Instituto Tecnológico
y de Estudios Superiores de Occidente

Reconocimiento de validez oficial de estudios de nivel superior según acuerdo secretarial 15018, publicado en el Diario Oficial de la Federación del 29 de noviembre de 1976.

Departmento de Matemáticas y Física

Maestría en Ciencia de Datos

Procesamiento de Audio Digital para la Clasificación de Sonidos Urbanos a través de una Red Neuronal

---

TESIS para obtener el **GRADO** de
MAESTRÍA EN CIENCIA DE DATOS

Tesis presentada por: **Ramón López Escudero**

Director de Tesis: **Dr. Fernando I. Becerra López**

Tlaquepaque, Jalisco, Mayo, 2023

Instituto Tecnológico y de Estudios Superiores de Occidente

Reconocimiento de validez oficial de estudios de nivel superior según acuerdo secretarial 15018, publicado en el Diario Oficial de la Federación del 29 de noviembre de 1976.

Departmento de Matemáticas y Física

Formato de aprobación de la Maestría en Ciencia de Datos

Título de la tesis:

Procesamiento de Audio Digital para la Clasificación de Sonidos Urbanos a través de una Red Neuronal

Autor:

Ramón López Escudero

Tesis aprobada para completar los requisitos de grado
para la Maestría en Ciencia de Datos.

Director de Tesis, **Dr. Fernando I. Becerra López**

Revisor de Tesis, **Dr. Riemann Ruiz Cruz**

Revisor de Tesis, **Dr. Saúl Alonso Nuño Sánchez**

Asesor Académico, **Dra. Rocío Carrasco Navarro**

Tlaquepaque, Jalisco, Mayo, 2023

Dedicatoria

En esta sección no sólo quiero dedicar mi trabajo a todos los seres que me dieron energía y motivación para culminar mis estudios, también quiero expresar mi eterno agradecimiento a la vida al darme la oportunidad de poder cumplir este objetivo personal. Han sido épocas de cambios y aprendizajes por lo que lograr estabilidad se vuelve un tema complejo. Sin embargo, se han abierto múltiples puertas que me han ayudado a encontrarme y plantear nuevos ideales, permitiéndome seguir adelante.

Primero quiero agradecerle al ITESO por haberme beneficiado con la beca de excelencia académica, sin ello me hubiese sido imposible cubrir los gastos necesarios para continuar con mis estudios (no podía creerlo cuando recibí la noticia). A pesar de haber iniciado el ciclo en 2020, el ITESO siempre contó con la infraestructura necesaria para abordar la problemática y brindarnos todas las herramientas necesarias. Esto me pareció formidable de principio a fin, ya que gracias a ello pude enfocarme en lo que me pende como estudiante. El cuerpo académico del programa en ciencia de datos es excelente, sumando motivaciones a querer aprender más, cuestionar y expandir conocimientos. Estoy muy contento de haber seleccionado al ITESO como mi casa de estudios a nivel posgrado.

También quiero agradecerle todo el empeño, apertura y motivación a mi asesor de tesis, Fernando. Tomé un par de clases con él y su pasión por la computación y matemáticas me generó mucha felicidad, es bonito ser estudiante de maestros como él. Posteriormente, siendo mi asesor de tesis, siempre se mostró abierto a todas las ideas que proponía, además de ser empático cuando notaba emoción de mi lado, ese tipo de acciones me mantuvieron motivado de principio a fin.

Para lograr integridad es necesario rodearte de personas que te amen, respeten e induzcan una sinergia favorable y sana para el crecimiento y evolución personal. Estoy muy agradecido de haber encontrado a personas tan valiosas como lo son mis amigos. En todo momento se han hecho presentes y me han brindado todo el apoyo y amor necesario para sentirme pleno. Nada de lo que logrado habría sido posible sin ellos.

Curiosamente, poco después de haber iniciado la maestría, me reencontré con la persona con la que actualmente estoy construyendo una nueva etapa de mi vida. En ella he encontrado una excelente compañera de vida, sintiéndome apoyado en todas las decisiones que tomo, ayudándome en los malos momentos y celebrando conmigo los éxitos logrados. En estos años de posgrado hemos compartido muchas vivencias, evolucionado juntos y apoyándonos el uno al otro para seguir adelante. Gracias por todo Sofía. Cabe mencionar que actualmente tenemos dos perritos que llegaron a nuestras vidas y nos han llenado de felicidad y risas, todos esos momentos son invaluables y me permiten seguir con una sonrisa en todo momento.

Dejé al final a las dos personas más importantes en mi vida: mis padres. Creo que es complicado poder entender todo lo que implica ser padre si aún no lo experimentas, pero se puede observar y sentir todo la energía, tiempo y espacio que implica serlo. Es una decisión que representa el amor y respeto hacia la vida. No tengo palabras para demostrar todo lo que los amo, siempre han estado cuidando mi espalda, siendo presentes en todo momento, escuchándome y aconsejándome en cada paso que doy. Sé que nos quedan muchos años por delante para seguirnos disfrutando. Gracias por todo.

