Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ingeniería en Ciencias y Sistemas

ANÁLISIS COMPARATIVO DE LAS TÉCNICAS DE DEEP LEARNING PERCEPTRÓN MULTICAPA Y REDES NEURONALES CONVOLUCIONALES APLICADAS A LA CLASIFICACIÓN MUSICAL

Luis Leonel Aguilar Sánchez
Asesorado por el Ing. Javier Estuardo Navarro Delgado

Guatemala, marzo de 2023

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

ANÁLISIS COMPARATIVO DE LAS TÉCNICAS DE DEEP LEARNING PERCEPTRÓN MULTICAPA Y REDES NEURONALES CONVOLUCIONALES APLICADAS A LA CLASIFICACIÓN MUSICAL

TRABAJO DE GRADUACIÓN

PRESENTADO A LA JUNTA DIRECTIVA DE LA
FACULTAD DE INGENIERÍA
POR

LUIS LEONEL AGUILAR SÁNCHEZ
ASESORADO POR EL ING. JAVIER ESTUARDO NAVARRO DELGADO

AL CONFERIRSELE EL TÍTULO DE

INGENIERO EN CIENCIAS Y SISTEMAS

GUATEMALA, MARZO DE 2023

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

NÓMINA DE JUNTA DIRECTIVA

DECANA	Inga. Aurelia Anabela Cordova Estrada
VOCAL I	Ing. José Francisco Gómez Rivera
VOCAL II	Ing. Mario Renato Escobedo Martínez
VOCAL III	Ing. José Milton de León Bran
VOCAL IV	Br. Kevin Vladimir Armando Cruz Lorente
VOCAL V	Br. Fernando José Paz González
SECRETARIO	Ing. Hugo Humberto Rivera Pérez

TRIBUNAL QUE PRACTICÓ EL EXAMEN GENERAL PRIVADO

DECANA	Inga. Aurelia Anabela Cordova Estrada
EXAMINADOR	Ing. Herman Igor Véliz Linares
EXAMINADOR	Ing. Neftalí de Jesús Calderón Méndez
EXAMINADOR	Ing. Álvaro Giovanni Longo Morales
SECRETARIO	Ing. Hugo Humberto Rivera Pérez

HONORABLE TRIBUNAL EXAMINADOR

En cumplimiento con los preceptos que establece la ley de la Universidad de San Carlos de Guatemala, presento a su consideración mi trabajo de graduación titulado:

ANÁLISIS COMPARATIVO DE LAS TÉCNICAS DE DEEP LEARNING PERCEPTRÓN MULTICAPA Y REDES NEURONALES CONVOLUCIONALES APLICADAS A LA CLASIFICACIÓN MUSICAL

Tema que me fuera asignado por la Dirección de la Escuela de Ingeniería en Ciencias y Sistemas, con fecha 9 de noviembre de 2021.

Luis Leonel Aguilar Sánchez

Guatemala, 15 de enero de 2023

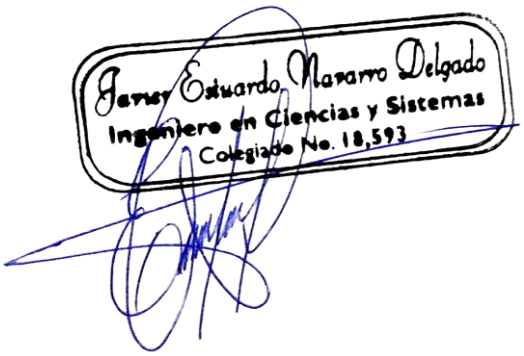
Ingeniero
Carlos Alfredo Azurdia
Coordinador de Privados y Trabajos de Tesis
Escuela de Ingeniería en Ciencias y Sistemas
Facultad de Ingeniería - USAC

Respetable Ingeniero Azurdia:

Por este medio hago de su conocimiento que en mi rol de asesor del trabajo de investigación realizado por el estudiante **LUIS LEONEL AGUILAR SÁNCHEZ** con carné **201603029 y CUI 3001 38180 0101** titulado **"ANÁLISIS COMPARATIVO DE LAS TÉCNICAS DE DEEP LEARNING PERCEPTRÓN MULTICAPA Y REDES NEURONALES CONVOLUCIONALES APLICADAS A LA CLASIFICACIÓN MUSICAL",** luego de corroborar que el mismo se encuentra finalizado, lo he revisado y doy fe de que el mismo cumple con los objetivos propuestos en el respectivo protocolo, por consiguiente, procedo a la aprobación correspondiente.

Al agradecer su atención a la presente, aprovecho la oportunidad para suscribirme,

Atentamente,



Ing. Javier Estuardo Navarro Delgado
Colegiado No. 18593

Universidad San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ingeniería en Ciencias y Sistemas

Guatemala 17 de enero de 2023

Ingeniero
Carlos Gustavo Alonzo
Director de la Escuela de Ingeniería
En Ciencias y Sistemas

Respetable Ingeniero Alonzo:

Por este medio hago de su conocimiento que he revisado el trabajo de graduación del estudiante **LUIS LEONEL AGUILAR SÁNCHEZ** con carné **201603029** y CUI **3001 38180 0101** titulado **"ANÁLISIS COMPARATIVO DE LAS TÉCNICAS DE DEEP LEARNING PERCEPTRÓN MULTICAPA Y REDES NEURONALES CONVOLUCIONALES APLICADAS A LA CLASIFICACIÓN MUSICAL",** y a mi criterio el mismo cumple con los objetivos propuestos para su desarrollo, según el protocolo aprobado.

Al agradecer su atención a la presente, aprovecho la oportunidad para suscribirme,

Atentamente,

Ing. Carlos Alfredo Azurdia
Coordinador de Privados
y Revisión de Trabajos de Graduación

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

LNG.DIRECTOR.062.EICCSS.2023

El Director de la Escuela de Ingeniería en Ciencias y Sistemas de la Facultad de Ingeniería de la Universidad de San Carlos de Guatemala, luego de conocer el dictamen del Asesor, el visto bueno del Coordinador de área y la aprobación del área de lingüística del trabajo de graduación titulado: **ANÁLISIS COMPARATIVO DE LAS TÉCNICAS DE DEEP LEARNING PERCEPTRÓN MULTICAPA Y REDES NEURONALES CONVOLUCIONALES APLICADAS A LA CLASIFICACIÓN MUSICAL**, presentado por: **Luis Leonel Aguilar Sánchez** , procedo con el Aval del mismo, ya que cumple con los requisitos normados por la Facultad de Ingeniería.

"ID Y ENSEÑAD A TODOS"

Ing. Carlos Gustavo Alonzo
Director
Escuela de Ingeniería en Ciencias y Sistemas

MSc. Ing. Carlos Gustavo Alonzo
Director
Escuela de Ingeniería en Ciencias y Sistemas

Guatemala, marzo de 2023

Decanato
Facultad de Ingeniería
24189101- 24189102
secretariadecanato@ingenieria.usac.edu.gt

LNG.DECANATO.OI.289.2023

La Decana de la Facultad de Ingeniería de la Universidad de San Carlos de Guatemala, luego de conocer la aprobación por parte del Director de la Escuela de Ingeniería en Ciencias y Sistemas, al Trabajo de Graduación titulado: **ANALISIS COMPARATIVO DE LAS TÉCNICAS DE DEEP LEARNING PERCEPTRÓN MULTICAPA Y REDES NEURONALES CONVOLUCIONALES APLICADAS A LA CLASIFICACIÓN MUSICAL,** presentado por: **Luis Leonel Aguilar Sánchez,** después de haber culminado las revisiones previas bajo la responsabilidad de las instancias correspondientes, autoriza la impresión del mismo.

IMPRÍMASE:

Inga. Aurelia Anabela Cordova Estrada
Decana

Guatemala, marzo de 2023

AACE/gaoc

ACTO QUE DEDICO A:

Mis padres	Luis Aguilar, y Alba Sánchez, por darme todo lo que necesité a lo largo de mi vida, su amor, esfuerzo, sacrificios y dedicación para forjarme como persona y como profesional.
Mis hermanos	Fhernando y Alcrissa Aguilar, por siempre motivarme a ser su ejemplo por seguir y enseñarme a ser niño de nuevo.
Mis amigos	Por su apoyo incondicional a lo largo de la carrera, su cariño, todos los momentos memorables y aventuras que vivimos.
Mi novia	Sindy González, por sus palabras de ánimo y aliento, su paciencia y calidez, por siempre sacarme una sonrisa, por cada viaje, salida y aventura, y lo más importante: por su amor.

AGRADECIMIENTOS A:

Universidad de San Carlos de Guatemala	Por ser mi casa de estudios y formarme como profesional.
Facultad de Ingeniería	Por retarme constantemente a superarme a mí mismo, y brindarme la oportunidad de ser uno más de sus alumnos.
Mi asesor	Ing. Javier Navarro, por haberme guiado, compartiendo sus valiosos consejos y retroalimentación en la elaboración de este trabajo.

ÍNDICE GENERAL

ÍNDICE DE ILUSTRACIONES

FIGURAS

TABLAS

LISTA DE SÍMBOLOS

Símbolo	Significado
GB	Giga Byte
Hz	Hercios
%	Porcentaje
∑	Sumatoria

GLOSARIO

AU Es un formato de archivo utilizado para almacenar datos de audio en una computadora. Es un formato sin compresión y sin pérdida de calidad.

Convolución La convolución es una técnica matemática utilizada en procesamiento de señales y visión artificial para combinar dos funciones de manera que el resultado sea otra función.

FFT La Transformada Rápida de Fourier (FFT, por sus siglas en inglés), es un algoritmo eficiente para calcular la Transformada de Fourier Discreta de una señal discreta.

Género musical El género musical de una canción es una categoría utilizada para clasificar y etiquetar música según sus características estilísticas y culturales.

Hertz Los Hertz o Hercios son la unidad de medida de la frecuencia en el Sistema Internacional de Unidades. En el contexto de la música, la frecuencia se refiere a la cantidad de veces que una onda sonora oscila por segundo.

Mel	Es una unidad de medida utilizada para describir la percepción subjetiva de la altura de un sonido utilizada por el oído humano.
MFCCs	Los coeficientes cepstrales de la frecuencia de Mel (MFFCs, por sus siglas en inglés), son una representación matemática de las características de una señal de audio. Se obtienen a partir de múltiples pasos que incluyen el análisis de Fourier, la aplicación de una ventana temporal y la transformación a la frecuencia de Mel. El resultado son un conjunto de coeficientes numéricos que representan las características de la señal de audio.
MP3	Es un formato de compresión de audio desarrollado por la *Fraunhofer Society*, una institución de investigación alemana. El formato MP3 permite comprimir archivos de audio sin una pérdida significativa de calidad.
Octava	Una octava es la relación musical entre dos notas en la que la frecuencia de una nota es exactamente el doble o la mitad de la frecuencia de la otra nota.
Python	Lenguaje de programación multiparadigma, que soporta programación orientada a objetos, imperativa y funcional, es un lenguaje interpretado, dinámico y multiplataforma.

STFT	La Transformada de Fourier de Tiempo-Frecuencia (STFT, por sus siglas en inglés), es una técnica matemática utilizada para analizar señales de audio y otros tipos de señales en tiempo y frecuencia.
WAV	WAV es un formato de archivo de audio desarrollado por Microsoft y IBM que se utiliza para almacenar audio digital sin compresión. El formato WAV es de estándar abierto y es compatible con una variedad de sistemas operativos y dispositivos de reproducción de sonido.

RESUMEN

Los géneros musicales influyen en la manera en la que la frecuencia de una canción varía a lo largo de su duración: géneros musicales como el *rock* tienden a tener una mayor variedad de frecuencias y patrones complejos de las mismas en comparación con otros géneros musicales como la música clásica. Estas diferencias en las características inciden en el surgimiento de patrones específicos para cada género musical reflejándolos en el espectrograma de frecuencias de Mel y en los coeficientes cepstrales en la frecuencia de Mel.

La clasificación de audio por género musical utilizando Deep Learning implica el uso de redes neuronales artificiales para analizar y clasificar a las canciones a partir de dichos patrones. Utilizando aprendizaje supervisado es posible entrenar a una red neuronal para reconocerlos y construir un sistema clasificador inteligente.

Las redes neuronales convolucionales (CNN, por sus siglas en inglés), y las redes neuronales perceptrón multicapa (MLP, por sus siglas en inglés), son dos tipos de redes neuronales que se utilizan comúnmente en tareas de clasificación.

Las CNN son especialmente adecuadas para el procesamiento de datos con estructuras especiales, como imágenes o señales de audio. Estas son capaces de aprender patrones complejos a través de la extracción de características mediante el uso de capas de conexión, parecido a los filtros de una cámara.

Las MLP son redes neuronales tradicionales que son capaces de aprender patrones complejos mediante el uso de un gran número de neuronas y de su arquitectura consistente de varias capas interconectadas.

Implementando estas dos técnicas de Deep Learning en un sistema clasificador de audio por género musical es posible comparar sus capacidades y limitantes; y permita decidir cuál de las dos es más apta para esa tarea de clasificación.

OBJETIVOS

General

Analizar y comparar la utilización de una red neuronal convolucional y una red neuronal perceptrón multicapa aplicadas a la clasificación de audio por género musical.

Específicos

1. Construir un sistema clasificador que permita a un usuario cargar un archivo digital de audio, procesar el archivo utilizando cualquiera de las dos redes neuronales antes descrita, y mostrar los resultados de la clasificación por género musical del mismo.

2. Diseñar y entrenar una red neuronal convolucional y una red neuronal perceptrón multicapa que clasifique audio a partir de la información obtenida de su espectrograma de Mel y los coeficientes cepstrales en la frecuencia de Mel.

3. Comparar la precisión y velocidad de predicción de los dos métodos de clasificación antes descritos, en general y por género musical.

4. Determinar la técnica de Deep Learning más apropiada para clasificar géneros musicales utilizando el espectrograma de Mel y los coeficientes cepstrales en la frecuencia de Mel.

INTRODUCCIÓN

Las técnicas de aprendizaje de máquina, en específico el Deep Learning, han sido demostradas efectivas para una amplia gama de tareas de clasificación. Empleándolas correctamente, estas técnicas pueden automatizar procesos de categorización y extracción de características en una manera confiable y rápida.

Una posible aplicación de las técnicas de reconocimiento del Deep Learning es la clasificación de audio por su género musical, el cual puede ser determinado a partir de ciertas características del sonido.

Para ello, se entrenará a una red neuronal convolucional y una red neuronal perceptrón multicapa empleando el procesamiento y análisis de la información contenida en archivos digitales de música. En específico se tomará como base la representación por espectrogramas de Mel y los coeficientes cepstrales en la frecuencia de Mel gracias a su manera de simular la percepción del sonido del oído humano.

A partir de construir un sistema clasificador de audio por los géneros musicales *blues*, música clásica, *country*, *rock* o *jazz;* se busca comparar y definir cuál de las dos redes neuronales entrenadas es más apta para esta tarea de clasificación. El criterio de selección estará basado en el porcentaje de aciertos y el tiempo de análisis de cada red neuronal.

1. SONIDO

El sonido es la sensación que se produce a través del oído en el cerebro y las causas físicas que lo provocan son las vibraciones de un medio elástico que pueden ser sólido, líquido y gaseoso. Estas vibraciones se producen por desplazamiento de las moléculas del aire debido a la acción de una presión externa. Cada molécula transmite la vibración a las que hay a su lado provocándose un movimiento en cadena (Sierra, 2011).

1.1. Características del sonido

El sonido es todo lo que oyen los seres humanos y los animales, resultado de los desplazamientos moleculares. Se transmite en forma de ondas, y depende de las características del medio en el que se propaga. Esta cualidad de onda le atribuye intrínsicamente una serie de propiedades o características de uso fundamental en su estudio.

1.1.1. Amplitud

Es un grado de movimiento de las moléculas de aire en una onda. En términos musicales es denominada intensidad. La intensidad y la amplitud están relacionadas proporcionalmente, mientras mayor amplitud mayor será la intensidad, esto se traduce a qué tan fuerte es el sonido.

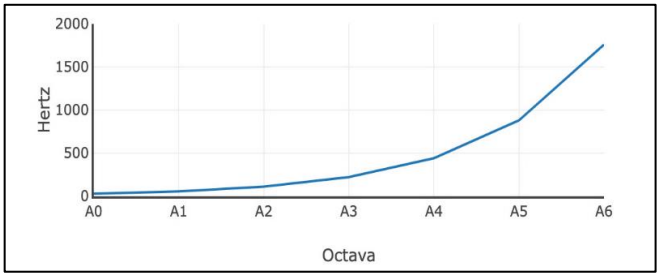
1.1.2. Frecuencia

La frecuencia está definida como un número de oscilaciones que una onda efectúa en un determinado intervalo de tiempo, se mide en hercios (Hz), y se interpreta como el número de ciclos por segundo.

En relación con la música la frecuencia está directamente relacionada con el tono y la altura de un sonido musical. Mientras mayor sea la frecuencia el tono es mayor y el sonido es más agudo, y mientras menor sea la frecuencia el tono será menor y el sonido es más grave. Las notas musicales tienen una frecuencia determinada, y los instrumentos musicales se afinan a teniendo como marco de referencia estas frecuencias.

El sistema musical está denotado por doce notas, divididas por octavas ordenadas de más graves a la izquierda a más agudas a la derecha. Cada octava aguda es el doble de la frecuencia de una octava grave, si se parten todas las octavas audibles a la mitad. Por ejemplo, un LA con una frecuencia de 440 Hz (esta nota es la utilizada para afinar instrumentos musicales), el LA en la octava anterior será más grave y tendrá una frecuencia de 220 Hz, y el LA en la octava siguiente será más agudo y tendrá una frecuencia de 880 Hz.

Figura 1. **Frecuencia en Hz de la nota LA por octava**



Fuente: elaboración propia, realizado con Matcha.io.

1.1.3. Intensidad

La intensidad es una cualidad para diferenciar sonidos fuertes a sonidos suaves. Está en función de la amplitud, la distancia a la que se inició el sonido y la capacidad auditiva del oyente. Cuando las oscilaciones de presión que alcanzan los oídos se encuentran en un determinado rango de frecuencias y de intensidad, se produce la sensación de oír.

1.1.4. Tono

El tono es la cualidad que permite distinguir entre un sonido agudo y otro grave. El tono está determinado principalmente por la frecuencia, aunque también puede cambiar con la presión.

1.1.5. Timbre

El timbre de un sonido es la cualidad con la que se puede distinguir dos sonidos de igual frecuencia e intensidad emitidos por dos focos sonoros diferentes. El timbre se debe a que generalmente un sonido no es puro y

depende principalmente del espectro de frecuencias que lo acompaña (Sierra, 2011).

2. MUSICA

En esencia, se puede decir que la música es el arte de ordenar los sonidos con el fin de crear una determinada emoción en el oyente (Alba, s.f.). El conocimiento de los componentes musicales y su terminología será de utilidad para el mejor entendimiento del caso de estudio realizado en capítulos posteriores.

2.1. Componentes musicales

A continuación, se describen los componentes musicales básicos y necesarios para describir correctamente las características de los géneros musicales.

2.1.1. Notas musicales

Las notas representan determinados sonidos musicales. Son representadas por signos que se describen en los espacios o en las líneas del pentagrama según su tono y duración. Las notas musicales son siete: do, re, mi, fa, sol, la, sí. Cada uno de los tonos en los que se puede tocar una nota tiene asociada una frecuencia, y cada octava que se utiliza, dobla o divide la frecuencia de una nota por un factor de dos.

