una vez tenidos todos los valores, se tomarán las decisiones descritas en la Ilustración 13 y en la Ilustración 14, esto se hará repetitivamente hasta que se pulse una tecla del teclado.
-

6.4 SALIDA DE LAS LUCES

6.4.1 FUNDAMENTOS DEL SISTEMA

Se trata de la última parte del proyecto y en la que se ven los resultados que se han recogido y analizado en los apartados anteriores, se tratará de hacer que leds RGB ayuden a mostrar el género en el que se ha clasificado la canción que está sonando mediante el color y la forma en la que luzcan.

Para mostrar el resultado del detector de beats, se hará que los leds luzcan y se apaguen a la frecuencia en que el detector detecte los bpm de la canción, así se hará que los leds brillen al ritmo de la canción y además, si el usuario lo desea identificar el género de la canción.

En esta parte también se incluirá el ensamblamiento de todos los apartados anteriores, de forma que corriendo este programa python se correrá el conjunto del sistema global.

6.4.2 DISEÑO

Como se ha introducido en el apartado de Fundamentos del sistema, en esta parte se incluye tanto la implementación del juego de luces, como el ensamblamiento de todo el sistema, en la Ilustración 15 se muestra un esquema de cómo será el diseño de la salida de luces.

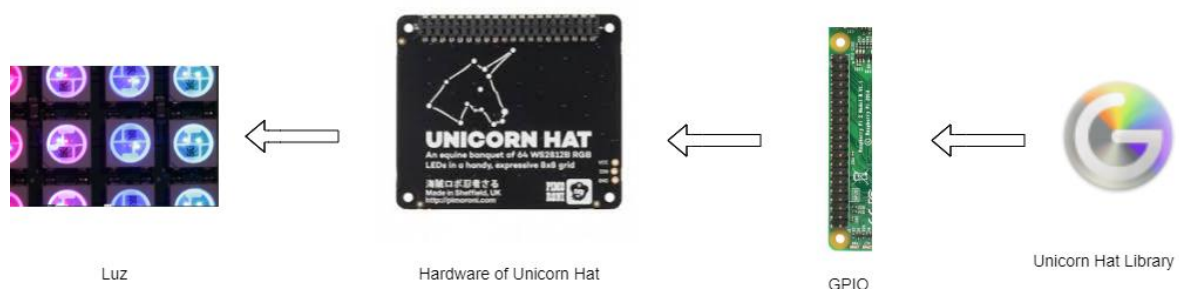


Ilustración 15- Esquema de salida de luces

A la hora de decidir cómo se representarán los géneros, se ha decidido la mezcla de figuras representativas con el uso de colores, ya que como la placa de Unicorn Hat permite usar

colores al tratarse de leds RGB. También se ha decidido usar figuras porque la placa tiene 64 leds, lo que aumenta considerablemente las opciones de uso que da.

Las relaciones entre los géneros clasificados y las figuras que aparecerán en la placa de leds vienen dadas en la Tabla 9 y esta relación es básicamente la letra inicial del nombre del género en que la canción sea clasificada. En el caso Reggae y Clásica, cuya inicial ya se usa en Rock y Country respectivamente, se usará el símbolo hippie de la paz para el reggae y el símbolo de una nota musical para la música clásica.

<i>Género</i>	<i>Figura</i>
Metal	M
Rock	R
Reggae	☰
Pop	P
Jazz	J
Clásica	♪
Blues	B
Disco	D
Country	C
Hip Hop	H

Tabla 9- Relación de géneros y figuras en la salida

La relación de los géneros clasificados y los colores RGB con las que las figuras de la Tabla 9 aparecerán vienen dados en la Tabla 10, cada color es más o menos diferente a los demás, para así, junto con las figuras anteriores poder distinguir cada género perfectamente.











<i>Género</i>	<i>Valores RGB</i>	<i>Apariencia</i>
Metal	(0,0,255)	
Rock	(255,0,0)	
Reggae	(246, 200, 29)	
Pop	(242,162,141)	
Jazz	(22,55,128)	
Clásica	(220,32,78)	
Blues	(93,71,139)	
Disco	(241, 33, 219)	
Country	(240,128,9)	
Hip Hop	(8,112,10)	

Tabla 10- Relación de géneros y colores en la salida

6.4.3 IMPLEMENTACIÓN

6.4.3.1 Estado del Arte

Para la implementación de esta parte se usará una librería llamada unicornhat que es una librería creada por Pimoroni, el fabricante de la placa Unicornhat específicamente para el uso de dicha placa.

La librería permite controlar los leds uno a uno o todos a la vez, permite, cambiar el brillo, encenderlos o apagarlos poniendo los colores que se deseen. También da la opción de obtener los valores rgb o el brillo de los leds además de otros valores como la forma general del conjunto de leds.

Para la implementación del programa general, lo primero que se hará, será capturar el audio que será analizado por el clasificador descrito en el apartado de Captura de datos. Una vez que se tenga el audio “.wav”, la salida del programa de captura de datos, se clasificará el audio grabado, con la clasificación que se genere del audio, se seleccionará la figura y el color correspondiente.

6.4.3.2 Pasos a seguir

1. Antes de elegir la figura y el color, se deberá inicializar la placa, eligiendo el modo de layout, que permite usar la mitad de la matriz o la matriz entera, en el programa se elegirá toda la placa, para poder utilizar todos los leds, se seleccionará el brillo de todos los leds a 0.5, ya que si elegimos más se pueden dañar los ojos
 2. También se obtendrá la forma de la placa con el método `get_shape()` que devuelve las dimensiones, en número de leds de la placa en forma del número de leds, esto será importante para encender los leds y saber que leds se deben encender.
 3. Cuando hayamos seleccionado el color y la forma que corresponda, se empezará a grabar indefinidamente los audios de 0.01 segundos y corriendo el clasificador de beats de la forma descrita en el apartado del Detector de beats.
 4. En este programa se incluye la detección de una nueva canción, en cuyo caso se volverá a inicializar el programa, realizando el mismo proceso desde el paso 1.
-

Capítulo 7. ANÁLISIS DE RESULTADOS

En este capítulo, se analizarán los resultados del sistema, para ello se dividirá el sistema en las dos partes fundamentales y más importantes, las cuales son el clasificador y detector de beats. Para ambas partes se valorarán los tres objetivos expuestos en el apartado de Objetivos, de forma que para ambas partes se analizará fundamentalmente la Latencia y la Precisión del clasificador y del detector de beats.

Para el objetivo de Capacidad de adaptación no se necesitará un especial análisis, puesto que se da por cumplido gracias al uso de la Raspberry Pi ya que debido a su pequeño tamaño, permite que se pueda transportar fácilmente, también debido a la naturaleza del sistema, el objetivo se puede cumplir de una forma más simple.

7.1 CLASIFICADOR

7.1.1 LATENCIA

El objetivo de mejorar la latencia es uno de los objetivos más importantes y uno de los que se encuentra más a nuestro alcance mejorar. El problema surge cuando a la hora de usar el clasificador para predecir el género de la canción que este sonando la latencia es demasiado grande si se usa el tiempo de grabación de 30 segundos, que es el tiempo de duración de las pistas usado en el dataset Gtzan Genre Collection.

El objetivo en este apartado es el de analizar todas las variantes de tiempo de grabación que podamos tener de forma que se pueda sacar la mejor combinación para que se reduzca el tiempo de latencia y la precisión del clasificador no se vea perjudicada, en otras palabras, se pretenderá optimizar el tiempo de grabación de forma que la precisión sea máxima.

El análisis de la precisión es un análisis subjetivo y depende del oyente, sin embargo, en este apartado se pretenderá objetivar dicho análisis lo máximo posible y sacar una herramienta única que pueda servir de análisis.