Abuelita, donde quiera que estés, siempre te llevo en el ama, gracias por tanto amor y sabiduría.

Procesamiento de Audio Digital para la Clasificación de Sonidos Urbanos a través de una Red Neuronal

Ramón López Escudero

Resumen

En respuesta directa al crecimiento poblacional en las grandes urbes, las ciudades inteligentes se han impulsado como un catalizador tecnológico, el cual permite aprovechar la disponibilidad de arquitecturas sensoriales distribuidas a lo largo de la urbe. Esto promueve nuevas áreas de estudio, siendo una de ellas la clasificación de sonidos ambientales (ESC, por sus siglas en inglés).

Múltiples esfuerzos se han desarrollado para generar algoritmos computacionales que infieran de manera adecuada la relación entre los sonidos urbanos y su taxonomía, área poco profundizada en comparación a otras ramas de investigación. Sin embargo la técnica óptima para resolver este tipo de problemas no ha sido identificada aún, por lo que existen áreas de oportunidad en este campo de investigación como lo son la selección del método adecuado para trasladar el sonido a una representación numérica, prepreocesamiento del audio digital, selección del modelo predictivo a implementar, entre otras.

La investigación presentada en esta tesis ahonda diferentes tópicos, abordando desde el procesamiento de señales para trasladar el espacio auditivo a una representación numérica a través de los coeficientes cepstrales de Mel (MFCCs, por sus siglas en inglés), hasta la selección y ajuste de un modelo de inteligencia artificial que tiene como foco principal un método de clasificación robusto para inferir la taxonomía de los sonidos contenidos en una urbe. Esta investigación se realizó utilizando el conjunto de datos"UrbanSound8k", el cual integra diez categorías de sonidos (motores de autos, ruido a consecuencia de obras civiles, etcétera), además de ser punto de referencia en diversas investigaciones.

Palabras Clave

- Clasificación de sonidos ambientales
- Coeficientes cepstrales de Mel
- Data augmentation
- Red neuronal
- Regularización con dropout

Digital Audio Processing for Urban Sound Classification through a Neural Network

Ramón López Escudero

Abstract

In response to population growth in big metropolises, smart cities are considered a technological catalyst, allowing the utilization of distributed sensory architectures throughout cities, and promoting new areas of research, one of which is the environmental sound classification (ESC).

Several efforts have been developed to generate computational algorithms to infer the relationship between urban sounds and their taxonomy, an area of study that has been less explored compared to other research branches. However, the optimal technique for solving this type of problem has not yet been identified, so there are many areas of opportunity in this field.

This work delves into various topics, ranging from signal processing to translate the auditory space into a numerical representation through Mel Frequency Cepstral Coefficients (MFCCs) to the selection and adjustment of an artificial intelligence model, having as its main focus the development of a robust classification method to infer the taxonomy of sounds contained within an urban environment. As the source of study the ”UrbanSound8k” dataset was utilized, which integrates ten different categories of sounds (car engines, construction noise, etc.), besides it’s a point of reference in various research studies.

Key Words

- Environmental sound classification
- Mel frequency cepstral coefficients
- Data augmentation
- Neural network
- Dropout regularization

Índice general

Índice de figuras

Página

Índice de cuadros

Página

Siglas

AdaGrad Gradiente adaptativo.

ADAM Estimación adaptativa del momentum.

AUC Área bajo la curva.

dB Decibelio.

DCT Transformada discreta del coseno.

DFT Transformada discreta de Fourier.

FFT Transformada rápida de Fourier.

Hz Hercio.

JAMS JSON con especificación de música anotada.

MFCCs Coeficientes cepstrales en la escala de Mel.

MUDA Aumento de datos musicales.

PSD Espectro de Potencia.

ReLu Unidad lineal rectificada.

RMSProp Propagación de la raíz cuadrada del valor medio de los cuadrados.

ROC Característica de operación del receptor.

SGD Gradiente descendente estocástico.

1 Introducción

En respuesta directa al crecimiento poblacional en las grandes urbes, las ciudades inteligentes se han impulsado como un catalizador tecnológico el cual permite aprovechar la disponibilidad de arquitecturas sensoriales distribuidas a lo largo de la urbe, promoviendo nuevas áreas de estudio, siendo una de ellas la clasificación de sonidos ambientales (ESC, por sus siglas en inglés) [1].

El procesamiento y clasificación computacional del sonido, técnicamente referido como acústica, ayuda a reconocer el ambiente, el habla y la música en un contexto urbano [2]. Entre los sonidos de ambiente, existen múltiples categorías de acústica, como personas hablando, disparos de armas de fuego, ladridos de perros, etcétera.

La caracterización de sonidos ambientales tiene una amplia gama de aplicaciones, como por ejemplo, sistemas de control de seguridad, sistemas de vigilancia de audio, detección de escenas del crimen mediante el uso de audio y video, etcétera [3].