2.1.2. Alteraciones a las notas

Las alteraciones son símbolos utilizados para modificar la altura de las notas. En la tabla I se muestra el símbolo y significado de cada una de estas alteraciones.

Tabla I. **Alteraciones a las notas y sus significados**

Alteración	Nombre	Efecto
♯	Sostenido	Sube medio tono a la altura de la nota
♭	Bemol	Baja medio tono a la altura de la nota
♮	Becuadro	Anula las alteraciones y devuelve la nota a su estado natural

Fuente: elaboración propia.

2.2. Melodía

Es la acertada sucesión de varios sonidos. Tiene una secuencia lineal a lo largo del tiempo y una identidad y significado en un entorno sonoro. Es una combinación de alturas y ritmo (Alba, s.f.).

Las melodías suelen estar formadas por una o más frases o motivos musicales que se repiten a lo largo de una canción o de una pieza musical en diversas formas.

2.3. Ritmo

Es la fuerza dinámica y organizativa de la música. Es el pulso o tiempo a intervalos constantes y regulares. Todo sonido puede convertirse en un ritmo: la

respiración, el sonido de un reloj encendido, las olas del mar, entre otros. Estos movimientos, regulares o imprevisibles, se suceden en el tiempo.

2.4. Duración

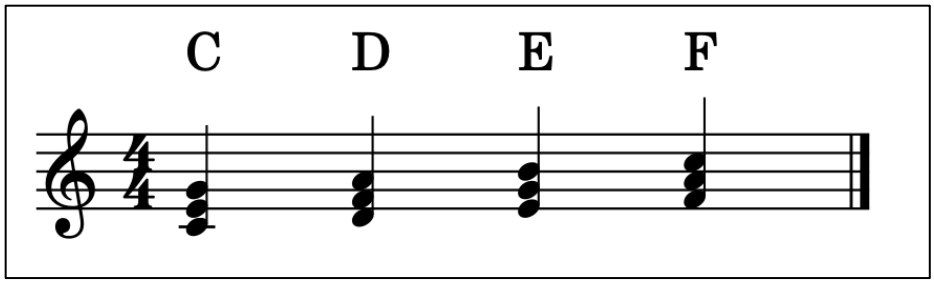
Los ritmos son formados al encadenar diferentes duraciones y valores rítmicos. Las duraciones más comunes son: redonda, blanca, negra, corchea y semicorchea. De izquierda a derecha, cada figura vale el doble de la anterior, de manera que una redonda equivale a dos blancas, una blanca equivale a dos negras y así sucesivamente.

2.5. Acordes

Un acorde es una reunión simultánea de tres, cuatro o más notas. El estudio de los acordes pertenece a la armonía, que es la teoría de los acordes, su esencia, sus oficios, usos y su sistema de clasificación (Alba, s.f.).

En la figura 2 se muestra un ejemplo de la escritura de los acordes do, re, mi y fa respectivamente.

Figura 2. **Ejemplo de cuatro acordes escritos en el pentagrama**



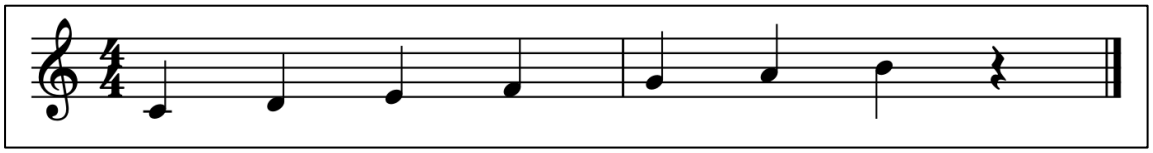
Fuente: elaboración propia, realizado con *flat.io*.

2.6. Escalas

Una escala es una sucesión ordenada de notas en forma ascendente (de un sonido grave a uno agudo) o descendente (de agudo a grave). Las escalas más comúnmente utilizadas en los géneros musicales a estudiar en este trabajo son:

- Diatónica: tiene siete sonidos, está formada por tonos y semitonos.

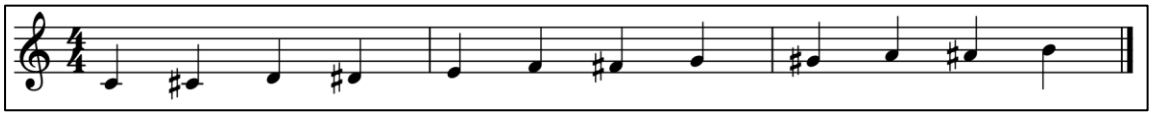
Figura 3. **Escala diatónica**



Fuente: elaboración propia, realizado con flat.io.

- Cromática: tiene doce sonidos, está formada exclusivamente por medios tonos.

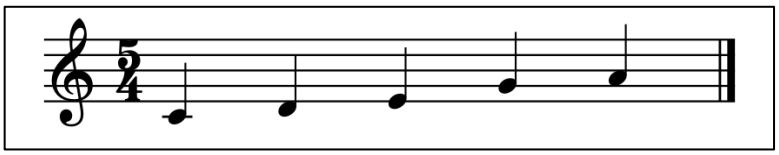
Figura 4. **Escala cromática**



Fuente: elaboración propia, realizado con flat.io.

- Pentatónica: tiene cinco sonidos, está formada por tonos e intervalos de tercera.

Figura 5. **Escala pentatónica**



Fuente: elaboración propia, realizado con flat.io.

3. GÉNEROS MUSICALES

Un género musical es una categoría que reúne composiciones musicales que comparten distintos criterios de afinidad, tales como su función, su instrumentación, el contexto social en que es producida o el contenido de su texto.

3.1. *Blues*

El *blues* es un género musical originado en las comunidades afroamericanas de los Estados Unidos a principios del siglo XX. Las canciones comúnmente tratan temas como la tristeza, la soledad, el trabajo duro y la lucha contra la opresión. Este género ha tenido una gran influencia en muchos otros géneros musicales, como el *rock* y el *jazz.*

Es caracterizado por una estructura musical simple y repetitiva, con un fuerte énfasis en el uso de la guitarra eléctrica y la armónica. Tiene como base la escala pentatónica.

3.2. Música clásica

La música clásica es un género musical desarrollado en Europa y América del Norte durante el siglo XVII al siglo XIX. Está caracterizada por su complejidad y refinamiento, y tiene una variedad de subgéneros y estilos como la ópera, la sinfonía, la música vocal y la música de cámara.

La música clásica se ejecuta principalmente con instrumentos de cuerda, viento, percusión y teclado, y se compone para ser interpretada en conciertos y teatros. Se basa en la tradición y en las reglas establecidas de la composición. Utiliza una notación simbólica compleja que le permite a los compositores prescribir de manera detallada el tiempo, la métrica, el ritmo, la altura y la ejecución precisa de cada pieza musical.

La música clásica utiliza la escala mayor, escala menor, escala menor melódica y escalas pentatónicas, aunque no está limitada únicamente a ellas.

3.3. *Country*

El *country* es un género musical que se originó en los Estados Unidos en el siglo XX. Recibió una gran influencia de la música *folk,* y se caracteriza por el uso de instrumentos acústicos como la guitarra, el banyo, la mandolina, el bajo y la armónica. Las canciones a menudo hablan sobre temas relacionados con la vida rural, el amor, el trabajo duro y los valores tradicionales. Ha sido de gran influencia en otros géneros como el *rock*.

Se caracteriza por utilizar acordes sencillos. Una estructura melódica de tres acordes y utiliza el patrón verso – estribillo – verso, emplear la escala pentatónica mayor y la escala mayor.

3.4. *Rock*

El *rock* es un género musical que generalmente consiste en un ritmo de tres acordes, un ritmo de fondo continuo, y una melodía. En sus inicios fue un derivado de múltiples géneros como el *blues*, R&B, *country, góspel*, *jazz* y *folk*. Todas estas influencias se combinaron en un género simple, con una estructura

parecida al *blues*, bailable y pegadiza. Los instrumentos musicales que normalmente se utilizan en el *rock* son: la guitarra eléctrica, teclados, bajo, batería y la voz.

3.5. *Jazz*

El *jazz* es un estilo que nace a finales del siglo XIX, mediante la confrontación de la música afroamericana con la europea, y que se expande en el mundo en el siglo XX. Sus características primordiales son: el swing, la improvisación y el fraseo.

Usualmente la melodía se construye en escalas mayores, menores, pentatónicas mayores y menores, y simétrica disminuida. Los instrumentos musicales que normalmente se utilizan en el *jazz* son: trompeta, trombón, saxofón, clarinete, batería, contrabajo, piano y guitarra.

4. ESPECTROGRAMAS

Una señal es una variación de una cantidad en el tiempo. Para el audio, la cantidad que varía es la presión en el aire. La información contenida en el dominio de la frecuencia es de utilidad para el procesamiento digital de señales (Roberts, 2020).

El espectro de frecuencias de una señal proporciona valiosa información en campos como reconocimiento de voz, audio, astronomía y diversas áreas de la ciencia y tecnología.

4.1. Series de Fourier

El matemático francés Jean Baptiste Joseph Fourier demostró en 1782 que una señal puede ser representada como una sumatoria infinita de senos y cosenos una vez algunas condiciones de carácter general se cumplan. A esta sumatoria se le conoce como series de Fourier. Las series de Fourier son útiles para descomponer señales periódicas en componentes simples.

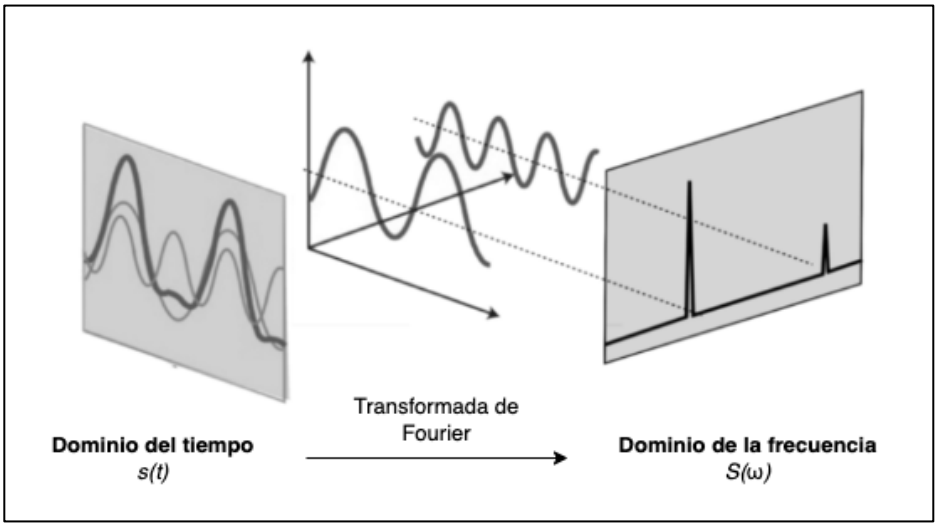
4.1.1. La transformada de Fourier

Las series de Fourier son de gran utilidad para representar señales periódicas. Sin embargo, en el procesamiento digital de señales se trabajan con señales no periódicas, de ello surge la necesidad de un método para representarlas. Esto se logra al garantizar que un segmento de la señal que se desea representar tenga un periodo infinito (Valiente, 2006).

Una señal de audio está compuesta de varias ondas de frecuencia individuales. Al tomar muestras de la señal a lo largo del tiempo, solo se captura las amplitudes resultantes. La transformada de Fourier resulta en una fórmula matemática que permite descomponer una señal en frecuencias individuales y la amplitud de la frecuencia.

Esto resulta en la conversión de una señal en el dominio del tiempo al dominio de la frecuencia. A este resultado se le llama espectro.

Figura 6. **Transformada de Fourier**



Fuente: Roberts (2020). *Understanding the Mel Spectrogram.* Consultado el 10 de octubre de 2021. Recuperado de https://medium.com/analytics-vidhya/understanding-the-mel-spectrogram-fca2afa2ce53.

4.1.2. Transformada rápida de Fourier

La transformada rápida de Fourier (FFT), es un método que permite calcular de manera eficiente la transformada discreta de Fourier. Es una herramienta poderosa que permite analizar el contenido de frecuencia de una señal. El único problema es cuando la frecuencia de la señal varía con el tiempo, siendo el caso de la mayoría de las señales de audio como la música o el habla. Estas señales se conocen como señales no periódicas. Para ello se utiliza la transformada de Fourier de tiempo corto.

4.1.3. Transformada de Fourier de tiempo corto

El análisis de señales en el dominio de tiempo-frecuencia (STFT, por sus siglas en inglés) comúnmente es realizado usando espectrogramas; por medio de la aplicación de la transformada de Fourier de tiempo corto a una señal dividida (Aguilar, 2020).

Para la generación de espectrogramas, la STFT se aplica a una señal, dividiéndola en varios segmentos de señales de tiempo corto desplazando una ventana de tiempo con algunas superposiciones, en un proceso llamado ventaneo. La ecuación general STFT de una señal S está dada por la ecuación:

$$S(m,k) = \sum_{i=0}^{n-1} s(n + mN')w(n)e^{-j\frac{2\pi}{N}nk}$$

Donde:

- $k = [0 : K]$ es el K-ésimo coeficiente de Fourier

- $K = \frac{N}{2}$ es el índice de frecuencia correspondientes a la frecuencia de Nyquist.

- $S(m, k)$ indica el índice (m) tiempo-frecuencia del espectrograma.

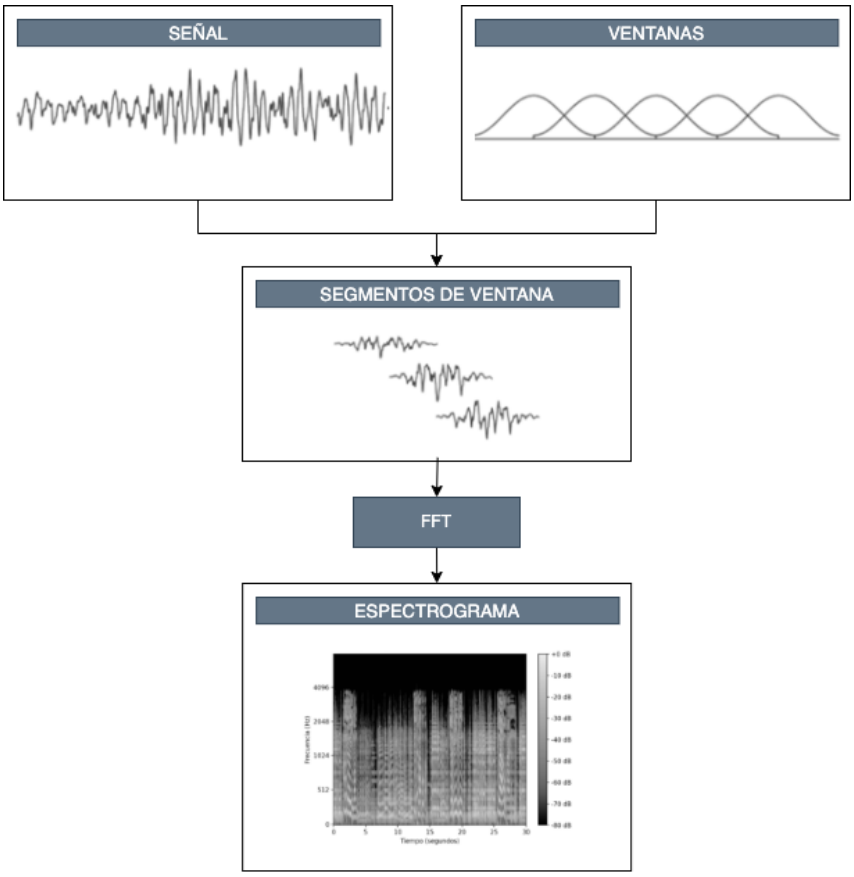
- N es la longitud del segmento de ventana.

- N'es el paso de desplazamiento de la ventana de tiempo.

- N'debe ser menor que N para producir una superposición entre las ventanas de tiempo.

- La función S depende de la función de ventana; y las formas de las ventanas vienen en tres distintas formas: simétrica, unimodal y gaussiana.

Figura 7. **Proceso de la STFT al aplicarla a una señal**



Fuente: elaboración propia, realizado con Draw.io.

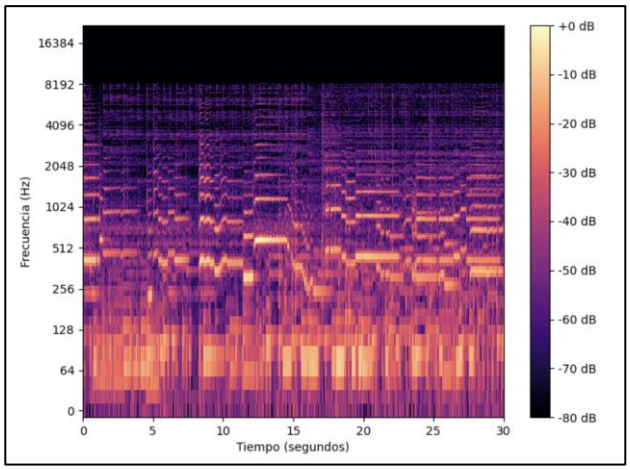
Un espectrograma presenta una relación entre la resolución de tiempo y la resolución de frecuencia; una ventana grande proporciona menor resolución en el tiempo y una mejor visualización de la frecuencia. Al obtener una ventana (es decir, un segmento de tiempo de la señal), las características espectrales son casi constantes; los segmentos obtenidos cambian la ventana de tiempo con cierta superposición.

El espectrograma se define como la magnitud de $S(m,k)$ representada como $A(m,k)$ como se muestra en la ecuación:

$$A(m,k) = \frac{1}{N} |S(m,k)|^2$$

La resolución del espectrograma se puede mejorar modificando la longitud de la ventana; una ventana ancha proporcionará una mejor resolución de frecuencia, pero baja resolución de tiempo (Aguilar, 2020).

Figura 8. **Imagen de un espectrograma de una canción de *rock***



Fuente: elaboración propia, realizado con Python y Matplotlib.

Se puede pensar en un espectrograma como un grupo de transformadas rápidas de Fourier apiladas una encima de la otra. Es una manera de representar visualmente el volumen o amplitud de una señal, ya que varía con el tiempo en diferentes frecuencias.

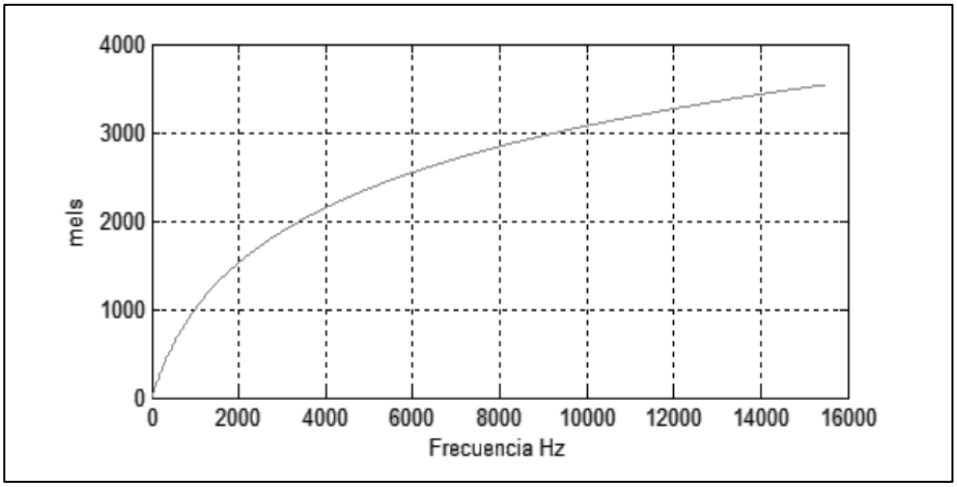
Como es posible observar en la figura 8, en un espectrograma se suele poner el eje vertical como una escala logarítmica de la frecuencia, en el eje horizontal se representa el tiempo, y en una tercera dimensión se representa la amplitud, de un par frecuencia-tiempo, representado por colores (Roberts, 2020).