Para realizar el análisis se han seleccionado un total de trece canciones que se usarán a modo representativo, incluyendo al menos una canción para cada género. Las canciones que incluirán en el conjunto de test usado para el análisis serán:

“*9na Sinfonía de Beethoven*”: Una de las obras maestras de la música clásica del compositor Ludwig Van Beethoven, representará al género clásico dentro del conjunto de test. (Rivas, 2013)

“*Me & The Devil Blues*”: Una canción de blues compuesta e interpretada por el músico estadounidense Robert Johnson, representará al género Blues dentro del conjunto de test. (Schroeder)

“*You Can Do It*”: Una canción del rapero Ice Cub, bastante repetitiva y con un ritmo marcado, representará al género hip hop en el conjunto de test.

“*Homecoming*”: Una canción del rapero Kayne West, que incluye al vocalista Chris Martin de Coldplay, es una canción que representa al hip hop, pero que incluye rasgos del pop que dificultan su clasificación.

“*Skill To Pay The Bills*”: Una canción del grupo de rap conocido como Bestie Boys, se trata de una canción representativa del hip hop, pero que introduce muchos cambios de voces y ritmos, que dificultan su clasificación.

“*Dreams Are Not The Same*”: Una canción del grupo de heavy metal conocido como The Order, representará al género metal dentro del conjunto de test.

“*Take me home, Country Roads*”: Una canción escrita e interpretado por el cantante americano John Denver, representará al género Country dentro del conjunto de test. (Lawrence, 2014)

“*Friday*”: Una canción del artista J.J. Cale que puede ser clasificada como Country, pero a su vez fácilmente confundida con géneros como el Blues o el Jazz, dependiendo de la parte que se escuche.

“*Toxic*”: Una famosa canción de los años 90 interpretada por la cantante Britney Spiers, representará al género pop dentro del conjunto de test.

“*Eventide*”: Una canción de uno de los artistas más importantes de jazz como es James Carter, representará al género jazz dentro del conjunto de test.

“*What The Hell*”: Una canción englobada en el género punk, el cual entra dentro del rock, interpretada por Avril Lavigne, representará al género rock dentro del conjunto de test.

“*Dance Monkey*”: Una canción muy actual de los DJ Tones and I, representará al género Dance dentro del conjunto de test.

“*Buffalo Soldier*”: Una canción de las canciones más populares de una de las cabezas más representativas del reggae, como es Bob Marley, representará al género Reggae dentro del conjunto de test. (Fano, 2018)

Para cada canción se grabarán 5 pistas de audio que empezarán en la misma parte de la canción. Para hacer que todas las pistas se puedan comparar en igualdad de condiciones, se usará para la programación el programa descrito en el apartado de Captura de datos, pero se modificarán los tiempos de grabación. Los tiempos de grabación que se usarán serán 5 segundos, 10 segundos, 15 segundos, 20 segundos, 25 segundos y 30 segundos, sabiendo que los 30 segundos son los que usa el dataset y, por tanto, los que en teoría deben tener mejores resultados.

También se debe tener en cuenta cuál es el tiempo total que el programa toma en cada uno de los tiempos, desde que se empieza a grabar hasta que el clasificador predice el género en potencia de la canción, para eso, en el test se ha creado una variable que va a medir el tiempo total de cada pista, se calculará la media y la varianza de los valores del tiempo total para cada tiempo a modo de estimador, los valores obtenidos, se muestran en la Tabla 11.

<i>Tiempos</i>	<i>Media</i>	<i>Varianza</i>
t = 5 s	17,46	0,0127

t = 10 s	17,3466667	0,13453333
t = 15 s	22,35	0,0364
t = 20s	27,6033333	0,01223333
t = 25s	32,74	0,0388
t = 30s	37,4633333	0,09943333

Tabla 11- Tiempos totales

Como se puede observar, la grabación no es todo lo que puede tomar tiempo, puesto que, para el tiempo de 5 segundos, aunque el tiempo de grabación es menor que el de 10 segundos, el tiempo total es muy parecido, lo que quiere decir que aunque sea menor tiempo de grabación, el clasificador tendrá más dificultades, al tratarse de menos tiempo, de clasificar la pista y por tanto, la latencia también se puede ver resentida.

De la Tabla 11 también se puede extraer cuál es el tiempo que tomo el clasificador meramente para predecir el género dado el archivo de audio, se puede observar, que para el tiempo de 5 segundos el clasificador toma aproximadamente 12.5 segundos de tiempo en predecir el género y para el resto de los tiempos alrededor de 7.5 segundos. Se puede ver que tiempos aceptables serían los 22.35 segundos totales, los 27.6 segundos o incluso los 32.74 segundos si los resultados son muy buenos, pero que los 37.5 segundos son demasiado, ya que prácticamente se pasa toda la canción.

A la hora de evaluar las predicciones se recurrirá a un sistema de puntuaciones que hará que se tenga un método común para todos los tiempos y canciones. En el sistema de puntuaciones se evaluarán los distintos géneros que pueden ser confundidos por cualquier persona común, también castigará las predicciones que poco tienen que ver con el género real. Se otorgarán 3 puntos a los géneros aceptados por las predicciones, 0 para los géneros detectados parecidos al género real y -3 para los géneros que no tienen que ver con los géneros reales.

En la Tabla 12 se muestran las puntuaciones que se otorgarán a cada género.

Géneros	Blues	Clásica	Jazz	Rock	Metal	Pop	Reggae	Country	Hip hop	Dance
Blues	3	0	0	-3	-3	-3	-3	0	-3	-3
Clásica	0	3	0	-3	-3	-3	-3	-3	-3	-3
Jazz	0	0	3	-3	-3	-3	0	0	-3	-3
Rock	-3	-3	-3	3	0	0	-3	-3	-3	0
Metal	-3	-3	-3	0	3	-3	-3	-3	-3	-3
Pop	-3	-3	-3	0	-3	3	0	0	-3	0
Reggae	-3	-3	0	-3	-3	0	3	-3	-3	-3
Country	0	-3	0	-3	-3	0	-3	3	-3	-3
Hip hop	-3	-3	-3	-3	-3	-3	-3	-3	3	0
Dance	-3	-3	-3	0	-3	0	-3	-3	0	3

Tabla 12- Puntuaciones de géneros

Se intentará calcular una puntuación para cada tiempo y para canción, para ello se deberá crear una fórmula para hacer que las puntuaciones sean equivalentes a cada canción, se tendrá en cuenta tanto la primera puntuación como la segunda, así como las probabilidades que el clasificador ha otorgado a cada predicción, como se explica en el apartado de Implementación del Clasificador.

Para el cálculo de las puntuaciones, se usará la Ecuación 2.

$$\text{Puntuación} = \text{Probabilidad1} * \text{Puntuación1} + \text{Probabilidad2} * \text{Puntuación2}$$

Ecuación 2- Fórmula de puntuaciones

En la Ecuación 2, se entiende como Probabilidad1 a la probabilidad que se le otorga a la primera predicción, se entiende como Puntuación1 al valor obtenido al usar la Tabla 12 con los géneros reales y predichos por el clasificador en la primera predicción, se entiende como

Probabilidad² a la probabilidad que se le otorga a la segunda predicción, se entiende como Puntuación² al valor obtenido al usar la Tabla 12 con los géneros reales y predichos por el clasificador en la segunda predicción.

Según la Ecuación 2, se obtienen valores entre 3 y -3 como puntuaciones para todas las canciones y tiempos, de forma que las puntuaciones más cercanas a 3 serán las que sean mejores y las puntuaciones más cercanas a -3 serán predicciones peores.

Gracias al sistema de puntuaciones que se usará, no solo se tendrá en cuenta el acierto de la primera predicción, sino que también se tendrá en cuenta la segunda predicción, de forma que tanto si falla, como si acierta la primera predicción se tenga en cuenta la segunda predicción castigando o premiando la puntuación según lo cerca que haya estado esta. El motivo de esta decisión es debido a que como las pistas dependen del tiempo en que se graben, se pueda tener en cuenta que la pista haya podido cambiar por estar muy ajustado el resultado al elegir otro momento de grabación.