4.2. La escala Mel

Los humanos no perciben las frecuencias en una escala lineal. Por ejemplo, la diferencia entre un do en la segunda posición a un do en la cuarta posición (de 65 Hz a 262 Hz) es mucho más perceptible que una diferencia entre un sol en la sexta posición a un LA en la sexta posición (1568 Hz a 1760 Hz), aunque la cantidad de Hertz de diferencia sean aproximadamente 200 en ambos casos.

La escala Mel fue propuesta por Stevens, Volkman y Neumann en 1937, es una escala musical basada en la percepción de tonos. En la escala Mel se observan tonos espaciados exponencialmente (la frecuencia), como tonos equiespaciados (los Mels). Esta propiedad es lo que hace que la escala Mel sea fundamental para el aprendizaje de máquina, ya que imita la manera en la que los humanos perciben el sonido.

Figura 9. **Relación de la frecuencia en Hertz con la escala de Mel**



Fuente: elaboración propia, realizado con Python y Matplotlib.

Se define que un sonido de 1000 Hz es también un sonido de 1000 Mels como punto de referencia. Una fórmula aproximada para la relación entre las frecuencias medidas en la escala de Mel y Hertz es la planteada en esta ecuación:

$$F_{mel} = 1{,}127.0148 * \log(1 + \frac{F_{Hz}}{700})$$

Donde:

- F_{mel} es la frecuencia en la escala de Mel
- F_{Hz} es la frecuencia en Hertz

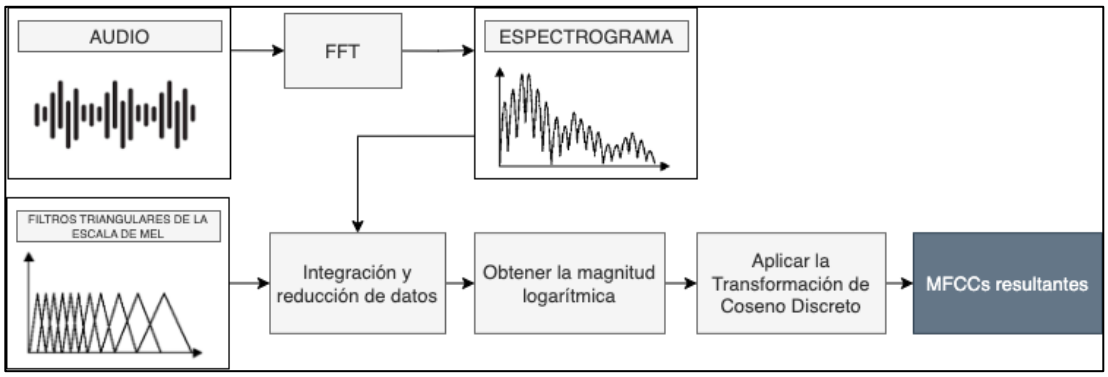
4.2.1. Coeficientes cepstrales en la frecuencia de Mel

Los coeficientes cepstrales en la frecuencia de Mel (MFCCs, por sus siglas en inglés), son la representación de una onda sonora en forma de una serie de coeficientes derivados de los coeficientes de la escala de Mel (Rodriguez, 2021).

Los MFCCs fueron originalmente utilizados en varias técnicas de procesamiento de voz. A medida que el campo de la recuperación de información musical comenzó a desarrollarse más como complemento del aprendizaje de máquina, se descubrió que también pueden representar excelentemente el timbre (Rodriguez, 2021). Para la obtención de los MFCCs se realiza el siguiente proceso:

- Convertir la señal de audio de frecuencia a la escala de Mel
- Obtener el logaritmo de la representación en la frecuencia de Mel del audio.
- Obtener la magnitud logarítmica y aplicarle la transformación de coseno discreto.

Figura 10. **Algoritmo de obtención de los MFCCs de un audio**



Fuente: elaboración propia, realizado con Draw.io.

5. INTELIGENCIA ARTIFICIAL

Durante cientos de años, los humanos han intentado entender cómo es posible que la materia que conforma a la vida humana puede percibir, entender, predecir y manipular un mundo mucho más grande y complicado que ella misma. El campo de la inteligencia artificial va más allá: no solo intenta comprender, sino que también se esfuerza en construir entidades inteligentes.

La inteligencia artificial es una ciencia relativamente nueva, su nombre tiene origen en 1956. Esta ciencia tiene muchas áreas de estudio que comprenden la solución de problemas, aprendizaje, percepción, demostración de teoremas y predicción de tendencias, entre otros (Espino, 2016).

En la actualidad, uno de los proyectos más ambiciosos de la informática es la inteligencia artificial; ya que ha sido utilizada para aportar conocimiento sobre el análisis y generación de información en diferentes áreas del conocimiento humano, una de ellas la música.

La inteligencia artificial sintetiza y automatiza las tareas intelectuales y es, por lo tanto, potencialmente relevante para cualquier ámbito de la actividad intelectual humana. En ese sentido, es un campo genuinamente universal (Russell, Norvig, Corchado y Joyanes, 2011).

La inteligencia artificial ha avanzado más rápidamente en la década pasada debido al mayor uso del método científico en la experimentación y comparación de propuestas. Ha logrado avances en el entendimiento de las bases teóricas de la inteligencia, y sus subcampos han encontrado cada vez

más elementos comunes con otras disciplinas (Russell, Norvig, Corchado y Joyanes, 2011).

5.1. Aprendizaje de máquina

En 1959, Arthur Samuel acuñó el término aprendizaje de máquina mientras trabajaba en IBM. Este término nace de la búsqueda de inteligencia artificial y del interés de los investigadores en que las máquinas aprendieran de los datos.

El aprendizaje de máquina, aprendizaje automático o Machine Learning, es fundamentalmente utilizar algoritmos para extraer información desde datos no procesados para representarlos en algún tipo de modelo. Este modelo es utilizado para inferir información acerca de otros datos que aún no han sido modelados. Esta rama de la inteligencia artificial ha sido de interés en las últimas décadas por sus múltiples aplicaciones y utilidades.

El término aprendizaje aplicado a computadoras puede ser visto como una manera de utilizar algoritmos para adquirir descripciones estructurales de los datos estudiados. Una computadora puede aprender algo al tener estructuras que representen a la información que están estudiando. Esta estructura permite que otra información que no fue proporcionada inicialmente pueda ser tratada y realizar operaciones con ella.

Las estructuras que se utilizan para describir la información, es decir el modelado, puede tomar varias formas: árboles de decisiones, regresiones lineales, redes neuronales, entre otros. Cada uno de estos modelos tiene una manera diferente de aplicar reglas a la información que es proporcionada o estudiada para predecir información que no es conocida o no es proporcionada.

5.2. Tipos de aprendizaje de máquina

Existen tres enfoques importantes en relación con el aprendizaje de máquina: el aprendizaje por reforzamiento, aprendizaje no supervisado y el aprendizaje supervisado.

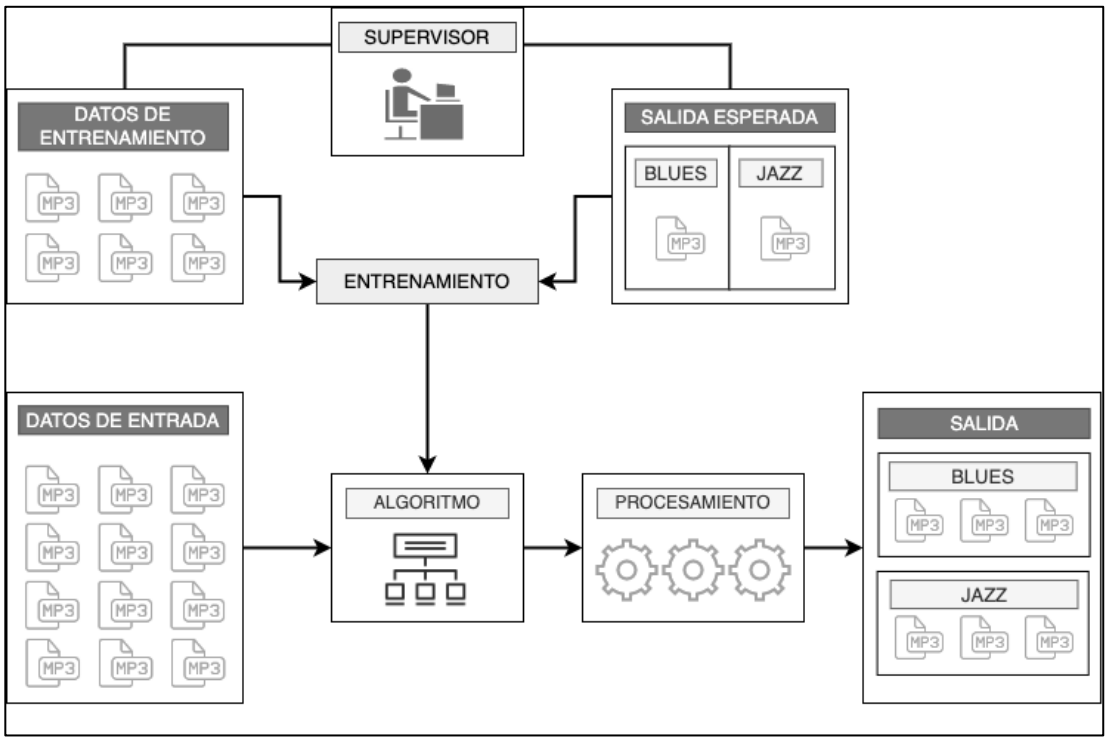
- Aprendizaje por reforzamiento: modelo de aprendizaje conductual, el algoritmo recibe retroalimentación del análisis realizado de los datos para obtener el mejor resultado.

- Aprendizaje no supervisado: utilizado cuando el problema requiere una gran cantidad de datos sin etiqueta. Este tipo de algoritmos no necesitan información acerca de los resultados esperados.

- Aprendizaje supervisado: como se aprecia en la figura 11, este inicia con un conjunto establecido de datos y una manera inicial de clasificarlos, esta información es proporcionada por la persona que está supervisando el entrenamiento.

La información en el aprendizaje supervisado incluye la información deseada. Se entrena al algoritmo entregando preguntas, denominadas características, y asignando una respectiva respuesta, denominada etiqueta. Este será el enfoque utilizado para el caso de estudio del capítulo 7.

Figura 11. **Esquema general del aprendizaje supervisado**



Fuente: elaboración propia, realizado con Draw.io.

6. REDES NEURONALES

Las redes neuronales son un tipo de modelo de aprendizaje de máquina; y han estado presentes desde al menos 50 años. A la unidad fundamental de una red neuronal se le denomina nodo, basándose en las neuronas biológicas de un cerebro de los mamíferos. Las conexiones entre neuronas también se basan en la biología del cerebro, ya que estas se desarrollan y entrenan con el paso del tiempo.

6.1. Neuronas biológicas

El conocimiento básico del funcionamiento de una neurona biológica es de utilidad para comprender de mejor manera los algoritmos próximos a describirse. Este apartado busca exponer las cualidades y funcionalidades más básicas y fundamentales de las neuronas biológicas.

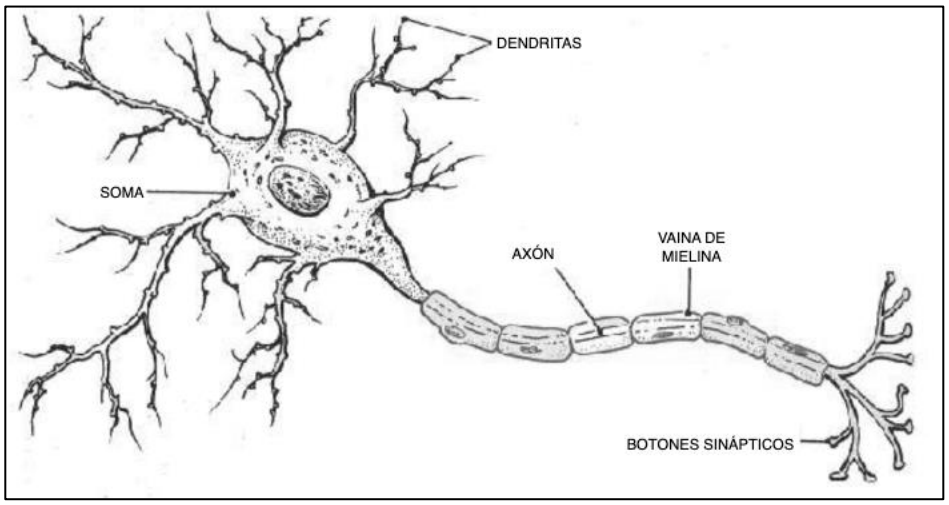
Las neuronas son células del cerebro que coleccionan, procesan y diseminan señales eléctricas. Las neuronas existen para comunicarse unas con las otras, y pasar impulsos electroquímicos a través de sinapsis, de una célula a otra, una vez el impulso sea lo suficientemente fuerte para seguir liberando químicos por medio de una hendidura sináptica. Básicamente, la fuerza del impulso debe sobrepasar un umbral mínimo, sino los químicos no se liberan.

Las neuronas están conformadas por cuatro elementos principales: soma, dendritas, axón y botones sinápticos, como es posible observar en la figura 12. El cuerpo o soma de la neurona es donde ocurren todos los procesos metabólicos de la neurona. Las dendritas son prolongaciones que nacen del

cuerpo o soma y que conforman una especie de ramas que recubren todo el centro de la neurona.

El axón es la única prolongación que nace del cuerpo o soma de la neurona, en la parte contraria a las dendritas. Se encarga de conducir el impulso eléctrico hasta los botones sinápticos, donde se liberan los neurotransmisores para informar a la siguiente neurona. Eventualmente, el axón se ramificará y se conectará a otras dendritas.

Figura 12. **Vista general de una neurona**



Fuente: Gollo (2008). *Synchronization between populations of neurons*. Consultado el 11 de octubre de 2021. Recuperado de https://digital.csic.es/bitstream/10261/18660/1/tesina.pdf.

Los botones sinápticos son los puntos de conexión entre el axón de una neurona y las dendritas de otras neuronas. La mayoría de los botones sinápticos envían señales desde el axón de una neurona a la dendrita de otra neurona.

Las neuronas son capaces de enviar señales electroquímicas generando algo conocido como potencial de acción. Esta señal viaja por medio del axón de la neurona y activa las conexiones sinápticas con otras neuronas. Los botones sinápticos, al recibir este potencial de acción secretan neurotransmisores. Estos neurotransmisores pueden excitar o inhibir a la neurona siguiente.

Un término importante por mencionar es la plasticidad, que se refiere a los cambios a largo plazo en la fuerza de las conexiones en respuesta al estímulo de los neurotransmisores discutido anteriormente. Las neuronas han demostrado formar nuevas conexiones e incluso migrar. Estos mecanismos de conexión y cambio son los conductores del proceso de aprendizaje del cerebro (Gollo, 2008).

6.2. Neuronas artificiales

Las neuronas artificiales están basadas en las neuronas biológicas. En este trabajo, al referirse a neurona desde este punto en adelante se refiere a una neurona artificial utilizada en una red neuronal.

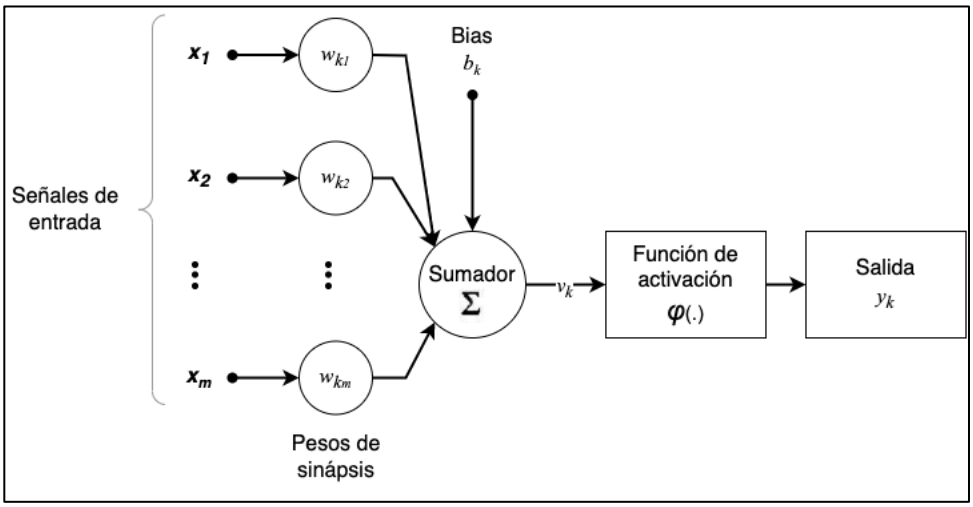
Las neuronas son unidades fundamentales de procesamiento de información en una red neuronal. Las neuronas se componen de tres elementos básicos, como se muestra en la figura 13.

- Conjunto de sinapsis: pueden ser vistos como los enlaces de conexión, tienen un valor asociado que representa su peso o fuerza, que puede ser positivo o negativo. Una señal que ingresa a una sinapsis conectada a una neurona es multiplicada por el peso que tiene la misma.

- Sumador: este elemento realiza una suma de las señales de entrada, tomando en cuenta el peso de su respectiva sinapsis de la neurona.

- Función de activación: limita la amplitud de la salida de la neurona, aplana los límites o el rango de amplitud permisible de una señal de salida a un valor finito.

Figura 13. **Modelo no lineal de una neurona**



Fuente: Hyakin (2009). *Neural Networks and Learning Machines*.

6.3. Comportamiento de una red neuronal

Las redes neuronales están compuestas de nodos o unidades conectadas por enlaces dirigidos. Un enlace de una unidad a otra sirve para propagar la activación, cada enlace tiene un peso asociado que determina la fuerza y señal de la conexión (Haykin, 2009).

El comportamiento de una red neuronal está definido por su arquitectura. Una arquitectura de una red es definida, en parte, por el siguiente número de neuronas que conforman la red, el número de capas que la conforma y el tipo de conexiones entre ellas.

6.4. Funciones de activación

Las funciones de activación son utilizadas en las redes neuronales para determinar el valor de salida de una neurona a partir de una entrada. Estas funciones son utilizadas en la capa de entrada, las capas ocultas, y la capa de salida. Existen múltiples funciones de activación, cada una con sus características únicas, y su utilización dependerá del problema y la arquitectura de la red neuronal.

6.4.1. Función de activación ReLU

La función ReLU (*Rectified Linear Unit*) es una función que devuelve el valor de la entrada si es mayor que cero, o cero si es menor o igual a cero. Es utilizada en las redes neuronales profundas para resolver problemas de aprendizaje no lineales. Está determinada por la función:

$$f(x) = \begin{Bmatrix} 0 & x < 0 \\ x & x \geq 0 \end{Bmatrix}$$

Donde:

- f es la función ReLU
- x es el valor de entrada

6.4.2. Función de activación *Softmax*

La función *Softmax* es utilizada en la capa de salida de una red neuronal para problemas de clasificación multiclase, donde cada neurona representa una clase diferente, por ejemplo, géneros musicales. Produce una salida que representa la probabilidad de que la entrada pertenezca a una clase determinada. Está definida por la función:

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

Donde:

- σ es la función Softmax
- $\vec{z}$ es el vector de entrada
- e^{z_i} es el valor de la exponencial del vector de entrada en la posición requerida.
- K es el número de clases a considerar.
- e^{z_j} es el vector de salida.

6.4.3. Función de Heaviside

La función Heaviside, también conocida como función de escalón de Heaviside, es una función matemática que se utiliza en análisis de señales y control de sistemas. Esta función toma valores de 0 para cualquier valor menor que 0, y 1 para valores mayores o iguales a 0. Está definida por la función:

$$H(x) = \begin{cases} 0 & x < 0 \\ 1 & x \geq 0 \end{cases}$$

Donde:

- H es la función Heaviside
- x es el valor de entrada

6.4.4. Función de activación de convolución

La convolución es una técnica matemática utilizada en el procesamiento de señales, análisis de imágenes y procesamiento de audio para combinar dos señales y obtener una señal resultante. La función de convolución se calcula multiplicando cada punto de una señal con una ventana deslizante de otra señal, y luego sumando los productos. Esto permite combinar las características de ambas señales de manera que se puedan extraer información útil de la señal resultante.

$$f * g(t) = \int_0^t f(t-s)g(s)ds \ \ donde\ t > 0$$

Donde:

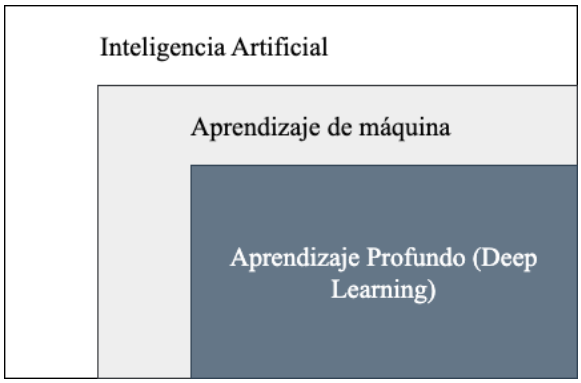
- $*$ es el operador de convolución
- f es la primera función
- g es la segunda función
- t es el tiempo

6.5. Deep Learning

Los conceptos de Deep Learning, aprendizaje de máquina e inteligencia artificial están relacionados a manera de subconjuntos.

El Deep Learning es un enfoque de la inteligencia artificial, un tipo de aprendizaje de máquina que alcanza gran potencia y flexibilidad mediante el aprendizaje de la representación del mundo, mediante conceptos jerárquicamente anidados. Se trata de formar conceptos complejos mediante la extracción y concatenación de conceptos muy simples (Ramos, 2020).

Figura 14. **Inteligencia artificial, aprendizaje de máquina y Deep Learning**



Fuente: Espino (2016). *Inteligencia Artificial*.

El Deep Learning se compone de varias técnicas y procedimientos algorítmicos en base a procesos por capas, tomando como base el aprendizaje de máquina. El objetivo es simular la manera de aprender de un ser humano, en específico la manera de reconocer imágenes, palabras o sonidos y el funcionamiento del cerebro, realizado a través de las neuronas.

6.6. El Perceptrón

El perceptrón es un modelo lineal utilizado para la clasificación binaria. Para una red neuronal, un perceptrón es una neurona que está utilizando la función escalón de Heaviside como su función de activación.

Para producir la entrada a la función de activación, en este caso la función escalón de Heaviside, se realiza el producto punto de la entrada x_n y los pesos de las conexiones w_n. La salida de la función de activación es la salida del perceptrón, y provee de una clasificación de los valores de entrada.

6.7. Redes neuronales Perceptrón Multicapa

Las redes neuronales perceptrón multicapa (MLP, por sus siglas en inglés) son redes de propagación hacia adelante, compuesta por dos o más capas de neuronas con posibilidad de tener diferentes funciones de activación.

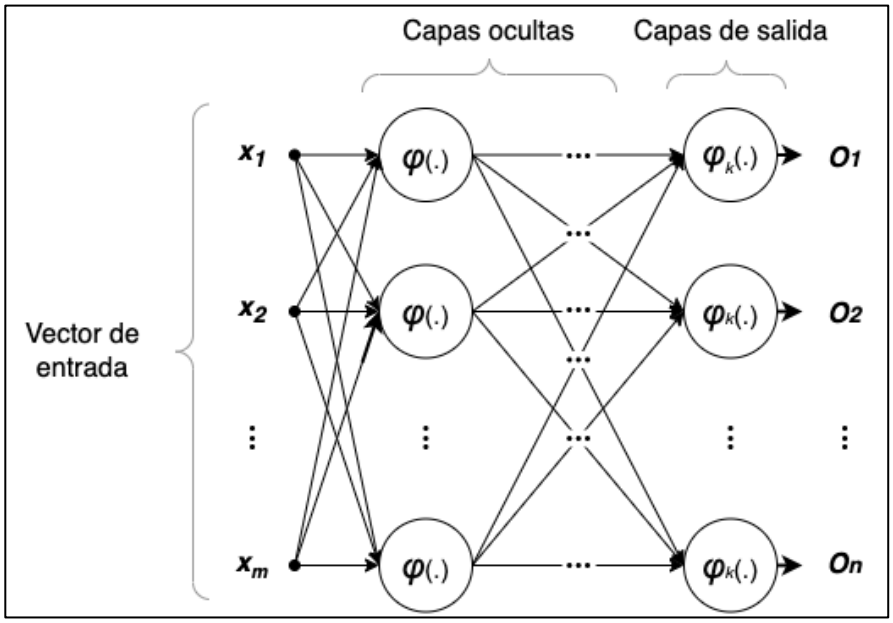
Estas redes neuronales aprovechan la naturaleza paralela de las redes neuronales para reducir el tiempo requerido por un procesador de secuencia para determinar la salida adecuada a partir de una entrada.

6.7.1. Topología de una MLP

Una MLP está formada por el vector de entrada, una o más capas ocultas y la capa de salida; las capas ocultas pueden tener cualquier número de neuronas. El tamaño del vector de entrada y el número de neuronas de la capa de salida se seleccionan de acuerdo con las entradas y salidas de la red respectivamente.

Las neuronas de cada capa deben tener la misma función de activación, pudiendo ser esta función diferente a la de otras capas. En la figura 15 se muestra cómo se ve una red neuronal con una entrada de un vector $\vec{x}$ con m valores, y una salida $\vec{o}$ con n valores, pudiendo diferir la cantidad de valores de entrada con la cantidad de valores de salida. La función de activación está representada por φ_k, denominando que esta puede ser diferente en cada capa, pero toda la capa debe tener la misma.

Figura 15. **Vista general de una MLP**



Fuente: elaboración propia, realizado con Draw.io.

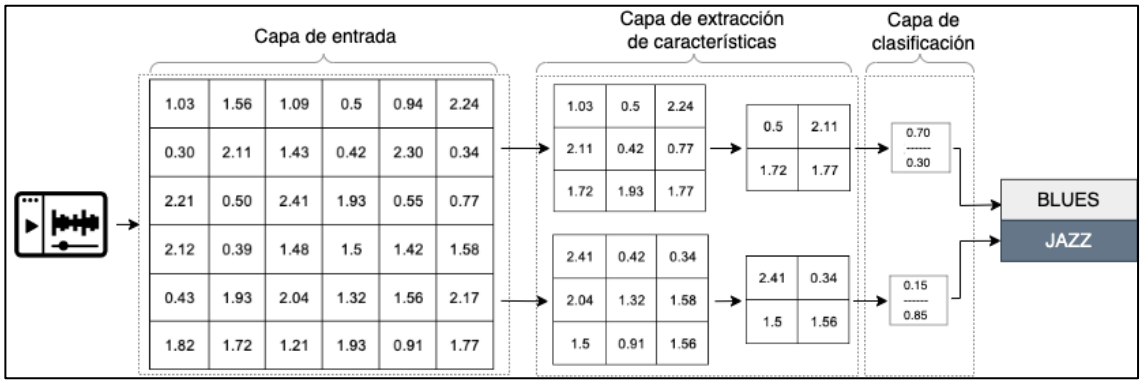
6.7.2. Redes neuronales convolucionales

Las redes neuronales convolucionales (CNN, por sus siglas en inglés), son una variante de las redes neuronales MLP, con la diferencia que las CNN realizan operaciones de convolución entre los parámetros de la red y los datos

de entrada, en lugar de un producto punto entre las entradas y los pesos (Ramos, 2020).

El objetivo de las redes neuronales CNN es aprender características de alto nivel en los datos a través de convoluciones. Son ampliamente utilizadas en el reconocimiento de objetos en imágenes y clasificación de imágenes. Pueden detectar caras, individuos, señales de carretera, y muchos otros aspectos visuales de la información. También son buenas para analizar sonido y para el análisis de texto vía reconocimiento óptico de caracteres.

Figura 16. **Vista general de una red neuronal CNN**



Fuente: elaboración propia, realizado con Draw.io.

La amplia gama de aplicaciones de las redes neuronales CNN es una de las razones por las que el mundo reconoce, y recientemente depende, del poder del Deep Learning. Las redes neuronales CNN están dominando en el desarrollo de la visión de las máquinas, que tiene aplicaciones en automóviles autónomos, robótica, drones, y medicina. También han sido utilizadas para resolver tareas difíciles como la generación y traducción de lenguaje natural y análisis de sentimientos.

En este trabajo la utilización de redes neuronales CNN se basa en la extracción de parámetros característicos de los espectrogramas de Mel anteriormente producidos a partir de los archivos de audio en el proceso de clasificación de géneros musicales.

6.7.3. Topología de una CNN

Las redes neuronales CNN transforman los datos de entrada desde una capa de entrada por medio de todas las otras capas conectadas en un conjunto de puntuaciones de clase dadas por la capa de salida.

La arquitectura de la CNN es un gráfico acíclico compuesto por capas de cada vez menos nodos, donde cada capa se alimenta de la siguiente (Haykin, 2009). Por ejemplo, la figura 16 contiene tres grupos mayores:

- La capa de entrada: acepta una entrada tridimensional, generalmente en la forma del espacio (largo y ancho), de la imagen y una profundidad representando los canales de color (generalmente tres, por la representación RGB del color).

- Capas de extracción de características: estas capas tienen un patrón repetitivo de una capa de convolución y otra capa de agrupación.

- Capas de clasificación: una o muchas capas conectadas que toman las características de orden alto y producen probabilidades de clase o puntuaciones.

6.7.4. Capas de convolución

Con la finalidad de que un nodo no tenga un conjunto impredecible de parámetros libres para aprender, los pesos en una capa se comparten en toda la capa. Los filtros aplicados en una capa convolucional son un conjunto único de filtros, que se deslizan a través del conjunto de datos de entrada (Haykin, 2009).

Este filtro está basado en cuatro parámetros ajustables:

- El tamaño: el área del filtro que recorrerá la imagen. Mientras más grande, mejor será la precisión de la clasificación, pero decaerá la eficiencia de la red.

- La profundidad: define el número de nodos en la capa.

- La zancada: es una medida de espacio entre las neuronas. Por defecto tiene el valor de 1.

- El relleno cero: es el proceso de establecer valores extremos de cada campo en cero. Aumentar el grado de relleno puede causar disminución en la efectividad.

6.7.5. Capa ReLU

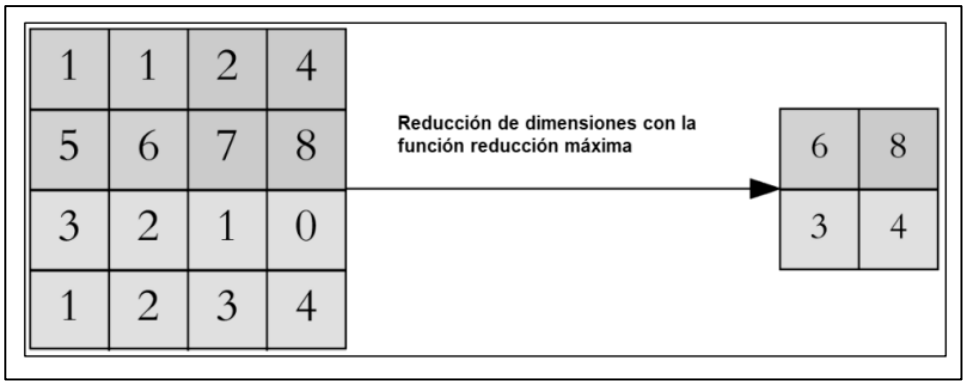
Después de pasar por el proceso de convolución, los mapas de características pasan por otro estado denominado activación utilizando la función de activación ReLU.

6.7.6. Capas de agrupación

Apilar las capas convolucionales permite una arquitectura que crea efectivamente características para datos de entrada complejos y ruidosos. Es común conectar capas convolucionales con capas de agrupación, esto con el fin de reducir las dimensiones espaciales ancho y alto del volumen de entrada para la siguiente capa convolucional.

La agrupación es una operación sobre mapas de entidades, donde se agregan múltiples valores de entidades en un solo valor en base a un criterio, como el mayor o el más común.

Figura 17. **Ejemplo de capa de agrupación por reducción máxima**



Fuente: Coursera (s.f.). *Aprendizaje automático*. Consultado el 20 de noviembre de 2021. Recuperado de https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2#a86a.

6.7.7. Capa de salida

Luego de haber procesado múltiples capas de convolución y agrupación, es necesario que la salida sea en forma de clase; la capa de salida tiene una función de pérdida, para calcular el error en la predicción. Es necesario que ingrese un vector unidimensional, este vector termina convirtiéndose en el vector de características, cada valor de este vector corresponde a un puntaje de clase y cada clase es como una categoría que podrá utilizar el algoritmo para clasificar.

7. CASO DE ESTUDIO: SISTEMA CLASIFICADOR

7.1. Descripción del caso de estudio

La utilización de técnicas de aprendizaje de máquina, en específico redes neuronales profundas, han sido demostradas ser efectivas para una amplia gama de tareas de clasificación. En el presente caso de estudio se pretende realizar un análisis de dos distintas técnicas de clasificación musical utilizando las técnicas de Deep Learning: redes neuronales perceptrón multicapa y redes neuronales convolucionales.

La clasificación se realizará a partir del procesamiento y análisis de la información contenida en archivos digitales de música para encontrar parámetros representativos que permitan utilizar técnicas de Deep Learning para determinar y clasificarlos por cinco diferentes géneros musicales.

Se utilizará la representación por espectrogramas de Mel, los MFCCs y otros parámetros numéricos a través de transformaciones para entrenar a ambas redes neuronales y así comparar su desempeño por acierto.

Como producto final, se busca crear un sistema clasificador, que tenga como entrada un archivo digital de música y como salida el archivo clasificado por género y un reporte de los parámetros que se utilizaron en su clasificación.

7.2. Tecnologías por utilizar

Existen numerosos entornos de trabajo, lenguajes de programación y librerías para crear y entrenar redes neuronales. Algunos de los factores importantes a considerar son la popularidad del lenguaje, las facilidades que da utilizarlo, el tamaño de la comunidad que lo utiliza, la cantidad de documentación y soporte disponible.

7.2.1. Lenguaje de programación Python

Para este caso de estudio se utilizará el lenguaje Python, dada su versatilidad y sencillez, además del enorme ecosistema de librerías y utilidades no solo de aprendizaje de máquina, sino también de manejo de arreglos, archivos digitales de audio, interfaz de usuario, entre otros.

Python provee soporte sobresaliente para todos los pasos del proceso de aprendizaje de máquina: importación y manipulación de datos, creación de modelos, entrenamiento de modelos, creación de gráficos, entre otros. Además, proporciona una sintaxis muy clara, fácil de leer y entender.

7.2.2. Librerías por utilizar

Se detallan algunas de las librerías utilizadas en la creación del sistema clasificador. Tomar en cuenta que no son las únicas utilizadas, la lista completa de librerías y herramientas utilizadas se encuentra en el repositorio de código compartido en el apéndice 5.

7.2.2.1. Librosa

Librosa es un paquete de Python para análisis de audio y música. Proporciona los componentes básicos necesarios para crear sistemas que tengan o trabajen de alguna manera con archivos digitales de audio o música. En específico este paquete fue utilizado para calcular el espectrograma de un audio en la escala de Mel y leer porciones del archivo a analizar.

7.2.2.2. Pandas

Pandas es una librería de código abierto que proporciona estructuras de datos y herramientas de análisis de datos de alto rendimiento y fáciles de usar para el lenguaje de programación Python.

7.2.2.3. Keras

Keras es un marco de trabajo de alto nivel para el aprendizaje de máquina, escrito en Python. Es utilizado para entrenar redes neuronales, y para utilizar algoritmos de aprendizaje de máquina.

Este marco de trabajo se basa en su habilidad de trabajar capas. Las capas son componentes básicos de las redes neuronales en Keras. Cada capa consta de una función de cálculo de tensor de entrada y tensor de salida.

7.2.2.4. Sklearn

Es una librería de aprendizaje de máquina para Python. Permite realizar varios algoritmos de clasificación, regresión y agrupamiento de valores.

7.3. Preparación

Antes de realizar el sistema es necesario crear una base de datos con información para entrenar a las redes neuronales, y realizar la decisión de qué herramienta se utilizará para la creación de esta base de datos.

Lo principal es encontrar suficientes canciones para el entrenamiento y clasificación musical de cada uno de los géneros a estudiar. Convenientemente, ya existe una base de datos que ocupa 1.32 GB con más de 1,000 canciones que utilizó G. Tzanetakis en el año 2002, en su estudio *Musical genre classification of audio signals.*

7.3.1. Preparación de la información

La base de datos de G. Tzanetakis cuenta con 1,000 extractos de canciones de 30 segundos cada uno, categorizados en 10 géneros musicales diferentes. La base de datos está compresa en formato ZIP para ser extraíble en cualquier computador, y cada audio está en el formato AU. Librosa trabaja con archivos con formato WAV, por lo cual se debe realizar una transformación masiva de esta información para poder manipularla.

Otro punto clave es eliminar los géneros que no se tomaron en cuenta en este caso de estudio. Los géneros para considerar son: *blues*, clásica, *country*, *jazz* y *rock*. Se deben eliminar los géneros: disco, *hip-hop*, metal, *pop* y *reggae* que están incluidos en la base de datos.