<i>Canción</i>	<i>t = 30 s</i>	<i>t = 25 s</i>	<i>t = 20 s</i>	<i>t = 15s</i>	<i>t = 10s</i>	<i>t = 5s</i>
9na Sinfonía de Beethoven	2,85	2,49	2,7	2,46	1,08	0
Me & The Devil Blues	0,96	1,2	1,05	1,02	1,68	0
Homecoming	0,03	-0,975	0,15	-0,18	-0,36	0,72
Eventilde	1,41	1,44	1,68	0,96	1,56	0,54
Skills To Pay The Bills	0,09	0,36	0,06	0,48	-1,02	1,98
Dreams Are Not The Same	2,97	2,82	2,31	2,82	2,04	1,02
Friday	0,96	0,63	0,84	0,63	0,84	0,72
Toxic	0,99	0,84	0,72	0,99	0,84	1,11

Take Me Home, Country Roads	1,86	1,95	1,5	1,65	1,2	0
What The Hell	2,01	1,62	1,86	1,83	0,63	-2,61
You Can Do It	1,23	1,38	1,62	1,83	1,68	2,01
Dance Monkey	0,33	0,24	0,68	-0,54	-0,96	-1,5
Buffalo Soldier	0,93	1,29	0,96	1,02	0,54	0,78
Total	16,62	15,285	16,13	14,97	9,75	4,77

Tabla 13- Puntuaciones obtenidas

Con los valores anteriores, se creará una gráfica de columnas apiladas, para apreciar de mejor forma la diferencia de cada tiempo y la influencia de cada canción. La gráfica se muestra en la Ilustración 16.

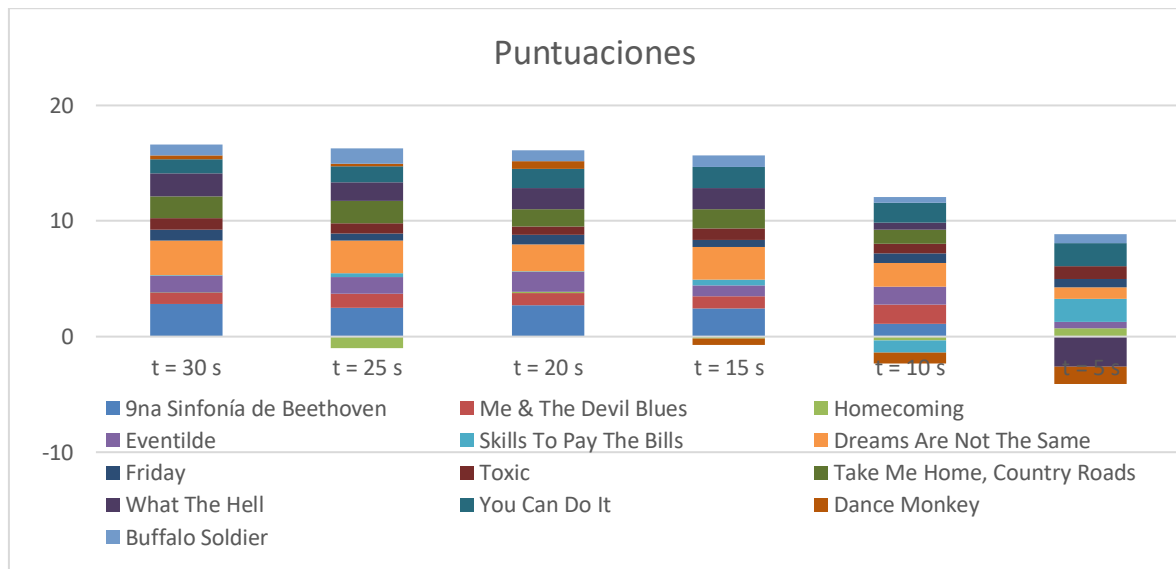


Ilustración 16- Gráficas de puntuaciones

Tanto en la Ilustración 16 como en la Tabla 13 se puede ver que el tiempo que mejor puntuación obtiene son las pistas de 30 segundos, pero las pistas de 25, 20 y 15 segundos también obtienen unas puntuaciones bastantes altas, destacando que las pistas de 15

segundos, no tienen ningún valor negativo, es decir, que no se han confundido en ningún momento.

7.1.2 PRECISIÓN

Para evaluar la precisión del modelo del clasificador se usarán herramientas tales como la exactitud, las pérdidas, la tasa de entrenamiento y la matriz de confusión, de esta última además se obtendrán varios parámetros.

Además, se evaluarán dos modelos de clasificador que se han creado, uno que se ha creado usando el dataset Gtzan Genre Collection íntegro y otro con el dataset truncado con las canciones añadidas como se explicó en el apartado Clasificador, de forma que se puedan comparar ambos modelos y decidir cuál será más preciso y mejor para usar.

El primer parámetro que se computará será la exactitud, dicho parámetro va de 0 a 1 y cuanto más cercano a 1 significará que el modelo será más exacto, ya que acertará más. Para observar mejor el progreso de la exactitud a lo largo de las épocas, se creará una gráfica con exponga la exactitud a lo largo de las iteraciones de entrenamiento para cada tasa comparada de ambos modelos.

En la Ilustración 17 y en la Ilustración 18, se muestran las gráficas de la tasa de entrenamiento que muestra la exactitud del modelo después de cada época de entrenamiento de los modelos creado por el dataset íntegro y truncado respectivamente.

Como se puede observar, en ambos modelos, al final de todas las épocas la exactitud aproximada será la misma y tendrá un valor de 0.7, pero para llegar a dicha exactitud, el modelo creado con el dataset truncado ha llegado a dicho valor de forma más irregular que el modelo creado con el dataset íntegro. Esto sugiere que el modelo creado con el dataset truncado le ha costado más entrenarse y ha sido más inestable, pero no es una diferencia muy destacable, puesto que la exactitud final es la misma.

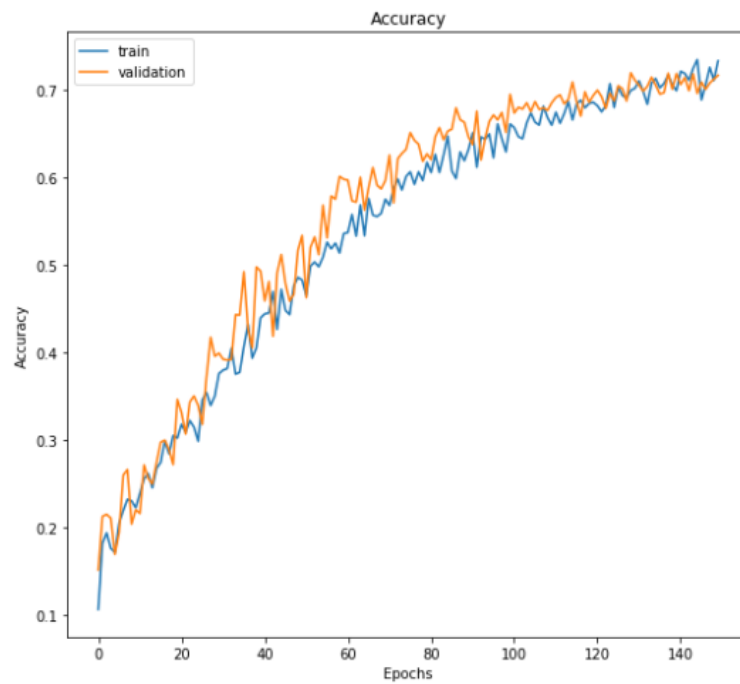


Ilustración 17- Tasa aprendizaje de exactitud creado con el modelo íntegro

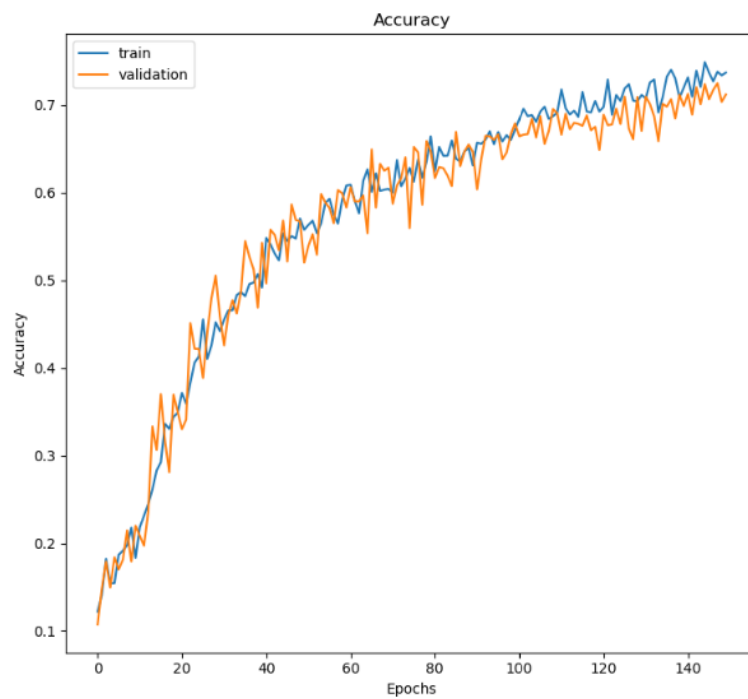


Ilustración 18- Tasa aprendizaje de exactitud creado con el modelo truncado

También se evaluarán las pérdidas de los dos modelos creados, en este caso y a contrario que en la exactitud, se pretende minimizar para que la precisión sea mejor, por lo tanto, cuanto menor sea el resultado de las pérdidas, menos errores cometerá el modelo y más preciso será.

Se creará, como se ha hecho anteriormente la tasa de entrenamiento para controlar cómo de rápido el modelo intenta ajustarse a los datos en cuestión y conocer la velocidad en que aprende.

En la Ilustración 19 y en la Ilustración 20 se muestran las tasas de entrenamiento de pérdida del modelo creado con el dataset íntegro y el modelo creado con el dataset truncado respectivamente.

Como ocurre con la exactitud, al acabar todas las épocas los modelos tienen las mismas pérdidas, de aproximadamente de 1, la diferencia nuevamente radica en cómo llegan ambos modelos a dicha pérdida, el modelo creado con el dataset truncado vuelve a mostrar dificultades e irregulares, sobre todo en que al inicio de la épocas empieza con un error mayor.

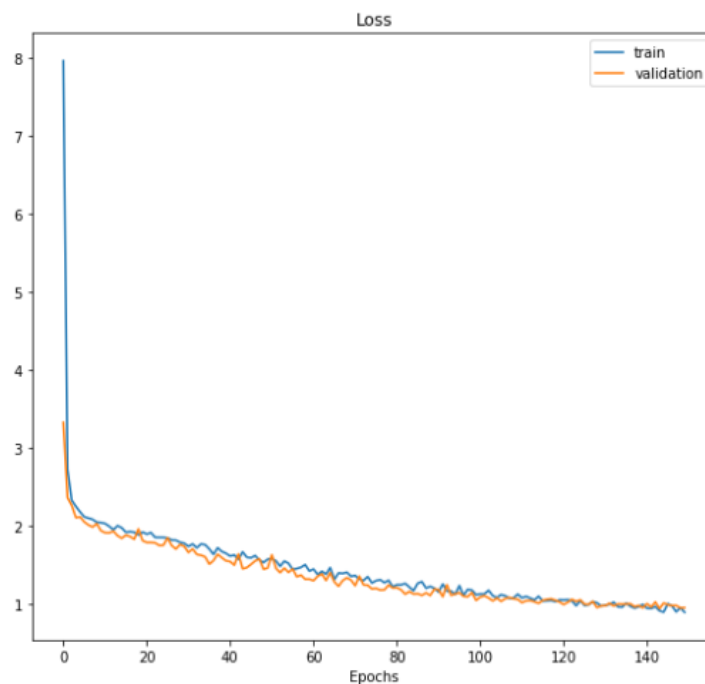


Ilustración 19- Tasa aprendizaje de pérdida creado con el modelo íntegro

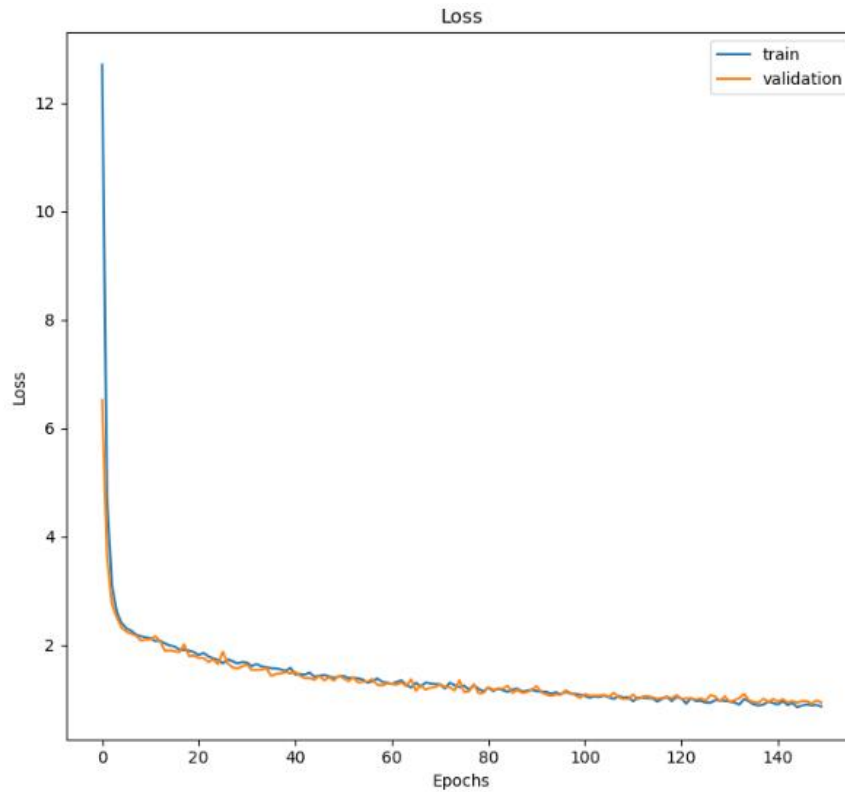


Ilustración 20- Tasa aprendizaje de pérdida creado con el modelo truncado

De las tasas de entrenamiento mostradas en la Ilustración 17, la Ilustración 18, la Ilustración 19 y la Ilustración 20, se puede extraer que al final de las épocas ambos modelos actúan de forma muy parecida tanto al evaluar las pérdidas y como al evaluar la exactitud, la mayor diferencia se encuentra en que el modelo truncado parece tener más problemas y ser más inestable sobre todo al principio de las épocas.

Por último, se usará la matriz de confusión para evaluar los dos modelos creados, de las dos matrices de confusión, se obtendrán los valores de la precisión, la exactitud, la sensibilidad, la especificidad y la F1 Score, este último será el valor resumen y el más importante de calcular.

En la Tabla 14 y en la Tabla 15, se muestran la matriz de confusión creada con el modelo que usó el dataset íntegro y la matriz de confusión creada con el modelo que usó el dataset truncado respectivamente.