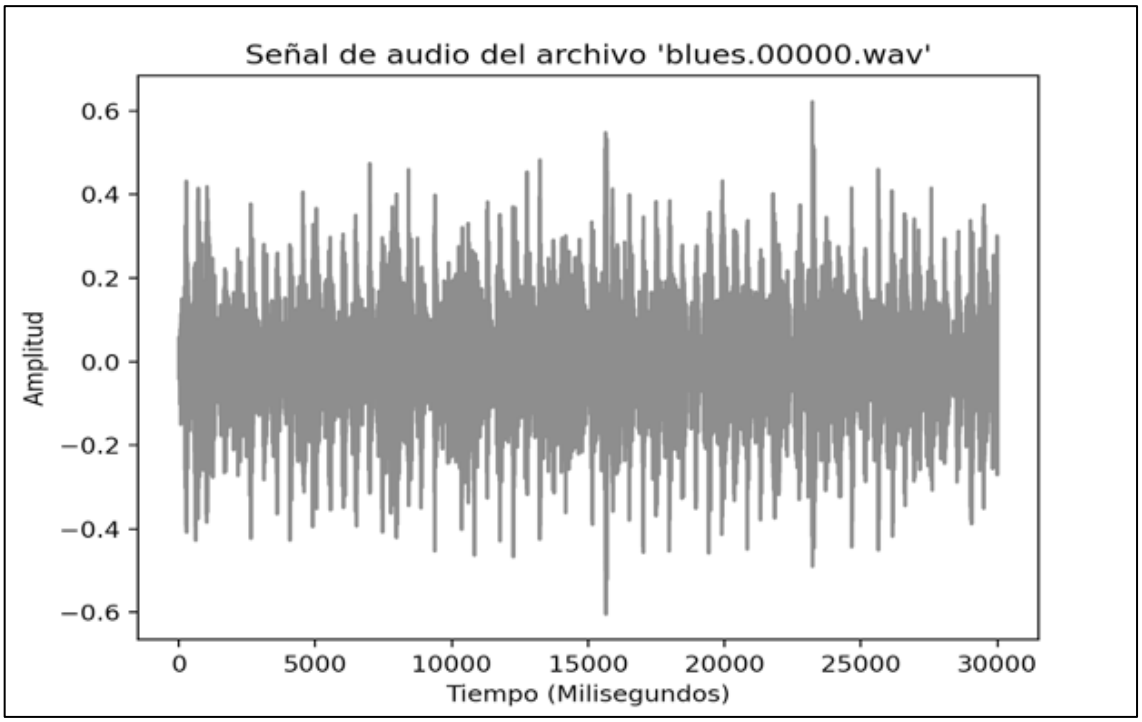
Con esta eliminación se obtendrá una base de datos con 500 archivos digitales de audio, 100 de cada género musical, divididos en carpetas con su

respectivo nombre del género. El código utilizado para realizar esta limpieza de datos se encuentra en el apéndice 5.

7.3.2. Extracción de características y parámetros musicales

Para iniciar con el sistema clasificador, el primer paso será leer un archivo digital de audio, y extraer su respectiva representación en algo conocido como una serie de tiempo. Esta representación permitirá obtener las métricas que se utilizarán como entrada de las dos redes neuronales. Por ejemplo, para el archivo *blues.000.wav*, que se encuentra en la base de datos dentro del folder del género *blues*, se obtiene la señal de audio de la figura 18.

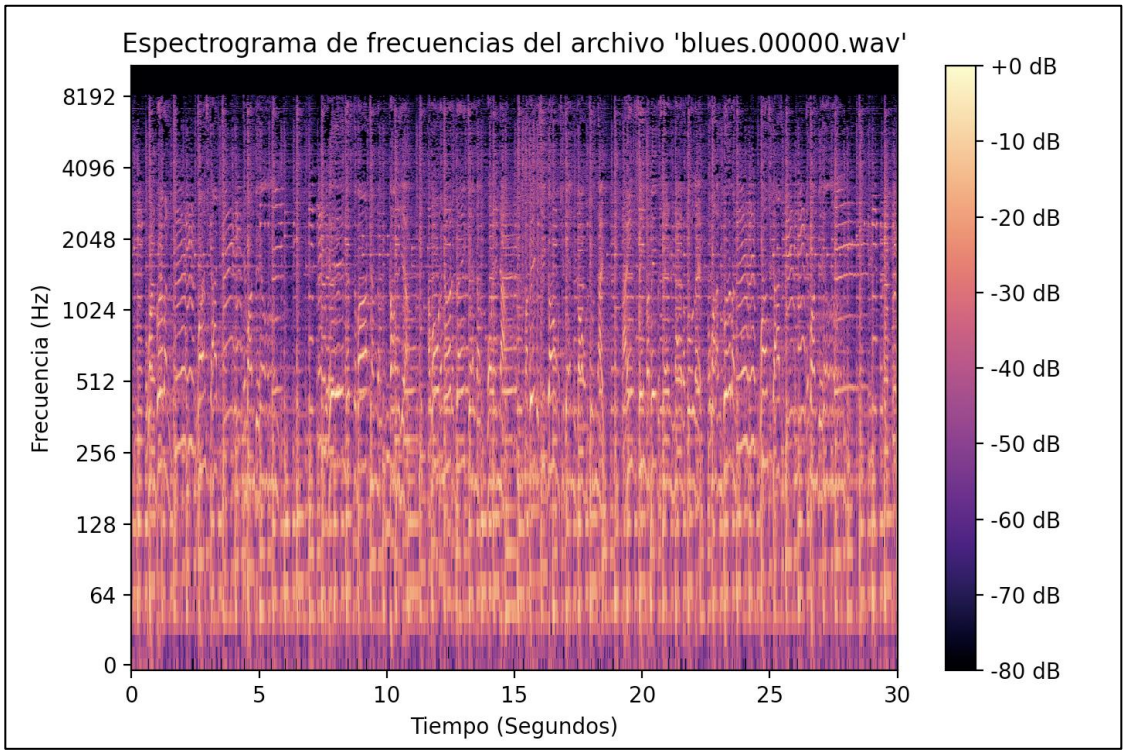
Figura 18. **Serie de tiempo de un archivo de audio**



Fuente: elaboración propia, realizado con Python, Librosa y Matplotlib.

Una vez obtenida la representación en el tiempo de la amplitud del sonido, es posible generar la conversión a un espectrograma. Este espectrograma está aún en una escala de frecuencias normal, y no es útil para realizar el análisis de género musical que se busca.

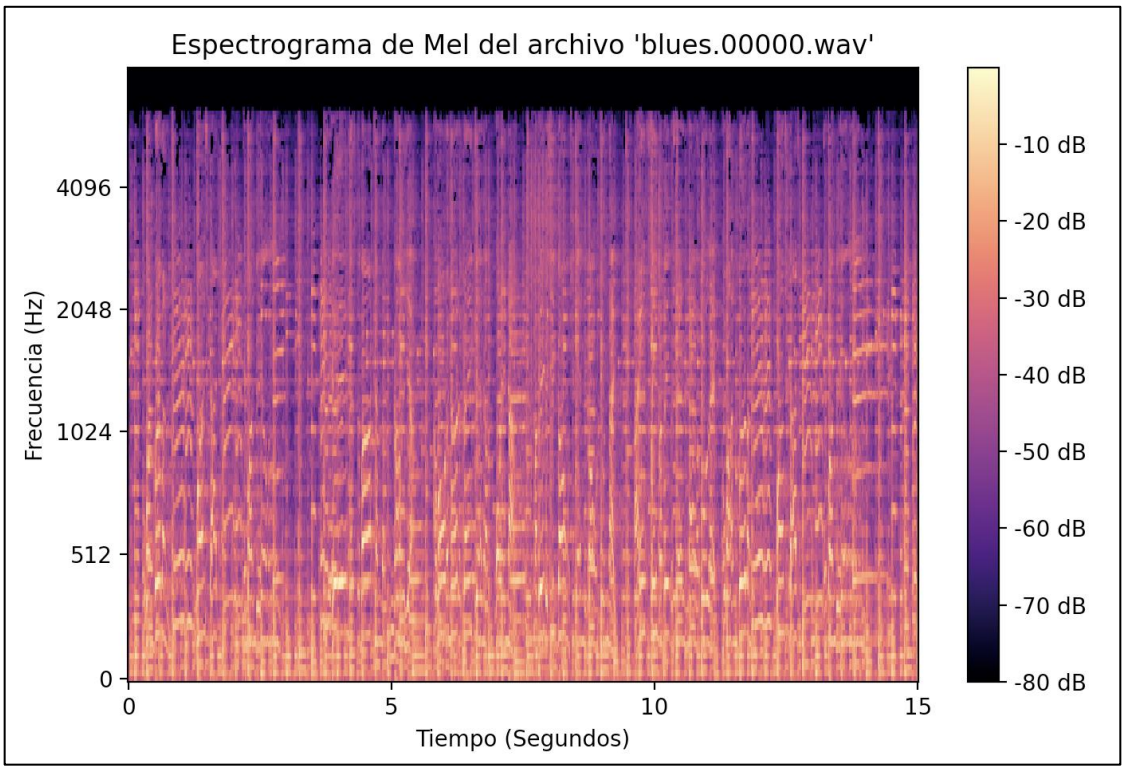
Figura 19. **Espectrograma de un archivo de audio**



Fuente: elaboración propia, realizado con Python, Librosa y Matplotlib.

A partir de generar el espectrograma de frecuencias, ya es posible hacer la conversión a la escala de Mel, y crear un diagrama parecido al de la figura 20. El objetivo de llegar hasta este punto es convertir la información del archivo de audio a una escala parecida a la que un humano puede interpretar auditivamente.

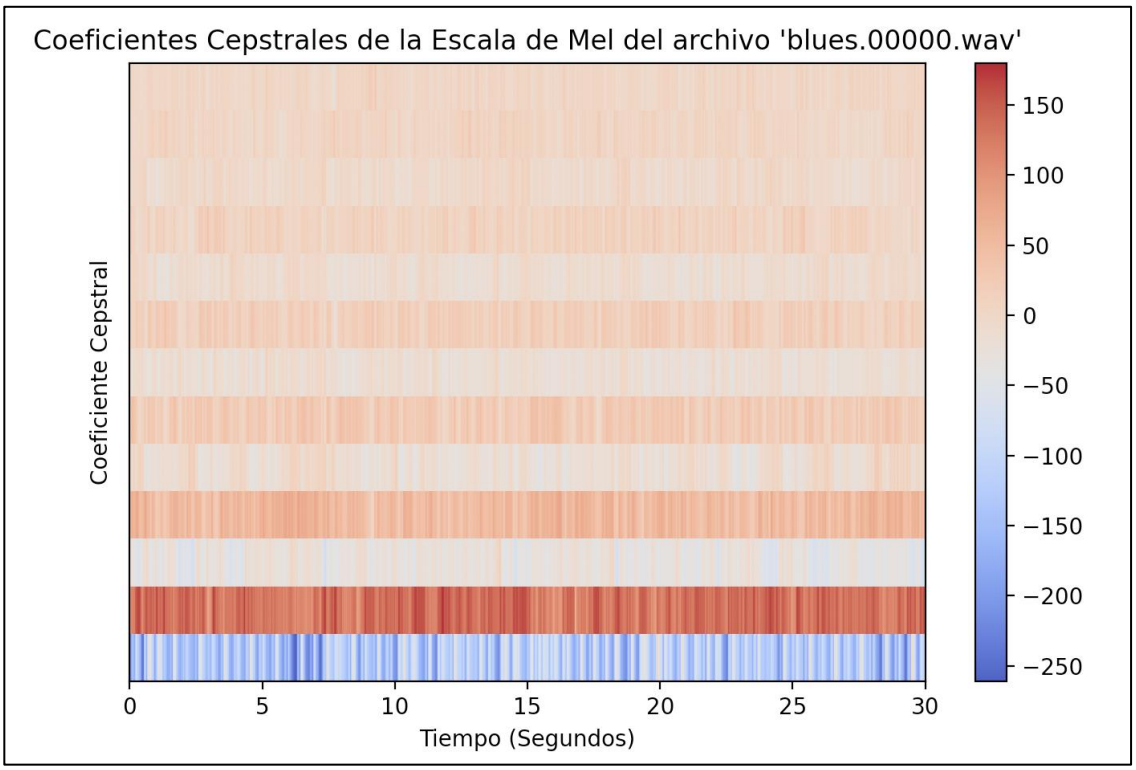
Figura 20. **Espectrograma de Mel de un archivo de audio**



Fuente: elaboración propia, realizado con Python, Librosa y Matplotlib.

El último paso para conseguir los datos de entrada de las redes neuronales es obtener los MFCCs. Estos coeficientes numéricos básicamente una representación del sonido basados en la percepción auditiva humana, y como se discutió en capítulos anteriores, son ampliamente utilizados para reconocimiento automático de voz y características del sonido. Es posible encontrar el código completo de cómo se obtuvieron estos valores en el apéndice 3.

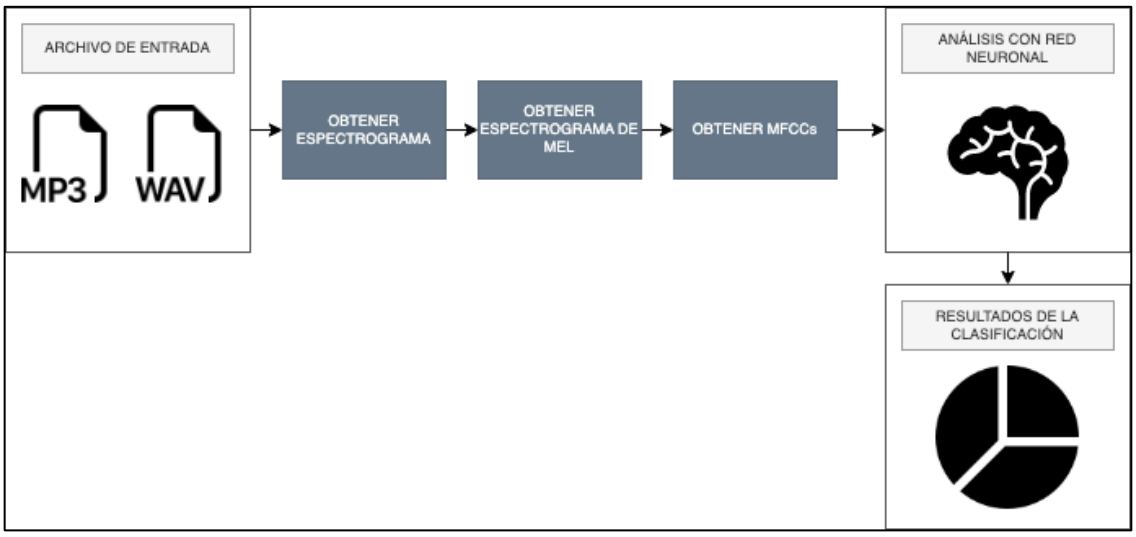
Figura 21. **MFCCs de un archivo de audio**



Fuente: elaboración propia, realizado con Python y Matplotlib.

El proceso que se realizó para un solo archivo digital se deberá replicar para todos los archivos en la base de datos para obtener ambas redes neuronales entrenadas y listas para recibir los archivos de entrada del usuario final. El diagrama resumido de los pasos anteriores puede observarse en la figura 22.

Figura 22. **Pasos para analizar y clasificar un archivo de audio**



Fuente: elaboración propia, realizado con draw.io.

7.4. Obtención de los datos de entrenamiento de las redes neuronales

Es necesario obtener los MFCCs de los 500 archivos de prueba y entrenamiento albergados en la base de datos, para ello se debe iterar y extraer los coeficientes de los archivos uno por uno, se creará una función específicamente para manejar esto.

Tras haber creado una función que devuelve los datos de todas las canciones, lo único que resta es transformarlos en información útil para ser procesada por Keras para crear ambas redes neuronales, esta información consiste en datos y etiquetas de prueba, y datos y etiquetas de entrenamiento.

A partir de este punto, los datos de entrenamiento para ambas redes neuronales ya están disponibles para ser utilizadas, y únicamente resta configurar las redes y entrenarlas para que empiecen a reconocer los géneros

musicales de las canciones. El código de esta sección se encuentra en el apéndice 5 de este trabajo.

7.5. Creación y entrenamiento de una CNN

La CNN utilizará múltiples capas, cada una con un propósito en específico para permitirle clasificar los géneros musicales de la mejor manera.

Las capas utilizadas se detallan a continuación:

- Capa convolucional: esta es la capa inicial de la red neuronal, utilizará 16 filtros de 3X3, con activación ReLU, para convertir los datos de entrada en un mapa de activación en dos dimensiones.

- Capa de agrupación: esta capa obtiene todos los valores máximos del mapa de activación generado en la capa anterior. En este caso, la ventana de visión de esta capa es de 2X4.

- Capa convolucional: se realizará nuevamente una convolución a los datos utilizando 32 filtros de 3X3 y activación ReLU.

- Capa de agrupación: otro filtro de valores máximos con una ventana de visión de 2X4.

- Capa de aplanamiento: esta capa se utilizará para linealizar los datos.

- Capa densa: se adjuntará una capa densa de 64 neuronas interconectadas con activación ReLU.

- Capa de pérdida: se definió una capa de 25 % de pérdida para evitar que la red neuronal se sobreajuste.

- Capa densa: esta es la última capa de la red neuronal, cuenta con activación Soft*max* para normalizar la salida y crear un vector de probabilidad de 5 espacios, donde cada uno representa un género musical y su valor asociado es la probabilidad de que la canción sea perteneciente a ese género.

Después de crear la red, será necesario compilarla y entrenarla. Se definirán 15 épocas de entrenamiento para esta red, valor determinado a prueba y error considerando la cantidad de entrenamientos que lleva a la red neuronal a tener un sobreajuste. Una vez finalizado este proceso, la red neuronal ya está lista para ser utilizada para predecir géneros musicales. El código de esta sección se encuentra en el apéndice 5 de este trabajo.

7.6. Diseño y entrenamiento de una MLP

La MLP utilizará cinco capas diferentes para clasificar los géneros musicales.

- Capa densa: la primera capa de la red neuronal será una capa densa de 128 neuronas con activación ReLU.

- Capa densa: a partir del resultado anterior, se aplicará una capa densa de 64 neuronas, también con activación ReLU.

- Capa de aplanamiento: se convertirá el resultado de las capas anteriores a una salida linealizada.

- Capa de pérdida: agregar una capa de pérdida del 25 % para evitar sobreajuste de la red neuronal.

- Capa densa: esta es la última capa de la red neuronal, cuenta con activación *Softmax* para normalizar la salida y crear un vector de probabilidad de 5 espacios, donde cada uno representa un género musical y su valor asociado es la probabilidad de que la canción sea perteneciente a ese género.

Después de crear la red, será necesario compilarla y entrenarla. Se definirán 10 épocas de entrenamiento para esta red, valor determinado a prueba y error considerando la cantidad de entrenamientos que lleva a la red neuronal a tener un sobreajuste. Una vez finalizado este proceso, la red neuronal ya está lista para ser utilizada para predecir géneros musicales. El código de esta sección se encuentra en el apéndice 5 de este trabajo.

7.7. Desarrollo de la aplicación

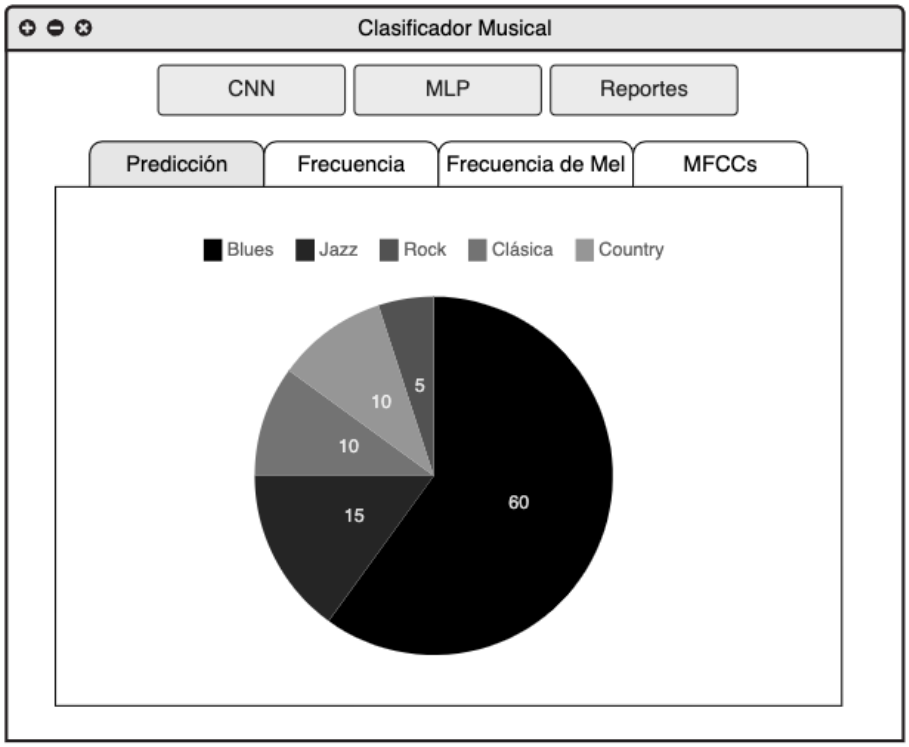
El caso de estudio propuesto da pauta a una aplicación de escritorio sencilla. La aplicación contará con la funcionalidad de clasificar un archivo digital con extensión MP3 o WAV, y proveerá de información acerca del género en el que se clasificó la canción.