MATRIZ DE CONFUSIÓN 1										
Metal	0,95	0	0	0	0	0	0	0	0	0,05
Disco	0,05	0,6	0	0,1	0	0,05	0	0	0,1	0,1
Classical	0	0	0,85	0	0	0,1	0	0	0	0,05
HipHop	0,05	0	0	0,8	0,05	0	0	0	0,1	0
Jazz	0	0,05	0	0,05	0,8	0	0,05	0,05	0	0,05
Country	0	0	0,1	0	0	0,65	0	0,05	0	0,2
Pop	0	0,05	0,1	0,05	0	0,05	0,75	0	0	0
Blues	0,05	0	0	0	0,15	0,05	0	0,7	0,05	0
Reggae	0	0	0	0,2	0	0	0,1	0	0,6	0,1
Rock	0,1	0	0,05	0	0	0,05	0	0	0,1	0,7
	Metal	Disco	Classical	Hiphop	Jazz	Country	Pop	Blues	Reggae	Rock

Tabla 14- Matriz de confusión del modelo creado con el dataset íntegro

MATRIZ DE CONFUSIÓN 2										
Metal	0,87	0,01	0	0,06	0	0	0	0	0,01	0,05
Disco	0,01	0,79	0	0,01	0	0,04	0,01	0,02	0,08	0,04
Classical	0	0,03	0,75	0,01	0,07	0,03	0	0,02	0,02	0,08
HipHop	0,01	0,05	0	0,8	0	0	0,05	0,01	0,08	0,01
Jazz	0,01	0,01	0,07	0	0,67	0,1	0,01	0,06	0,02	0,06

Country	0	0,01	0,01	0	0,04	0,66	0,02	0,04	0,04	0,16
Pop	0	0,05	0	0,05	0	0,01	0,85	0	0,03	0,01
Blues	0,03	0,02	0,01	0,01	0,03	0,11	0,01	0,62	0,07	0,09
Reggae	0	0,09	0	0,06	0,01	0,01	0,01	0,02	0,76	0,04
Rock	0,17	0,07	0,01	0,02	0	0,13	0,04	0,05	0,02	0,51
	Metal	Disco	Classical	Hiphop	Jazz	Country	Pop	Blues	Reggae	Rock

Tabla 15- Matriz de confusión del modelo creado con el dataset truncado

A continuación, con los resultados obtenidos en las tablas que muestran las matrices de confusión, se resumirán los cálculos de las métricas mencionadas anteriormente, también se detallará con la matriz de confusión que se han usado para ambos modelos, como se calcularán las métricas con los valores de las matrices, pues los valores de las matrices representan la precisión ya ahí puesto.

Se deberán introducir las abreviaciones que se van a usar en las fórmulas de las métricas necesarias.

VP será la cantidad de positivos que fueron clasificados correctamente como positivos por el modelo.

VN será la cantidad de negativos que fueron clasificados correctamente como negativos por el modelo.

FN será la cantidad de positivos que fueron clasificados incorrectamente como negativos, también conocidos como falsos negativos.

FP es la cantidad de negativos que fueron clasificados incorrectamente como positivos, también conocidos como falsos positivos.

El primer parámetro que se describirá será la exactitud o en inglés accuracy, se refiere a la cantidad de predicciones positivas que fueron correctas y se encuentra representado en la Ecuación 3.

$$Exactitud = \frac{VP + VN}{VP + FP + FN + VN}$$

Ecuación 3- Fórmula de la exactitud

Para el caso de las matrices que en este caso se usarán, la exactitud se calculará tal y como se ha usado y descrito en la página 53.

El siguiente parámetro que se describirá será la precisión o en inglés precision, se refiere al porcentaje de casos positivos detectados y se encuentra representado en la Ecuación 4.

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

Ecuación 4- Fórmula de la precisión

Para nuestro caso el parámetro de precisión viene dado directamente en la matriz de confusión, por lo que no se debe calcular nada más.

El próximo parámetro será la sensibilidad o en inglés recall, indica la capacidad del estimador para discriminar los casos positivos de los negativos y se encuentra representado en la Ecuación 5.

$$Sensibilidad = \frac{VP}{VP + FN}$$

Ecuación 5- Fórmula de la sensibilidad

Para el cálculo de la sensibilidad de cada género se sumarán todos los valores que aparecen en la matriz de los géneros predichos por el algoritmo, es decir, se sumarán las columnas para cada género y se le restará la unidad.

El siguiente parámetro será la especificidad o en inglés Especificity, indica la fracción de verdaderos negativos y que viene representado en la

$$\text{Especificidad} = \frac{VN}{VN + FP}$$

Ecuación 6- Fórmula de la especificidad

Para el cálculo de la especificidad de cada género se deberán todos los porcentajes de los errores para cada género, es decir, las filas y columnas de un género, menos donde coinciden a esta suma se le restará la unidad.

Por último, se calculará el F1 Score, que como ya se ha dicho será un valor resumen al considerarse un valor que engloba la precisión y la sensibilidad. La fórmula se encuentra en la Ecuación 7.

$$F1\ Score = \frac{2 * Sensibilidad * Precisión}{Sensibilidad + Precisión}$$

Ecuación 7- Fórmula de F1 Score

F1 Score será importante para detectar si el dataset estaba con desequilibrio y suele ocurrir que se obtenga un valor de precisión de la clase mayoritaria y una sensibilidad baja en la clase minoritaria.

Se ha creado una tabla para cada modelo que engloban los cálculos especificados anteriormente, las tablas serán la Tabla 16 y la Tabla 17.

Género	<i>Precisión</i>	<i>Sensibilidad</i>	<i>Especificidad</i>	<i>F1 Score</i>
Metal	0,95	0,75	0,5	0,83823529
Disco	0,6	0,9	0	0,72
Classical	0,85	0,75	0,6	0,796875

HipHop	0,8	0,85	0,4	0,82424242
Jazz	0,8	0,6	0,55	0,68571429
Country	0,65	0,7	0,35	0,67407407
Pop	0,75	0,85	0,6	0,796875
Blues	0,7	0,9	0,6	0,7875
Reggae	0,6	0,65	0,25	0,624
Rock	0,7	0,45	0,15	0,54782609

Tabla 16- Parámetros del modelo creado con el dataset íntegro

Género	<i>Precisión</i>	<i>Sensibilidad</i>	<i>Especificidad</i>	<i>F1 Score</i>
Metal	0,87	0,77	0,64	0,81695122
Disco	0,79	0,66	0,54	0,71917241
Classical	0,75	0,9	0,64	0,81818182
HipHop	0,8	0,78	0,51	0,78987342
Jazz	0,67	0,85	0,51	0,74934211
Country	0,66	0,57	0,25	0,61170732
Pop	0,85	0,85	0,7	0,85
Blues	0,62	0,78	0,4	0,69085714
Reggae	0,76	0,63	0,39	0,68892086
Rock	0,51	0,46	0	0,48371134

Tabla 17- Parámetros del modelo creado con el dataset truncado

Con los datos recaudados en las tablas anteriores, se ha creado un diagrama de columnas apiladas, que resumirá los resultados de las tablas y se podrá ver de forma más directa los

resultados de los modelos, el diagrama se muestra en la Ilustración 21

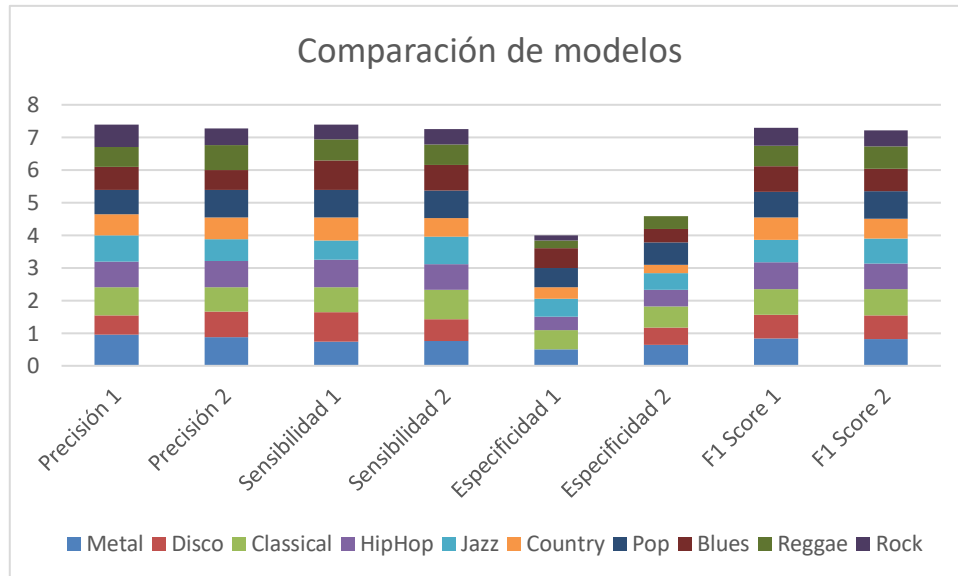


Ilustración 21- Diagrama de comparación de los modelos

Como se puede observar en la Ilustración 21, ambos modelos rinden de forma muy parecida, destaca el modelo creado con el dataset truncado teniendo una especificidad mejor que el modelo creado con el dataset íntegro, pero la diferencia no es ampliamente remarcable.