7.7.1. Vista general de la aplicación

La aplicación contará con una vista simple y concisa de las acciones a realizar, se podrán clasificar las canciones utilizando una CNN o una MLP. Ambas funcionalidades estarán controladas por un botón que las activará correspondientemente, como se puede observar en la figura 23. La aplicación

únicamente mostrará la predicción final, y los diagramas del espectrograma de frecuencia, espectrograma de frecuencias de Mel, y los MFCCs del archivo analizado.

Figura 23. **Maqueta de la vista inicial de la aplicación**



Fuente: elaboración propia, realizado con Moqups.com.

7.8. Resultados obtenidos

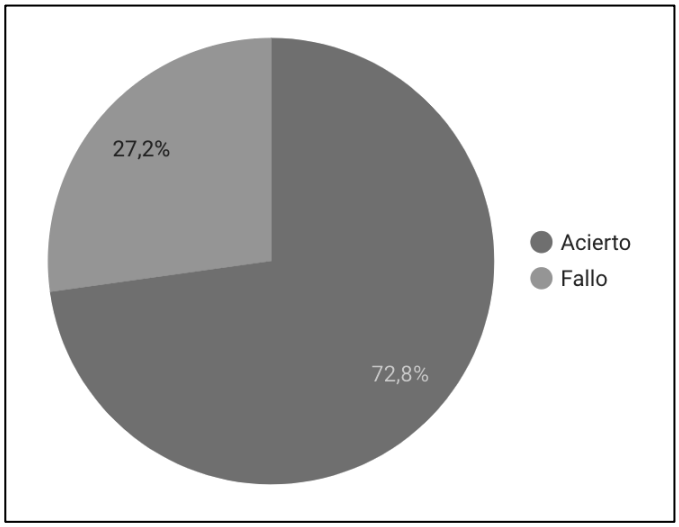
Luego de finalizar todos los servicios y funciones necesarios para analizar el género musical de un archivo de audio, es posible construir un análisis de qué tan bien categorizan las redes neuronales los géneros musicales, en el desempeño de cantidad de aciertos, y también en el

rendimiento del tiempo que toman en predecir. A partir de analizar los 500 archivos de audio y cuantizar sus aciertos y desempeño, se obtuvieron los descritos a continuación.

7.8.1. Resultados de la CNN

La CNN tuvo un porcentaje de acierto del 72.8 % de las veces, con 364 aciertos de 500 análisis.

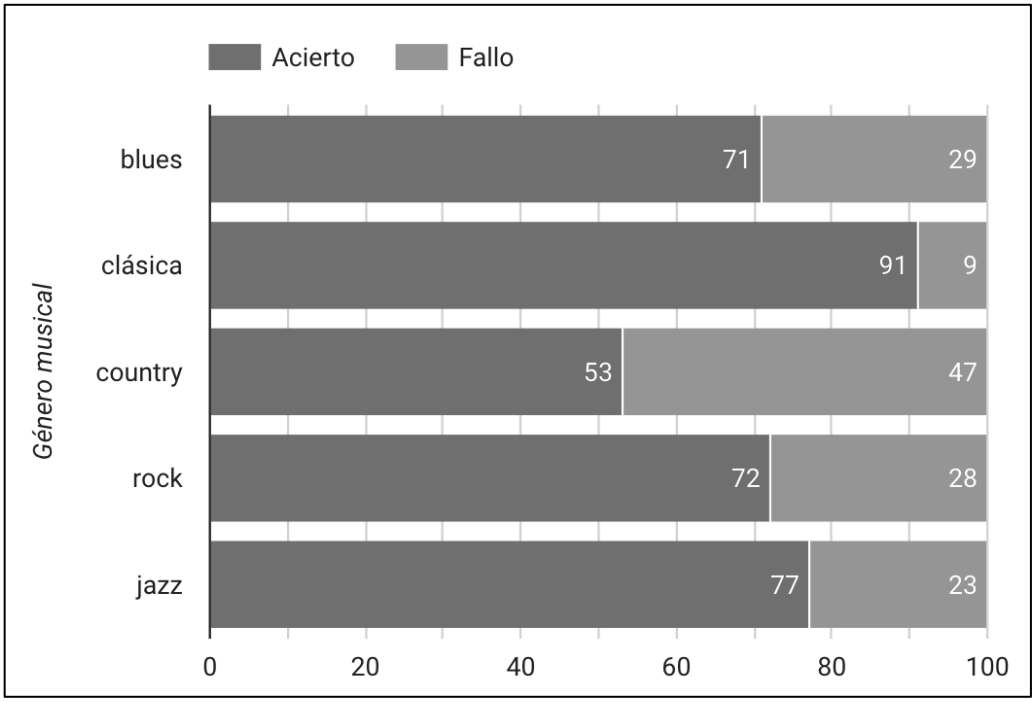
Figura 24. **Precisión de la CNN general**



Fuente: elaboración propia, realizado con Looker Studio.

La cantidad de aciertos por género musical es: 71 % para el género *blues*, 91 % para el género música clásica, 53 % para el género *country*, 72 % para el género *rock*, y 77 % para el género *jazz*. El resultado relativamente bajo para el género *country* es de esperarse, porque hay mucho parecido a nivel musical entre el *country* y el *rock,* teniendo múltiples escalas, acordes y ritmo compartido entre ellos.

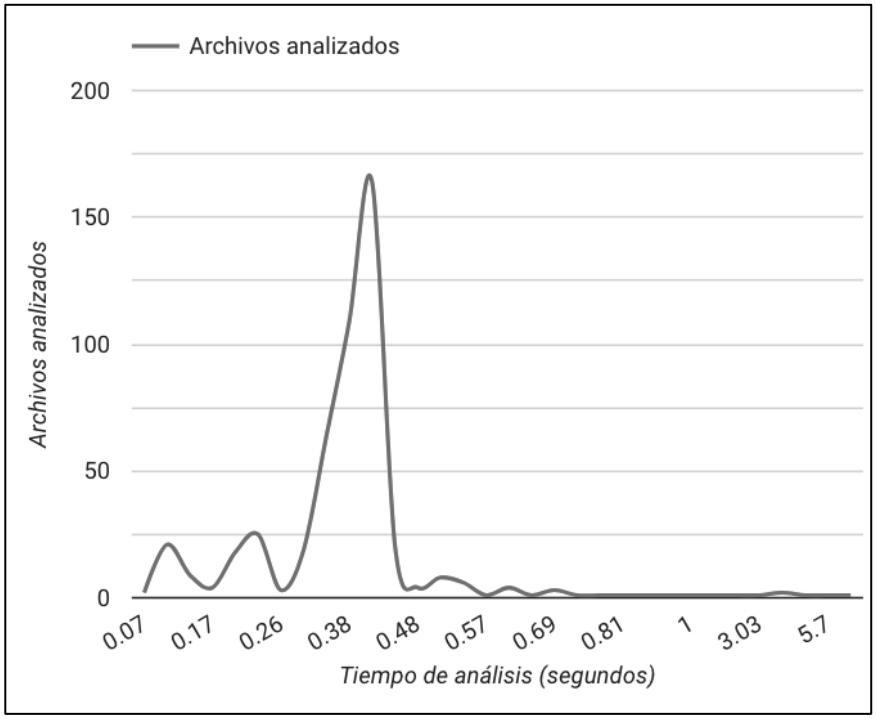
Figura 25. **Precisión de la CNN por género musical**



Fuente: elaboración propia, realizado con Looker Studio.

En relación con la velocidad de análisis, la media de tiempo para predecir es de 0.42 segundos, con una desviación estándar de 0.38 segundos, un mínimo de 0.07 segundos y un máximo de 9.58 segundos.

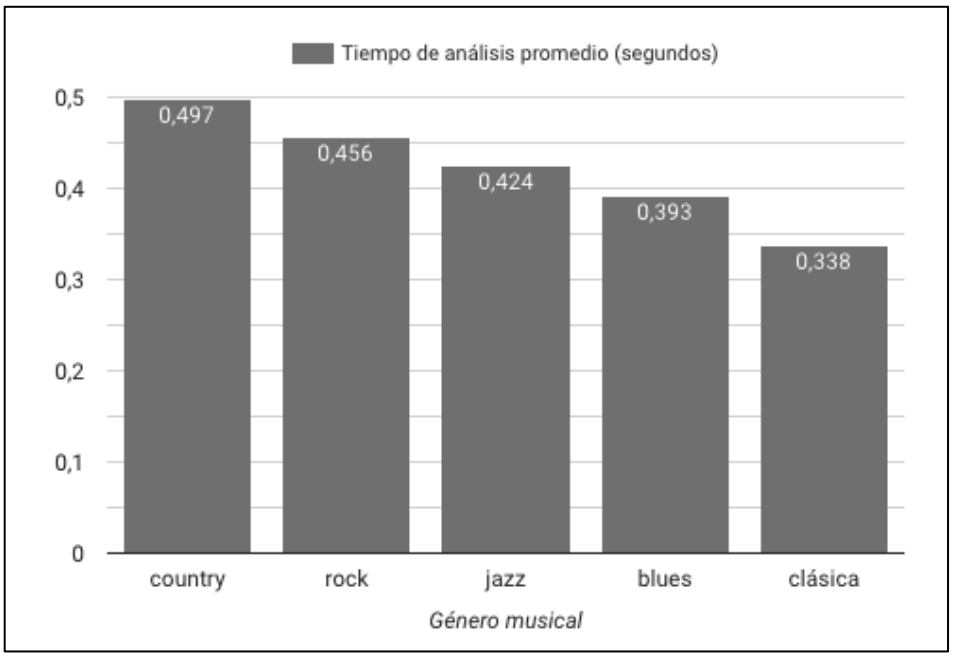
Figura 26. **Tiempo de análisis de la CNN general**



Fuente: elaboración propia, realizado con Looker Studio.

Por último, el tiempo de análisis promedio por género es: 0.393 segundos para *blues*, 0.338 segundos para música clásica, 0.497 para *country*, 0.456 para *rock* y 0.424 para *jazz*. La variación de tiempo de análisis para cada género musical resultó no ser significativo.

Figura 27. **Tiempo de análisis de la CNN por género musical**

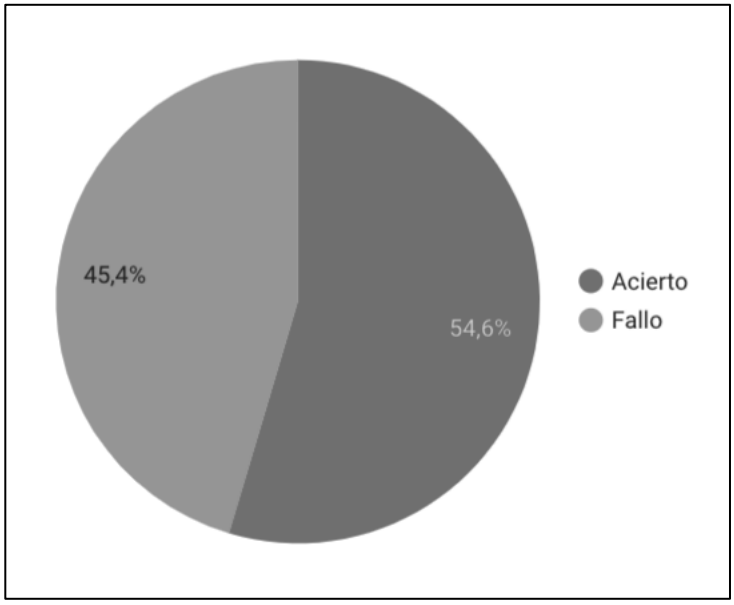


Fuente: elaboración propia, realizado con Looker Studio.

7.8.2. Resultados de la MLP

La MLP tuvo un porcentaje de acierto del 54.6 % de las veces, con 273 aciertos de 500 análisis.

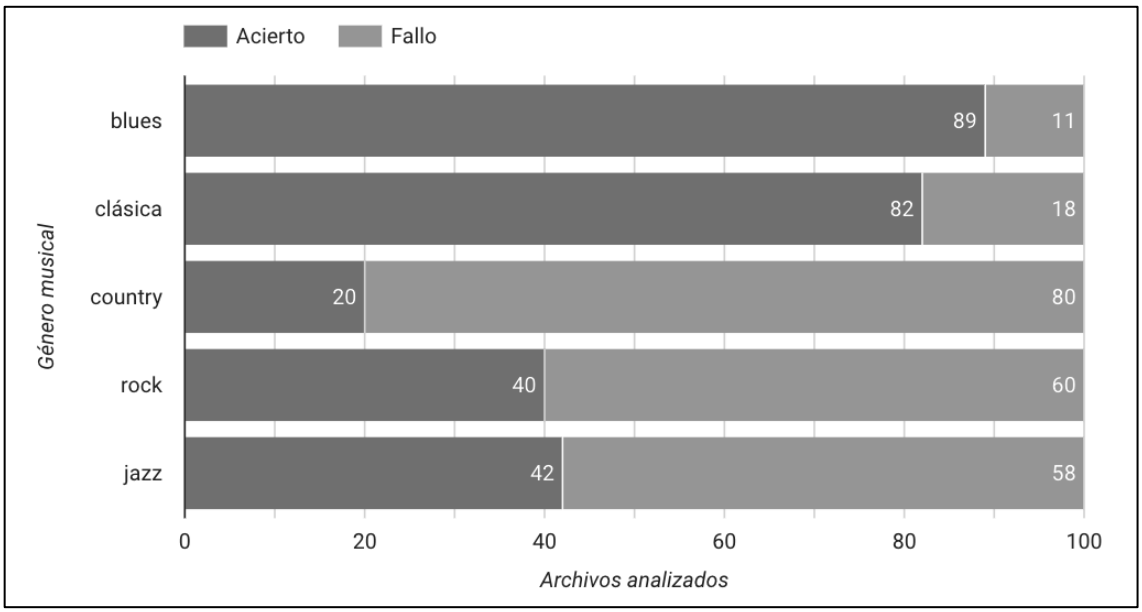
Figura 28. **Precisión de la MLP general**



Fuente: elaboración propia, realizado con Looker Studio.

La cantidad de aciertos por género musical es: 89 % para el género *blues*, 82 % para el género música clásica, 20 % para el género *country*, 40 % para el género *rock*, y 42 % para el género *jazz*. Se puede notar que la red neuronal tuvo mucho conflicto encontrando diferencias entre géneros musicales parecidos, es decir *blues*, *rock*, *country*; y mucha facilidad diferenciando entre géneros diferentes como música clásica.

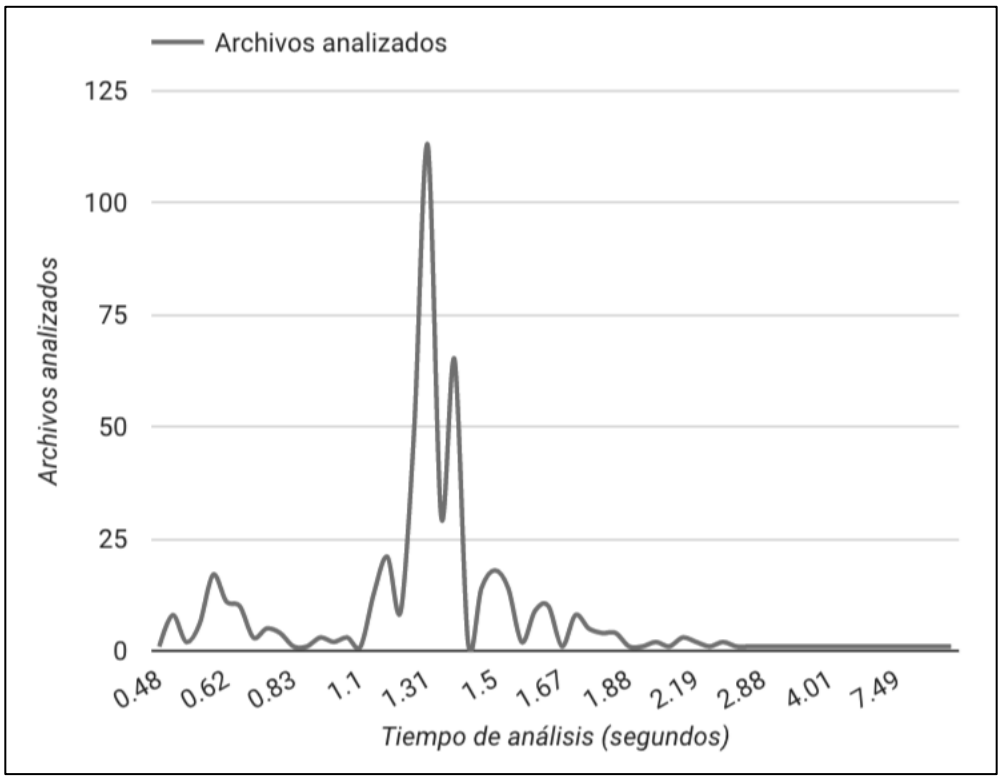
Figura 29. **Precisión de la MLP por género musical**



Fuente: elaboración propia, realizado con Looker Studio.

En relación con la velocidad de análisis, la media de tiempo para predecir es de 1.61 segundos, con una desviación estándar de 1.31 segundos, un mínimo de 0.48 segundos y un máximo de 9.12 segundos.

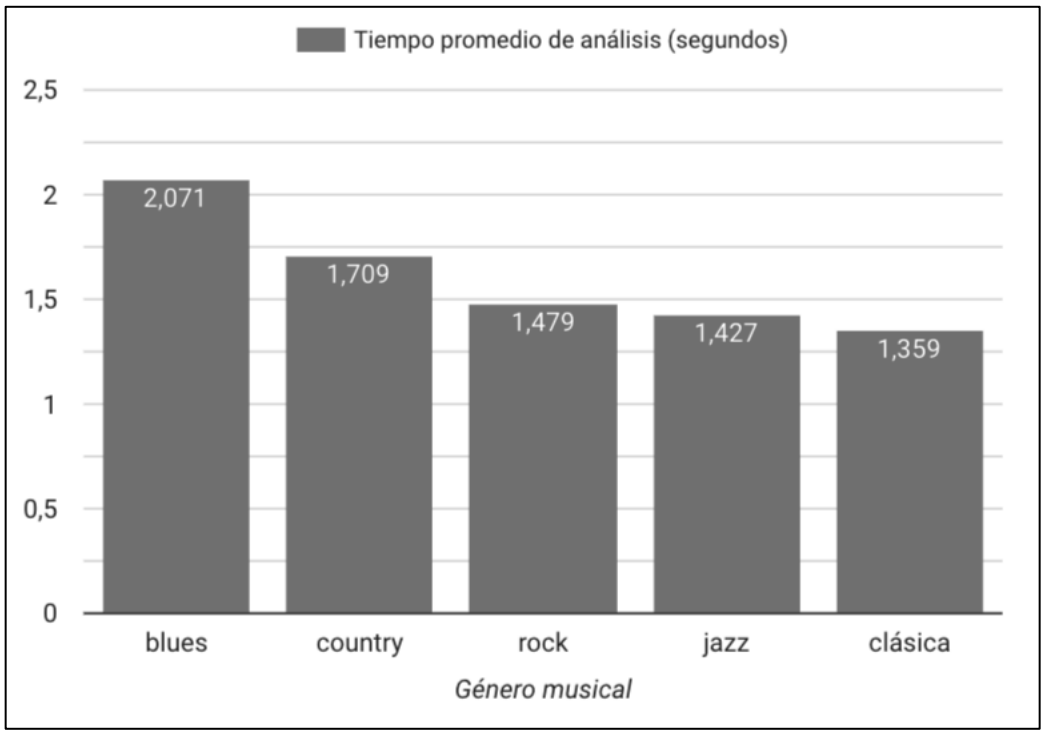
Figura 30. **Tiempo de análisis de la MLP general**



Fuente: elaboración propia, realizado con Looker Studio.

Por último, el tiempo de análisis promedio por género es: 2.071 segundos para *blues*, 1.359 segundos para música clásica, 1.709 para *country*, 1.479 para *rock* y 1.427 para *jazz*.

Figura 31. **Tiempo de análisis de la MLP por género musical**



Fuente: elaboración propia, realizado con Looker Studio.

CONCLUSIONES

1. Se construyó un sistema utilizando Python, Librosa, Keras y otras herramientas para implementar dos diferentes técnicas de Deep Learning y permitir a un usuario analizar un archivo digital de audio y clasificarlo en los géneros musicales: *blues*, música clásica, *country*, *rock* o *jazz*.

2. Se diseñó y entrenó una red neuronal convolucional implementando como entrada los coeficientes cepstrales en la frecuencia de Mel, como salida un vector de la probabilidad de ser clasificado en cada género musical, y siete capas: dos capas de convolución, dos capas de agrupamiento, dos capas densas, una capa de aplanamiento y una capa de pérdida.

3. Se diseñó y entrenó una red neuronal perceptrón multicapa implementando como entrada los coeficientes cepstrales en la frecuencia de Mel, como salida de un vector de probabilidad de ser clasificado en cada género musical, y cinco capas: tres capas densas, una capa de aplanamiento y una capa de agrupamiento.

4. La red neuronal convolucional consiguió un desempeño de 0.42 segundos de tiempo de predicción en promedio, y 72.8 % de precisión con 364 aciertos de 500 pruebas, mientras que la red neuronal perceptrón multicapa obtuvo un desempeño de 1.51 segundos de tiempo de predicción en promedio, y 54.6 % de precisión con 273 aciertos de 500 análisis.

5. Tras analizar el desempeño general de ambas redes neuronales, se concluye que la red neuronal convolucional es la más adecuada para realizar una tarea de clasificación de audio por género musical, tanto por su desempeño en el tiempo de predicción, como por la precisión de su predicción.

RECOMENDACIONES

1. Clasificar a los géneros musicales seleccionados de manera precisa resultó complicado, debido a los múltiples aspectos en común relacionados a su ritmo, escalas, melodía y acordes utilizados. Sería más factible considerar otros factores para su clasificación, o elegir géneros musicales con características más pronunciadas y diferentes entre ellas.

2. Optimizar la precisión de clasificación y considerar la utilización de otros parámetros musicales en conjunto con los utilizados para mejorar la búsqueda de diferencias significativas entre cada género propuesto.

3. Expandir la cantidad de géneros musicales a clasificar y tener en cuenta más datos para un mejor entrenamiento de las redes neuronales.

4. Aplicar la funcionalidad de clasificar archivos digitales utilizando redes neuronales puede expandirse a un sistema de indexación y categorización de datos automático, aplicaciones educativas, reconocimiento de sentimientos por la voz, entre otros.

REFERENCIAS

1. Aguilar, R. (2020). *Análisis de espectrogramas de señales EEG* (tesis de doctorado). Universidad Autónoma de Puebla, México. Recuperado de https://repositorioinstitucional.buap.mx/bitstream/handle/20.500.12371/10364/20201117125934-2112-T.pdf.

2. Alba, A. (s.f.). *Teorìa musical.* Santiago de Chile: Valparaiso. Recuperado de http://www.bibliotecanacionaldigital.gob.cl/colecciones/BND/00/MU/MU0019198.pdf.

3. Coursera (s.f.). Aprendizaje automático [Mensaje en un blog]. Recuperado de https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2#a86a.

4. Espino, L. (2016). *Inteligencia artificial.* Guatemala, Guatemala: Kindle.

5. Gollo, L. (2008). *Synchronization between populations of neurons* (tesis de maestria). Universidad de las Islas Baleares, España. Recuperado de https://digital.csic.es/bitstream/10261/18660/1/tesina.pdf.

6. Haykin, S. (2009). *Neural networks and learning machines.* Canadá: Pearson.

7. Ramos, E. (2020). *Deep Learning para la visión artificial e identificación del personal administrativo y docente de la Universidad Nacional Micaela Bastidas de Apurímac 2018.* Perú: Universidad Nacional del Altiplano.

8. Roberts, L. (5 de marzo, 2020). Understanding the Mel Spectrogram [Mensaje en un blog]. Recuperado de https://medium.com/analytics-vidhya/understanding-the-mel-spectrogram-fca2afa2ce53.

9. Rodriguez, M. (2021). *Detección automática de géneros musicales* (tesis de licenciatura). Universidad Politécnica de Madrid, España. Recuperado de https://oa.upm.es/66592/1/TFG_MARIA_LUCAS_RODRIGUEZ.pdf.

10. Russell, S., Norvig, P., Corchado, J. y Joyanes, L. (2011). *Inteligencia artificial un enfoque moderno.* Madrid, España: Pearson.

11. Sabra, A. (16 de febrero, 2021). *Learning from audio: the Mel Scale, Mel Spectrograms, and Mel Frequency Cepstral Coefficients* [Mensaje en un blog]. Recuperado de https://towardsdatascience.com/learning-from-audio-the-mel-scale-mel-spectrograms-and-mel-frequency-cepstral-coefficients-f5752b6324a8.

12. Sierra, M. (2011). *Sistema caracterizador de equipos de audio* (tesis de licenciatura). Universidad Nacional Autónoma de México, México. Recuperado de http://www.ptolomeo.unam.mx:8080/xmlui/handle/132.248.52.100/276.

13. Valiente, H. (2006). *Implementación de un analizador de espectro para frecuencias de audio utilizando un procesador digital de señales (DSP)* (tesis de licenciatura). Universidad de San Carlos de Guatemala, Guatemala. Recuperado de http://biblioteca.usac.edu.gt/tesis/08/08_0171_EO.pdf.

APÉNDICES

Apéndice 1. **Utilización del sistema clasificador**

Se lista la serie de pasos que un usuario debe realizar para analizar un archivo WAV o MP3 y obtener su clasificación utilizando el sistema realizado en el caso de estudio del séptimo capítulo del presente trabajo.

El primer paso es acceder a la aplicación, para ello se creó un archivo ejecutable para computadoras con sistema operativo Windows, disponible en el repositorio compartido en el apéndice 5. Para sistemas operativos basados en Linux será necesario clonar y compilar el código del proyecto con los pasos listados en el apéndice 2. Al ejecutar la aplicación se mostrará una ventana parecida a la figura A.

Continuación del apéndice 1.

Figura A. **Vista inicial del sistema clasificador**

Una vez ejecutado el sistema se presentará la vista inicial del mismo. En esta se presentan tres botones: un botón para utilizar la red neuronal convolucional titulado CNN, un botón para utilizar la red neuronal perceptrón multicapa titulado MLP y el último botón para visualizar una lista de reportes en de la creación y performance de cada red neuronal titulado Reportes, respectivamente.

Inicialmente el sistema presenta una foto vacía, únicamente para representar que no se ha clasificado ningún archivo todavía. También se presenta una serie de pestañas por las que el usuario puede navegar que se detallarán a continuación.

Al presionar el botón CNN, por ejemplo, se abrirá un diálogo de selección de un archivo, es posible abrir únicamente archivos MP3 o WAV. En este ejemplo se abrirá el archivo *prueba - blues.wav*, que contiene una canción de *blues.*

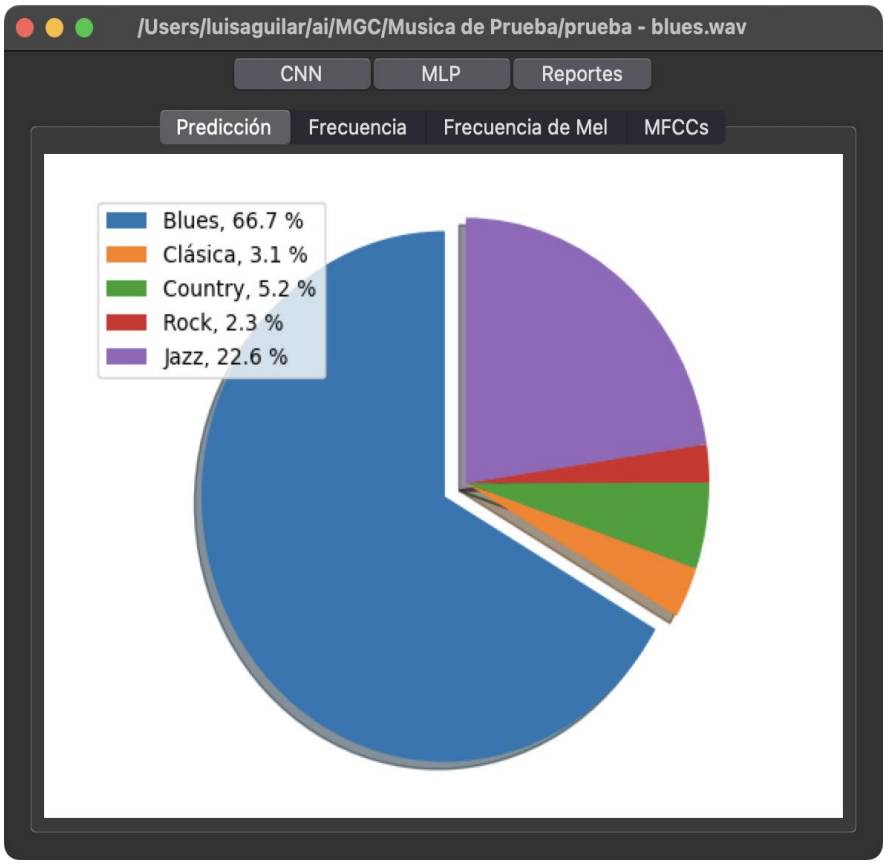
Figura B. **Diálogo de selección del archivo a clasificar**

Fuente: elaboración propia, realizado con captura de pantalla.

Continuación del apéndice 1.

El archivo será analizado automáticamente, y tras unos segundos de espera, se mostrará una gráfica de pie indicando el porcentaje de probabilidad calculado para cada género musical.

Figura C. **Vista de predicción**



Fuente: elaboración propia, realizado con captura de pantalla.

Continuación del apéndice 1.

También es posible explorar el espectrograma de frecuencias del archivo analizado utilizando la pestaña Frecuencia, el espectrograma de frecuencia de Mel utilizando la pestaña Frecuencia de Mel, y el diagrama de los coeficientes cepstrales en la frecuencia de Mel utilizando la pestaña MFCCs.

Estos diagramas no fueron compartidos en este apéndice porque son parte de las figuras contenidas en el caso de estudio del séptimo capítulo de este trabajo.

Fuente: elaboración propia, realizado con Word y captura de pantalla.

Apéndice 2. **Comandos de instalación, ejecución y entrenamiento del sistema clasificador**

```
# *** Asumiendo que ya se tiene instalado python 3.6 y pip ***

# --- Instalar y ejecutar el sistema ---
$ pip install virtualenv

# 1. Crear un entorno virtual
$ virtualenv -p python3.6 venv

# 2. Activar el entorno virtual
$ source venv/bin/activate

# 3.1 Instalar las dependencias para Linux
$ pip install -r requirements.txt

# 3.2 Instalar las dependencias para Windows
$ pip install -r requirements_windows.txt

# 4. Ejecutar el programa
$ python main.py

# --- Limpiar la base de datos original (Opcional) ---
$ python limpieza_datos.py

# --- Entrenar a los modelos (Opcional) ---

# 1. Entrenar el modelo de la red neuronal convolucional
$ python cnn.py

# 2. Entrenar el modelo de la red neuronal perceptron multicapa
$ python mlp.py

# --- Ejecutar la prueba de confiabilidad y desempeño (Opcional) ---
$ python prueba_confiabilidad.py
```

Fuente: elaboración propia, realizado con Python.

Apéndice 3. **Código utilizado para realizar una prueba de desempeño y precisión a las redes neuronales utilizadas en el caso de estudio**

```python
# Función para iterar la base de datos y obtener los resultados de los 500 archivos
# Esta función es útil para comprobar el rendimiento de la red neuronal
# Esta función no es necesaria para el funcionamiento del programa
# Un ejemplo del archivo resultante se puede encontrar en el repositorio
def iterar_db():
  directorios = ["blues", "classical", "country", "rock", "jazz"]
  generos = ["blues", "classical", "country", "rock", "jazz"]
  resultados = []
  for directorio in directorios:  # Iterar todos los directorios de la base de datos
    actual = f"./db/{directorio}"

    for archivo in os.scandir(actual): # Escanear todos los archivos en cada directorio
      print(f'>> Analizando {archivo}')

      # Predecir con la red neuronal perceptron multicapa
      vector_probabilidad_mlp = predecir(archivo, opcion="mlp")[0]
      # Predecir con la red neuronal convolucional
      vector_probabilidad_cnn = predecir(archivo, opcion="cnn")[0]

      tiempo_inicio_mlp = time.time() # Tiempo de inicio de la predicción de la red neuronal perceptron multicapa
      # Obtener el índice del valor más alto (género más probable)
      indice_prediccion_mlp = np.argmax(vector_probabilidad_mlp)
      tiempo_fin_mlp = time.time() # Tiempo de fin de la predicción de la red neuronal perceptron multicapa

      # Obtener el índice del valor más alto (género más probable)
      indice_prediccion_cnn = np.argmax(vector_probabilidad_cnn)
      tiempo_fin_cnn = time.time() # Tiempo de fin de la predicción de la red neuronal convolucional

      # Obtener el género más probable calculado con la red neuronal perceptron multicapa
      prediccion_mlp = generos[indice_prediccion_mlp]
      # Obtener el género más probable calculado con la red neuronal convolucional
      prediccion_cnn = generos[indice_prediccion_cnn]

      indice_esperado = generos.index(directorio) # Obtener el índice del género esperado actual

      # Obtener el valor de la predicción del género esperado
      prediccion_mlp_esperado = vector_probabilidad_mlp[indice_esperado]
      # Obtener el valor de la predicción del género esperado
      prediccion_cnn_esperado = vector_probabilidad_cnn[indice_esperado]

      # Verificar si la predicción es correcta para la red neuronal perceptron multicapa
      estado_mlp = revisar_resultado(prediccion_mlp, directorio)
      # Verificar si la predicción es correcta para la red neuronal convolucional
      estado_cnn = revisar_resultado(prediccion_cnn, directorio)

```

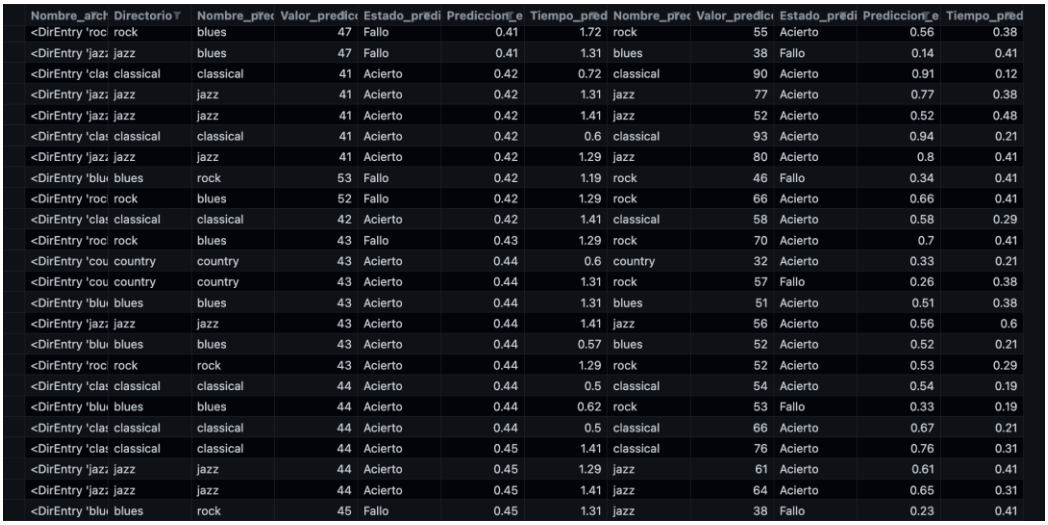
Continuación del apéndice 3.

```
        # Verificar si la predicción es correcta para la red neuronal perceptron multicapa
        estado_mlp = revisar_resultado(prediccion_mlp, directorio)
        # Verificar si la predicción es correcta para la red neuronal convolucional
        estado_cnn = revisar_resultado(prediccion_cnn, directorio)

        resultado = {
          # -- Datos generales --
          'nombre_archivo'          : archivo, # Nombre del archivo
          'directorio'              : directorio, # Nombre del directorio
          # -- Red neuronal perceptron multicapa --
          # Nombre del género más probable calculado con la red neuronal perceptron multicapa
          'nombre_prediccion_mlp'   : prediccion_mlp,
          # Valor del género más probable calculado con la red neuronal perceptron multicapa
          'valor_prediccion_mlp'    : int(vector_probabilidad_mlp[indice_prediccion_mlp] * 100),
          # Estado de la predicción, 'Acierto' si el resultado fue correcto, 'Fallo' si no
          'estado_prediccion_mlp'   : estado_mlp,
          # Valor de la predicción del género esperado
          'prediccion_esperada_mlp' : prediccion_mlp_esperado,
          # Tiempo de ejecución de la predicción de la red neuronal perceptron multicapa
          'tiempo_prediccion_mlp'   : round((tiempo_fin_mlp - tiempo_inicio_mlp) * 100000, 2),
          # -- Red neuronal convolucional --
          # Nombre del género más probable calculado con la red neuronal convolucional
          'nombre_prediccion_cnn'   : prediccion_cnn,
          # Valor del género más probable calculado con la red neuronal convolucional
          'valor_prediccion_cnn'    : int(vector_probabilidad_cnn[indice_prediccion_cnn] * 100),
          # Estado de la predicción, 'Acierto' si el resultado fue correcto, 'Fallo' si no
          'estado_prediccion_cnn'   : estado_cnn,
          # Valor de la predicción del género esperado
          'prediccion_esperada_cnn' : prediccion_cnn_esperado,
          # Tiempo de ejecución de la predicción de la red neuronal convolucional
          'tiempo_prediccion_cnn'   : round((tiempo_fin_cnn - tiempo_fin_mlp) * 100000, 2)
        }
        resultados.append(resultado) # Agregar el resultado a la lista de resultados

    df = pd.DataFrame(resultados) # Crear un dataframe con los resultados
    df.to_csv('resultados.csv', index=False) # Guardar los resultados en un archivo csv

iterar_db() # Ejecutar la prueba
```

Al ejecutar el código anterior, se obtiene un archivo de valores separados por comas en el cual se encuentra todos resultados de los 500 archivos analizados, como es posible apreciar en la figura D.

Continuación del apéndice 3.