Para indagar más en profundidad en la comparativa de los modelos, se ha creado una tabla en el que se restan los resultados de uno con los de otros, se ha seleccionado el modelo creado con el dataset truncado para ser el modelo que reste, de forma que los géneros y los valores negativos indicarán que ese modelo será mejor y por el valor que se especifique.

Los datos de comparación se encuentran en la Tabla 18 y como se puede observar en la tabla, el modelo creado con el dataset truncado funciona mejor en líneas generales que el modelo creado con el dataset íntegro. Es importante destacar que hay géneros como el rock, el blues o el country, que funcionan mejor en el modelo creado con el dataset íntegro, mientras que otros géneros como el género disco, pop o reggae tienen un rendimiento superior en el modelo creado con el dataset truncado.

Género	<i>Precisión</i>	<i>Sensibilidad</i>	<i>Especificidad</i>	<i>F1 Score</i>	<i>Total</i>
Metal	0,08	-0,02	-0,14	0,02128407	-0,05871593
Disco	-0,19	0,24	-0,54	0,00082759	-0,48917241
Classical	0,1	-0,15	-0,04	-0,02130682	-0,11130682
HipHop	0	0,07	-0,11	0,03436901	-0,00563099
Jazz	0,13	-0,25	0,04	-0,06362782	-0,14362782
Country	-0,01	0,13	0,1	0,06236676	0,28236676
Pop	-0,1	0	-0,1	-0,053125	-0,253125
Blues	0,08	0,12	0,2	0,09664286	0,49664286
Reggae	-0,16	0,02	-0,14	-0,06492086	-0,34492086
Rock	0,19	-0,01	0,15	0,06411475	0,39411475
Total	0,12	0,15	-0,58	0,07662453	-0,23337547

Tabla 18- Comparación de modelos

Con todos los datos y gráficas recogidos, se deberá tomar la decisión de que modelo será mejor para el sistema y actuará de forma más precisa para que a la hora de recibir una canción nueva el modelo sea capaz de detectarla de forma más precisa.

7.1 DETECTOR DE BEATS

Para el detector de beats no se dividirá el análisis en latencia y precisión como se ha hecho en el Clasificador, puesto que la latencia se considerará una característica intrínseca de la precisión, puesto como el detector de beats será un programa que actúa a tiempo real, no se puede considerar preciso el detector de beats si la latencia es muy grande.

Además, es importante destacar que computar la latencia no hay forma objetiva de realizarse, pues la única forma puede ser observar que el detector de beats actúa de forma genérica bien, teniendo en cuenta que para un ser humano a simple vista no hay una forma científica de computar como se produce un beat.

Para evaluar la precisión del algoritmo, se usará la página web Tunebat, que es un recurso para ayudar a DJs, productores músicos y entusiastas de la música a encontrar información clave y los BPM, para más de 40 millones de canciones, y para descubrir pistas para mezclas armónicas y cualquier otro propósito imaginable. (Tunebat)

Se usará el programa de detector de beats del proyecto y se obtendrán los BPM que este obtiene, se compararán los BPM del programa que usará el sistema con los BPM que se obtienen de la página Tunebat. Las canciones que se usarán para comprobar el funcionamiento del detector de beats se usarán las canciones detalladas en la Tabla 19, se ha prendido tener un repertorio de canciones variado.

<i>Canción</i>	<i>Autor</i>	<i>Género</i>
Take me Home Country Road	John Denver	Country
Buffalo Soldier	Bob Marley	Reggae
Waterloo	ABBA	Pop
Gettin Over You	David Guetta	Dance
Forever Young	Alphaville	Pop
Summer of 69	Brian Adams	Rock
Smells Like Teen Spirit	Nirvana	Metal
The Blue Danube	Strauss	Classical
Game of Thrones Soundtrack	Ramin Djawadi	Classical
Red Red Wine	UB40	Reggae

Beautiful Girls	Danny Avila	Dance
We dem Boyz	Will Kalifa	Hip hop
Homecoming	Kayne West	Hip hop
Fine China	Future	Hip hop
Blinding Lights	The Weeknd	Dance

Tabla 19- Tabla de canciones para test de detector de beats

Cada canción de la Tabla 19, se ha reproducido entera cinco veces con el programa de detector de beats corriéndose a la vez. La razón de haberse reproducido cinco veces es para evitar posibles errores en alguna de las grabaciones, se ha calculado la media de cada canción para comparar la media de los BPM detectados por los BPMs que indica el programa de Tunebat. En la Tabla 20 se muestran las medias y los resultados de la media y varianza de los tests para el detector de beats.

<i>Canción</i>	<i>Medida 1</i>	<i>Medida 2</i>	<i>Medida 3</i>	<i>Medida 4</i>	<i>Medida 5</i>	<i>Media</i>	<i>Varianza</i>
Take me Home Country Road	165	167	155	160	163	162	22
Buffalo Soldier	127	140	111	135	150	132,6	215,3
Waterloo	144	138	139	150	141	142,4	23,3
Gettin Over You	131	138	132	135	126	132,4	20,3
Forever Young	136	132	144	139	138	137,8	19,2
Summer of 69	134	140	143	145	143	141	18,5
Smells Like Teen Spirit	118	128	121	115	111	118,6	41,3
The Blue Danube	74	88	110	82	96	90	190
Game of Thrones Soundtrack	172	167	166	172	171	169,6	8,3

Red Red Wine	128	133	94	109	119	116,6	243,3
Beautiful Girls	127	136	132	132	136	132,6	13,8
We dem Boyz	128	132	126	139	127	130,4	28,3
Homecoming	138	129	131	137	133	133,6	14,8
Fine China	165	156	172	170	158	164,2	50,2
Blinding Lights	169	167	168	163	165	166,4	5,8

Tabla 20- Cálculos de medias BPM

Como se puede observar las canciones Buffalo Soldier, y Red Red Wine son las que más varianza tienen, lo que indica que el detector de beats ha tenido más problemas y ha sido más inestable a la hora de detectar los beats en las mencionadas canciones.

Para comparar los resultados obtenidos con los resultados de Tunebat se ha creado la Tabla 21 donde se muestran las medias anteriores con los resultados de la página Tunebat, se ha agregado además una columna donde se muestra la diferencia en valor absoluto entre las medias calculadas y los resultados de la página web.