Figura D. **Primeras filas del archivo resultante del análisis ejecutado**

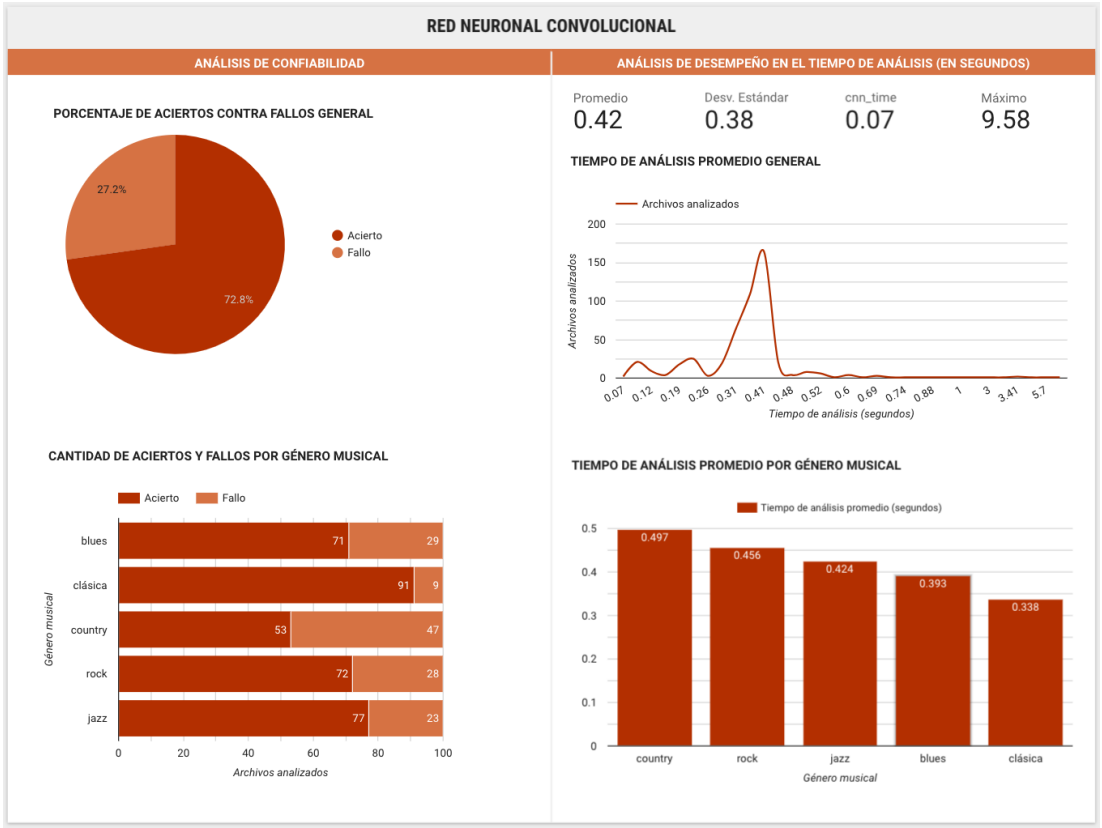


Nombre_arch	Directorio	Nombre_pred	Valor_predic	Estado_predi	Prediccion_e	Tiempo_pred	Nombre_pred	Valor_predic	Estado_predi	Prediccion_e	Tiempo_pred
<DirEntry 'roc	rock	blues	47	Fallo	0.41	1.72	rock	55	Acierto	0.56	0.38
<DirEntry 'jaz	jazz	blues	47	Fallo	0.41	1.31	blues	38	Fallo	0.14	0.41
<DirEntry 'cla	classical	classical	41	Acierto	0.42	0.72	classical	90	Acierto	0.91	0.12
<DirEntry 'jaz	jazz	jazz	41	Acierto	0.42	1.31	jazz	77	Acierto	0.77	0.38
<DirEntry 'jaz	jazz	jazz	41	Acierto	0.42	1.41	jazz	52	Acierto	0.52	0.48
<DirEntry 'cla	classical	classical	41	Acierto	0.42	0.6	classical	93	Acierto	0.94	0.21
<DirEntry 'jaz	jazz	jazz	41	Acierto	0.42	1.29	jazz	80	Acierto	0.8	0.41
<DirEntry 'blu	blues	rock	53	Fallo	0.42	1.19	rock	46	Fallo	0.34	0.41
<DirEntry 'roc	rock	blues	52	Fallo	0.42	1.29	rock	66	Acierto	0.66	0.41
<DirEntry 'cla	classical	classical	42	Acierto	0.42	1.41	classical	58	Acierto	0.58	0.29
<DirEntry 'roc	rock	blues	43	Fallo	0.43	1.29	rock	70	Acierto	0.7	0.41
<DirEntry 'cou	country	country	43	Acierto	0.44	0.6	country	32	Acierto	0.33	0.21
<DirEntry 'cou	country	country	43	Acierto	0.44	1.31	rock	57	Fallo	0.26	0.38
<DirEntry 'blu	blues	blues	43	Acierto	0.44	1.31	blues	51	Acierto	0.51	0.38
<DirEntry 'jaz	jazz	jazz	43	Acierto	0.44	1.41	jazz	56	Acierto	0.56	0.6
<DirEntry 'blu	blues	blues	43	Acierto	0.44	0.57	blues	52	Acierto	0.52	0.21
<DirEntry 'roc	rock	rock	43	Acierto	0.44	1.29	rock	52	Acierto	0.53	0.29
<DirEntry 'cla	classical	classical	44	Acierto	0.44	0.5	classical	54	Acierto	0.54	0.19
<DirEntry 'blu	blues	blues	44	Acierto	0.44	0.62	rock	53	Fallo	0.33	0.19
<DirEntry 'cla	classical	classical	44	Acierto	0.44	0.5	classical	66	Acierto	0.67	0.21
<DirEntry 'cla	classical	classical	44	Acierto	0.45	1.41	classical	76	Acierto	0.76	0.31
<DirEntry 'jaz	jazz	jazz	44	Acierto	0.45	1.29	jazz	61	Acierto	0.61	0.41
<DirEntry 'jaz	jazz	jazz	44	Acierto	0.45	1.41	jazz	64	Acierto	0.65	0.31
<DirEntry 'blu	blues	rock	45	Fallo	0.45	1.31	jazz	38	Fallo	0.23	0.41

Los datos resultantes fueron utilizados para crear un tablero de visualización de resultados, utilizando la herramienta Looker Studio de Google.

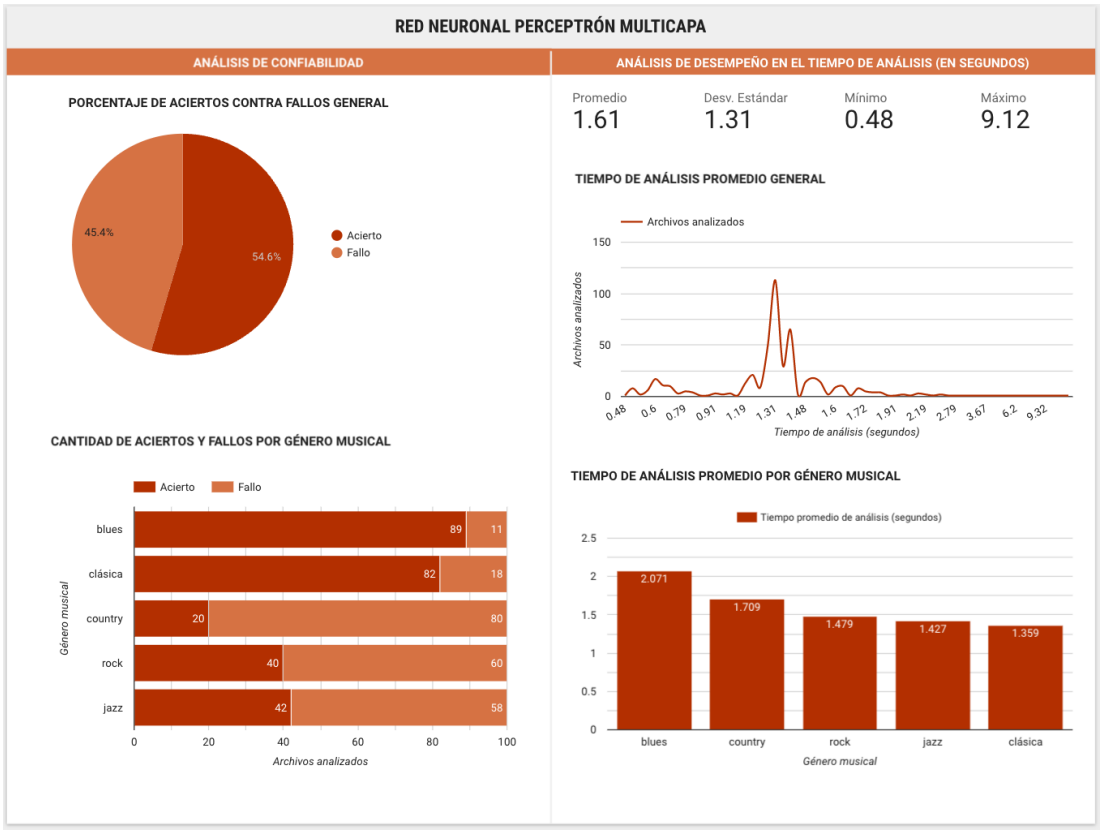
Continuación del apéndice 3.

Figura E. **Tablero de resultados de la CNN**



Continuación del apéndice 3.

Figura F. **Tablero de resultados de la MLP**

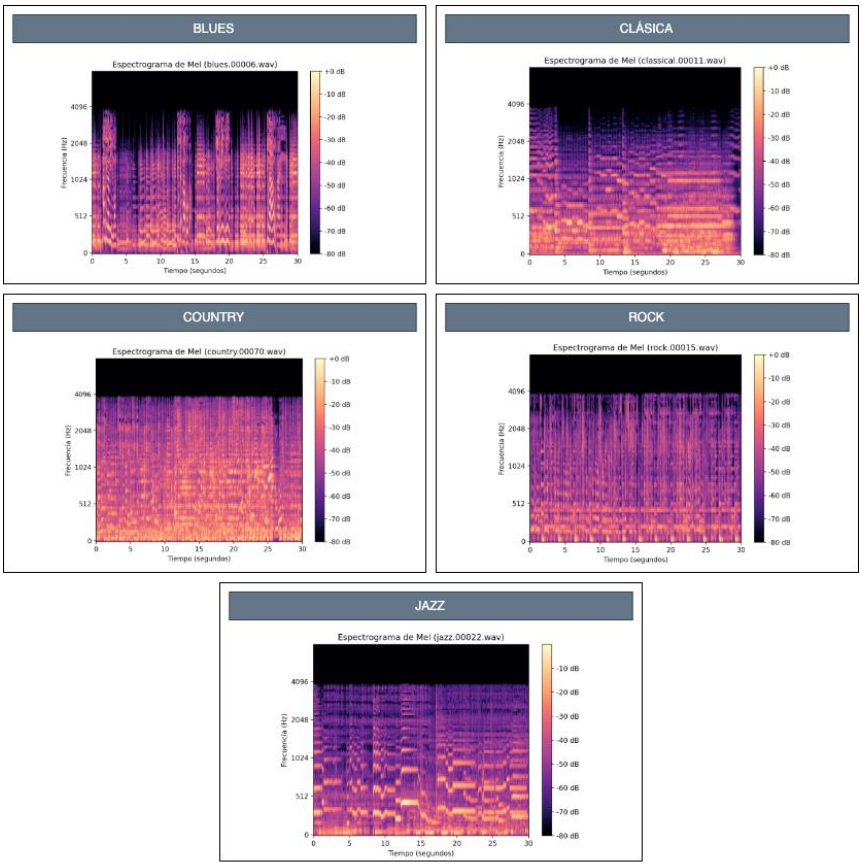


Fuente: elaboración propia, realizado con Looker Studio.

Apéndice 4. **Comparación entre los espectrogramas de frecuencia de Mel de cada género musical**

Se presenta un ejemplo de un espectrograma de Frecuencias de Mel por cada género musical considerado en el caso de estudio del séptimo capítulo. Es posible observar las diferencias claras entre géneros muy distintos, como la música clásica y el *blues*, y el parecido tan claro entre géneros musicales con características compartidas, como el *country* y el *rock*.

Figura G. **Espectrogramas de frecuencia de Mel por género musical**



Fuente: elaboración propia, realizado con draw.io.

Apéndice 5. **Código utilizado para preparar, crear y entrenar a las redes neuronales**

Se presentan las secciones de código más relevantes que fueron utilizadas en la creación de las redes neuronales, la preparación de la información para entrenarlas, y la ejecución de estas. El código completo estará siempre disponible en la página *GitHub*, siguiendo el enlace https://github.com/leoaguilar97/MGC.

Figura H. **Conversión de archivos de formato AU a WAV**

```python
from pydub import AudioSegment # pip install pydub
import os

cwd = os.getcwd()

# 1. Definir los directorios por buscar directorios
genres = ["blues", "classical", "country", "jazz", "rock"]

# 2. Definir el formato de los archivos de entrada y salida
# Se busca transformar los archivos de formato .au a .wav
input_format = "./db/{genre}/{genre}.{file_number}.au" # por ejemplo: ./db/blues/blues.00000.au
output_format = "./db/{genre}/{genre}.{file_number}.wav" # por ejemplo: ./db/blues/blues.00000.wav

i = 0
for genre in genres:
    for i in range(100):
        # 3. Definir el nombre de los archivos de entrada y salida
        file_number = (str(i)).zfill(5) # El número de archivo debe tener 5 dígitos

        # 4. Definir el nombre de los archivos de entrada y salida
        file_name = input_format.format(genre=genre, file_number=file_number)
        output_name = output_format.format(genre=genre, file_number=file_number)

        print(f"{file_name} > {output_name}")

        # 5. Convertir el archivo de entrada a formato wav y guardarlo en el archivo de salida
        # para que sea compatible con librosa
        au_audio = AudioSegment.from_file(file_name, format="au")
        au_audio.export(output_name, format="wav")
```

Continuación del apéndice 5.

Figura I. **Definición de los datos de entrenamiento y de prueba**

```
mfccs, etiquetas = obtener_datos_de_entrenamiento()
mfccs_entrenamiento, mfccs_prueba, etiquetas_entrenamiento, etiquetas_prueba = train_test_split(
  mfccs,                # Datos obtenidos de los 500 archivos de audio
  etiquetas,            # Blues, Classical, Country, Rock, Jazz
  random_state=100,
  stratify=etiquetas,   # Definir cómo están etiquetados los datos
  test_size=0.2         # 20% de los datos se utilizarán para pruebas
)

mfccs_entrenamiento /= mfccs_entrenamiento.min()  # Normalizar los datos de entrenamiento
mfccs_prueba /= mfccs_entrenamiento.min()         # Normalizar los datos de prueba

# Convertir a un arreglo de 128x657 con un solo canal de color
mfccs_entrenamiento = mfccs_entrenamiento.reshape(mfccs_entrenamiento.shape[0], 128, 657, 1)
mfccs_prueba = mfccs_prueba.reshape(mfccs_prueba.shape[0], 128, 657, 1)

# 5 generos en labels
etiquetas_entrenamiento = to_categorical(etiquetas_entrenamiento, 5) # Convertir a 5 categorias (géneros musicales)
etiquetas_prueba = to_categorical(etiquetas_prueba, 5)               # Convertir a 5 categorias (géneros musicales)
```

Figura J. **Obtención de datos de las canciones de prueba**

```
def obtener_informacion_de_las_canciones():
    labels = []
    mel_specs = []
    directorios = [
      "blues", "classical", "country", "rock", "jazz"
    ]
    label_dict = { "blues": 0, "classical": 1, "country": 2, "rock": 3, "jazz": 4 }

    for directorio in directorios:                    # Recorre todos los directorios a procesar
        actual = f"./db/{directorio}"
        for file in os.scandir(actual):               # Recorre todos los archivos de cada directorio
            labels.append(directorio)                 # Agrega el nombre del directorio a la lista de etiquetas
            S_db = obtener_mfccs_de_un_archivo(file)  # Obtiene los MFCCs de cada archivo
            mel_specs.append(S_db)                    # Agrega los MFCCs a la lista de MFCCs

    X = np.array(mel_specs)
    labels = pd.Series(labels)
    y = labels.map(label_dict).values

    # "X" es un arreglo que contiene los MFCCs de cada archivo
    # "y" es un arreglo que contiene las etiquetas de cada archivo, pero en formato numérico
    return X, y
```

Fuente: elaboración propia, realizado con Python

Continuación del apéndice 5.

Figura K. **Definición, compilación y entrenamiento de una CNN**

```
# Agregar la primera capa, una convolucional
# 16 filtros, kernel de 3x3, activacion relu y el input shape definido anteriormente
red_neuronal.add(
  Conv2D(filters=16, kernel_size=3, activation="relu", input_shape=(128, 657, 1))
)

# Agregar una capa MaxPooling2D para obtener todos los mayores
# tamaño pool: 2x4
red_neuronal.add(MaxPooling2D(pool_size=(2, 4)))

# Agregar otra capa convolucional
# 32 filtros, kernel de 3x3 y activacion relu
red_neuronal.add(Conv2D(filters=32, kernel_size=3, activation="relu"))

# De nuevo una capa MaxPooling2D
# tamaño pool: 2x4
red_neuronal.add(MaxPooling2D(pool_size=(2, 4)))

# Agregar una capa de aplanamiento
red_neuronal.add(Flatten())

# Agergar una capa Dense de 64 neuronas, con activacion relu
red_neuronal.add(Dense(64, activation="relu"))

# Ignorar 25% de los nodos resultantes para prevenir overfitting
red_neuronal.add(Dropout(0.25))

# Capa final para obtener los porcentajes de cada género
red_neuronal.add(Dense(5, activation="softmax"))

# Compilar la red neuronal
red_neuronal.compile(
  loss="categorical_crossentropy",
  optimizer="adam",
  metrics=["accuracy"]
)

# Entrenar el modelo utilizando la informacion de prueba generada anteriormente
history = red_neuronal.fit(
  entr_espect,
  entr_etiquetas,
  batch_size=16,
  validation_data=(prueba_espect, prueba_etiquetas),
  epochs=15,
)

```

Continuación del apéndice 5.

Figura L **Definición, compilación y entrenamiento de una red MLP**

```
# Crear un modelo secuencial
red_neuronal = Sequential()

# Capa de entrada
# 128 neuronas, 657 entradas, 1 canal, activacion relu
red_neuronal.add(Dense(128, input_shape=(128, 657, 1), activation='relu'))

# Capa oculta
# 64 neuronas, activacion relu
red_neuronal.add(Dense(64, activation='relu'))

# Capa oculta
# Aplanar la salida de la capa anterior
red_neuronal.add(Flatten())

# Capa oculta
# Ignorar 25% de los nodos resultantes para prevenir overfitting
red_neuronal.add(Dropout(0.30))

# Capa de salida
# 5 neuronas, activacion softmax
# La salida de esta capa es un vector de 5 posiciones
# Cada posición representa la probabilidad de que la entrada sea de un género musical
red_neuronal.add(Dense(5, activation='softmax'))

# Compilar la red neuronal
red_neuronal.compile(
  loss='categorical_crossentropy',
  optimizer='adam',
  metrics=['accuracy']
)

# Entrenar el modelo utilizando la informacion de prueba generada anteriormente
history = red_neuronal.fit(
  entr_espect,
  entr_etiquetas,
  batch_size=16,
  validation_data=(prueba_espect, prueba_etiquetas),
  epochs=10
)
```

Fuente: elaboración propia, realizado con Python y Keras.