<i>Canción</i>	<i>Género</i>	<i>Media</i>	<i>Real</i>	<i>Diferencia</i>
Take me Home Country Road	Country	162	164	2
Buffalo Soldier	Reggae	132,6	124	8,6
Waterloo	Pop	142,4	148	5,6
Gettin Over You	Dance	132,4	130	2,4
Forever Young	Pop	137,8	137	0,8
Summer of 69	Rock	141	139	2
Smells Like Teen Spirit	Metal	118,6	117	1,6
The Blue Danube	Classical	90	90	0
Game of Thrones Soundtrack	Classical	169,6	170	0,4
Red Red Wine	Reggae	116,6	89	27,6
Beautiful Girls	Dance	132,6	130	2,6
We dem Boyz	Hip hop	130,4	130	0,4
Homecoming	Hip hop	133,6	130	3,6
Fine China	Hip hop	164,2	166	1,8
Blinding Lights	Dance	166,4	171	4,6

Tabla 21- Comparación de BPMs

En la tabla se puede observar que los resultados son bastante parecidos a los de la página Tunebat, excepto para las canciones Buffalo Soldier y Red Red Wine, lo que quiere decir que el detector de beats funciona bastante bien para la mayoría de las canciones. Cabe destacar que las canciones Buffalo Soldier y Red Red Wine pertenecen ambas al género reggae, por lo que se puede decir que el detector de beats encuentra más problemas para canciones de reggae, pero dichos problemas no son excesivamente destacables, ya que al

funcionar el programa, los beats detectados, no son muy diferentes a los que una persona normal puede detectar o no se hace muy extraño.

Capítulo 8. CONCLUSIONES Y TRABAJOS FUTUROS

En el proyecto, se ha completado el objetivo que es el de crear un sistema que mezcle luces con música creando una experiencia sensorial que mezcle la vista con el oído. Para ello se ha dividido el proyecto en varias partes, de las que destacan fundamentalmente un clasificador que captura el sonido de la música que este sonando y que clasificará las canciones captadas según el género que corresponda. También destaca el detector de beats que indicará cuando se produzca un beat en la canción a tiempo real, para hacer que las luces luzcan al ritmo de la música.

Para el clasificador se ha evaluado el tiempo de grabación y de captura del audio de forma que el tiempo se minimice y que la precisión se maximice. Para ello en el apartado de Latencia dentro del Clasificador del capítulo de Análisis de resultados se ha evaluado para diferentes tiempos, entre los que se incluyen 5, 10, 15, 20, 25 y 30 segundos, la precisión del clasificador. De los resultados de dicho apartado se puede decir que los mejores tiempos que se pueden seleccionar para maximizar la precisión pueden fluctuar entre 15, 20 y 25 segundos.

De los tiempos anteriores, finalmente se seleccionará el menor de ellos, es decir, el de 15 segundos, que es el que menor latencia da. Seleccionando el programa que grabe 15 segundos, el clasificador desde que comience a grabar, hasta que se obtiene la predicción tardará aproximadamente de media 22.3 segundos. A partir de ese tiempo los leds de la matriz comenzarán a lucir al ritmo de la música con el color y la figura representativa que aparecen en la Tabla 9 y la Tabla 10. Cuando el programa detecte que hay un cambio de canción, el clasificador se volverá a llamar y después de aproximadamente 22.3 segundos los leds volverán a lucir de nuevo.

Además, para el clasificador se han creado dos modelos, un modelo usando el dataset Gtzan Genre Collection íntegro y otro modelo usando el dataset Gtzan Genre Collection truncado,

es decir, se le han añadido varias canciones cuyo número de canciones se muestran en la Tabla 7.

Se ha evaluado cuál de los dos modelos actúa de forma más precisa, el desarrollo de la evaluación se encuentra en el apartado de Precisión del Clasificador dentro del capítulo de Análisis de resultados, en el que se ha utilizado la exactitud, la pérdida y la matriz de confusión junto con los parámetros que se obtienen de ella para la evaluación.

Los resultados obtenidos, son muy parecidos para ambos modelos, pero en líneas generales, el modelo creado con el dataset truncado actúa un poco mejor y sobre todo actúa mejor para detectar géneros que pueden ser más usados por el sistema, como por ejemplo los géneros disco y pop, para favorecer a estos géneros es a la razón por lo que el dataset se ha truncado.

A la hora de mejorar el sistema, en concreto, para mejorar el clasificador, la mejor opción recaería en aumentar el dataset y aumentar la variación en la grabación de las canciones, por aumentar el dataset se entiende a un aumento importante tal que se multiplique por 10 el número de muestras por género. Con el aumento, también se debe aumentar el número de hiperparámetros para aumentar así la precisión del modelo.

Esta mejora no se ha podido hacer debido a una falta de tiempo y recursos, se calcula que se tardará 1 mes extra en recolectar todos los datos necesarios.

La evaluación del detector de beats tal y como se puede apreciar en la Tabla 21, sugiere que el detector de beats funciona correctamente, para los ejemplos usados, el mayor problema se encuentra al detectar las canciones del género reggae, ya que este se vuelve más inestable, pero este problema no es muy destacable porque el resultado con las canciones de este género no es muy negativo.

Como una posible mejora para el futuro del proyecto del detector de beats es hacer que si el clasificador detecta una canción del género reggae, el detector de beats actúe de forma distinta para este género, de forma que como las canciones de reggae suelen ser más lentas de lo normal y dado que el detector de beats, detecta en ellas más beats de lo que son, se

ajuste más la detección para hacer que se detecten menos beats o que los beats que el detector marque como beats estén más asegurados.

A modo de conclusión, cabe destacar que el proyecto no se limita únicamente a trabajar con música y hacer una respuesta con un juego de luces, sino que, al tener la capacidad de analizar la música, ayuda en otras aplicaciones que sirven para acercar la música a todos y brindar las experiencias que todas las personas sienten al escuchar música, por ejemplo, a personas sordas.

Con el sistema, a través de las luces, personas con problemas auditivos, se les abrirá un nuevo mundo haciendo la música visible. La luz puede servir para interpretar otras realidades, realidades que no siempre podemos captar con nuestros sentidos y que permite crear lazos entre las personas.

Por último, se debe destacar que el sistema, no solo este limitado a la placa Unicorn Hat de Pimoroni, sino que permite un gran número de variantes en forma de tiras de leds. La tecnología led es una tecnología que demuestra de forma muy palpable su versatilidad y potencialidad de ser vehículo para la comunicación. El sistema permite que se pueda instalar en cualquier tipo de espacios, desde escenarios, hasta sobre micrófonos, bocinas y algunos instrumentos.

Capítulo 9. BIBLIOGRAFÍA

- ABC Tecnología. (21 de 07 de 2013). *ABC*. Obtenido de ¿Qué es Raspberry PI y para qué sirve?: <https://www.abc.es/tecnologia/informatica-hardware/20130716/abc-raspberry-como-201307151936.html>
- Amazon. (s.f.). *Amazon*. Obtenido de GreenClick: https://www.amazon.es/120-ledes-Manguera-betriben-resistente-iluminaci%C3%B3n/dp/B074PQGTZR/ref=sr_1_3_sspa?dchild=1&keywords=Al+r%C3%ADtimo+de+la+m%C3%BAscica&qid=1590003657&sr=8-3-spons&psc=1&spLa=ZW5jcmlwdGVkUXVhbGlmaWVyPUEyMENLTIBNQ0M3WlVJmVuY3J5cHRIZElkPUEwMzYw
- Arce, J. I. (26 de 7 de 2019). *Health Big Data*. Obtenido de La matriz de confusión y sus métricas: <https://www.juanbarrios.com/matriz-de-confusion-y-sus-metricas/>
- Balagueró, T. (13 de 11 de 2018). *Deusto Formación*. Obtenido de ¿Qué son los datasets y los dataframes en el Big Data?: <https://www.deustoformacion.com/blog/programacion-diseno-web/que-son-datasets-dataframes-big-data>
- Brian McFee. (2015). *librosa: Audio and Music Signal Analysis in Python*. Obtenido de http://conference.scipy.org/proceedings/scipy2015/pdfs/brian_mcfree.pdf
- Brian McFee. (7 de 6 de 2015). *Librosa: v0.4.0*. Obtenido de <https://zenodo.org/record/18369#.Xv-1UygzZPY>
- Burgal, J. U. (11 de 07 de 2018). *En mi local funciona*. Obtenido de Deep Learning básico con Keras (Parte 2): Convolutional Nets: <https://enmilocalfunciona.io/deep-learning-basico-con-keras-parte-2-convolutional-nets/>
-

- Cornieles, P. (6 de 2 de 2019). *IA LATAM*. Obtenido de Entendiendo las redes neuronales: De la neurona a RNN, CNN y Deep Learning: <https://ia-latam.com/2019/02/06/entendiendo-las-redes-neuronales-de-la-neurona-a-rnn-cnn-y-deep-learning/>
- Equipo Paradigma Digital. (2020). *Machine Learning: 50 Conceptos clave para entenderlo*. Creative Commons.
- Fano, V. M. (21 de 5 de 2018). *La Soga*. Obtenido de Buffalo Soldier: la canción de los afroamericanos que lucharon por el hombre blanco: <https://lasoga.org/buffalo-soldier-la-cancion-de-los-afroamericanos-que-lucharon-por-el-hombre-blanco/>
- File Extension. (s.f.). *File Extension*. Obtenido de ¿Cuál es el archivo H5?: <https://www.file-extension.org/es/extensions/h5>
- García, O. (11 de 6 de 2014). *Proyectum*. Obtenido de Fases del proyecto: <https://www.proyectum.com/sistema/blog/fases-del-proyecto/#:~:text=La%20estructuraci%C3%B3n%20en%20fases%20permite,su%20direcci%C3%B3n%20planificaci%C3%B3n%20y%20control.&text=La%20terminaci%C3%B3n%20de%20esta%20fase,cambiar%20o%20terminar%20el%20proye>
- Gartzman, D. (19 de 8 de 2019). *Towards Data Science*. Obtenido de Getting to Know the Mel Spectrogram: <https://towardsdatascience.com/getting-to-know-the-mel-spectrogram-31bca3e2d9d0>
- González, J. I. (8 de 10 de 2019). *uc3m*. Obtenido de Procesamiento de Audio con Técnicas de Inteligencia Artificial: https://e-archivo.uc3m.es/bitstream/handle/10016/29348/TFG_Jesus_Iriz_Gonzalez.pdf?sequence=1&isAllowed=y
- Howard, P. (s.f.). *Pimoroni*. Obtenido de Getting Started with Unicorn HAT: <https://learn.pimoroni.com/tutorial/unicorn-hat/getting-started-with-unicorn->
-

- Programo ERDO SUM. (s.f.). *programoergosum*. Obtenido de <https://www.programoergosum.com/cursos-online/raspberry-pi/232-curso-de-introduccion-a-raspberry-pi/instalar-raspbian>
- Refraction Productions. (s.f.). *Refraction Productions*. Obtenido de Qué es la frecuencia de muestreo (Sampling Rate) y profundidad de bits (Resolución): <https://refractionproductions.com/que-es-frecuencia-de-muestreo-profundidad-resolucion-bits-audio/>
- Rivas, M. V. (2013). *La Novena sinfonía de Beethoven: historia, ideas y estética*. Academia.edu. Obtenido de La Novena sinfonía de Ludwig van Beethoven.
- Rodríguez, V. (9 de 11 de 2018). *Vicente Rodríguez*. Obtenido de Dropout y Batch Normalization: <https://vincentblog.xyz/posts/dropout-y-batch-normalization>
- Rubén Gómez Fuentes, Daniel Lago Agudo. (6 de 2016). *Universidad Complutense de Madrid*. Obtenido de Inteligencia ambiental en el Internet de las Cosas: <https://eprints.ucm.es/38509/1/Memoria%20TFG.pdf>
- Sastrón, J. (7 de 5 de 2017). *Producciones El Sótano*. Obtenido de Entendiendo conceptos básicos de los analizadores FFT: <https://www.produccioneselotano.com/entendiendo-conceptos-basicos-de-los-analizadores-fft/>
- Schroeder, P. R. (s.f.). *Robert Johnson, Mythmaking and contemporary American Culture*.
- Sharma, S. (6 de 9 de 2017). *Towards Data Science*. Obtenido de Activation Functions in Neural Networks: <https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6>
- Tunebat. (s.f.). *Key & BPM Database and Music Finder*. Obtenido de <https://tunebat.com/>
-

Universidad de Alcalá. (s.f.). *Master Data Scientist*. Obtenido de Scikit-learn, herramienta básica para el Data Science en python: <https://www.master-data-scientist.com/scikit-learn-data-science/>

ANEXO I: ALINEACIÓN DEL PROYECTO CON LOS ODS

Dimensión SDG: Economía.

Papel: Secundario.

Identidad SDG: SDG 9: Crear una industria inclusiva y sostenible, junto con la innovación y la infraestructura, que den rienda suelta a las fuerzas económicas dinámicas y competitivas que generan el empleo y los ingresos. Que se desempeñe un papel clave a la hora de introducir y promover nuevas tecnologías, a su vez, facilitar el comercio internacional y permitir el uso eficiente de recursos.

Objetivo: Aumentar el acceso de las pequeñas industrias y otras empresas, particularmente en los países en desarrollo, a los servicios financieros, incluidos créditos asequibles, y su integración en las cadenas de valor y los mercados.

Dimensión SDG: Economía.

Papel: Secundario.

Identidad SDG: SDP 12: Promover que el buen consumo y una producción mundial que dependen demasiado del uso del medio ambiente natural y de los recursos de una manera continua, disminuya. Promover un consumo y producción sostenibles consistente en hacer más y mejor con menos. Se trata de desvincular el crecimiento.

Objetivo: Ayudar a los países en desarrollo a fortalecer su capacidad científica y tecnológica para avanzar hacia modalidades de consumo y producción más sostenibles

Dimensión SDG: Economía.

Papel: Primario.

Identidad SDG: SDP 8: El crecimiento económico inclusivo y sostenido puede impulsar el progreso, crear empleos decentes para todos y arriesgar los estándares de vida.

Objetivo: Lograr niveles más elevados de productividad económica mediante la diversificación, la modernización tecnológica y la innovación, entre otras cosas centrándose en los sectores con gran valor añadido y un uso intensivo de la mano de obra.

ANEXO II

Para acceder a todo el código del proyecto, así como los ejemplos grabados para los tests y las tablas que se han hecho en Excel para el apartado de Análisis de resultados se puede acceder mediante el siguiente repositorio de github:

<https://github.com/alfreddo98/Music-Detecting-System>

Dentro del repositorio, se encontrarán dos carpetas:

Data: contiene los datos que se usarán, es decir le modelo creado anteriormente y puesto en forma de .h5 y ejemplos de canciones en formato .WAV.

Src: Se encuentra todo el código creado y usado en el proyecto, se encuentra dividido en tres carpetas, BeatDetector, Classifier y Tests, además en el directorio se encuentra el programa final del proyecto, el cual cuando se corra, será el que finalmente funcione,

BeatDetector: Dentro de BeatDetector, se encuentran dos programas el programa llamado recorder será importado por beat y en el se encuentra la clase recorder con todos los métodos necesarios para grabar. En el programa beat se encuentra todo el código que importando recorder grabará y detectará los beats.

Classifier: En esta carpeta se encuentran los programas relativos al clasificador, en modelCreator se encuentra el programa que creará el modelo que se guardará en formato .h5, en el programa Recognition esta el código que se usará para reconocer las canciones importando el modelo creado por modelCreator y cogiendo el archivo de audio .WAV que se especifique. Por último, el programa leeraudio se usará para capturar el audio y guardarlo en formato .WAV.

Tests: Contiene todas las canciones que se han grabado con el programa leeraudio y se usaron como test, en especial para la parte de latencia del clasificador. También se encuentra el Excel con todos los datos recogidos y todas las tablas creadas y usadas para este documento, en especial y sobre todo las tablas del apartado de Análisis de resultados.