Múltiples esfuerzos se han desarrollado para generar algoritmos computacionales los cuales clasifiquen de manera adecuada sonidos urbanos en diferentes momentos y espacios. Sin embargo, las limitaciones persisten ya que los sonidos ambientales suelen presentar baja calidad debido a que la fuente emisora del sonido no está colocada cerca de los sensores de sonido. Por lo tanto, una escena ambiental consiste en el traslape de múltiples sonidos, lo cual resulta en un problema para su correcta caracterización [4].

Muchos de los estudios previamente realizados respecto a la acústica urbana se han enfocado principalmente en caracterizar el **habla humana** y **música**, basando los primeros algoritmos de clasificación en características diseñadas manualmente (conjunto de datos usando información existente y preprocesándola para resaltar ciertos rasgos). Recientemente se han propuesto algoritmos de aprendizaje profundo, siendo las redes neuronales convolucionales la técnica más exitosa ya

[1] R. A. et al., "Sound classification and processing of urban environments: A systematic literature review," *Sensors*, vol. 22, no. 22, 2022

[2] A. Bansal and K. N., "Environmental sound classification: A descriptive review of the literature," *Intelligent Systems with Applications*, vol. 16, 2022

[3] F. H. Al-Hattab, Y. and A. Shafie, "Rethinking environmental sound classification using convolutional neural networks: optimized parameter tuning of single feature extraction," *Neural Computing and Applications volume*, vol. 33, 2021

[4] A. Bansal and K. N., "Environmental sound classification: A descriptive review of the literature," *Intelligent Systems with Applications*, vol. 16, 2022

que este tipo de arquitectura tiene la habilidad de aprender tanto aspectos esenciales del audio como aspectos genéricos (sonidos de fondo que den contexto sobre donde y como fue capturado la muestra de sonido).

En cuanto a la clasificación de **sonidos urbanos**, la mayoría de las soluciones propuestas se basan igualmente en redes neuronales convolucionales o redes neuronales recurrentes. Sin embargo, la técnica óptima para resolver este tipo de problemas no ha sido definida aún, por lo que existen muchas áreas de oportunidad en este campo de investigación. Además, la caracterización del sonido urbano no se ha implementado en escenarios urbanos reales, teniendo así un futuro brillante para el aprendizaje profundo aplicado a los sistemas de sonido urbano, con un enorme potencial para complementar otras formas de detección automatizada, como cámaras de video, etcétera [5].

[5] R. A. et al., "Sound classification and processing of urban environments: A systematic literature review," *Sensors*, vol. 22, no. 22, 2022

Este trabajo tiene como foco principal investigar de manera precisa la relación de los sonidos urbanos y su taxonomía, área de investigación poco profundizada en comparación a otras ramas de investigación como lo es el procesamiento de lenguaje natural, música, bioacústica, etcétera. Subsecuente a lo antes mencionado, se ahondó en técnicas de procesamiento digital de señales, buscando trasladar el espacio auditivo a una representación numérica a través de los coeficientes cepstrales de Mel, permitiendo así el análisis y clasificación de audios en contextos urbanos.

Se hizo uso del conjunto de datos **"UrbanSound8k"** [6], el cual integra sonidos urbanos clasificados en diez categorías distintas (motores de autos, sonidos de construcciones, etcétera), además de haber sido objeto de estudio en previas investigaciones, en donde se presentan diversas técnicas de extracción de características y modelos de clasificación. Salamon y Bello (2017) [7] presentan un modelo de aprendizaje profundo (redes neuronales convolucionales) utilizando como fuente de conocimiento para el modelo los coeficientes cepstrales de Mel, obteniendo un resultado global de exactitud de 79 %. Das et al. (2020) [8] obtienen una exactitud global del 98.81 % empleando redes neuronales convolucionales y redes de memoria de corto-largo plazo, además de integrar técnicas procesamiento de audio como lo son los coeficientes cepstrales de Mel y el cromagrama, aunado al aumento de datos a través de la deformación de ciertas características de audio, como cambio de tonalidad, estrechamiento temporal, etcétera. Por último, un estudio realizado por Mushtaq y Su (2020) [9] es considerado actualmente como el estado del arte, ya que presentó excelentes resultados de exactitud no sólo utilizando el conjunto de datos "UrbanSound8k" (97.98 %.), sino también en otros conjuntos de datos que igualmente buscan la

[6] J. C. . B. J. P. Salamon, J., "A dataset and taxonomy for urban sound research," *ACM international conference on Multimedia*, vol. 22, pp. 1041–1044, 2014

[7] J. Salamon and J. P. Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," *CoRR*, vol. abs/1608.04363, 2016

[8] D. J. et al., "Environmental sound classification using convolution neural networks with different integrated loss functions," *Expert Syst*, vol. 39, 2021

[9] M. J. and S. Su, "Efficient classification of environmental sounds through multiple features aggregation and data enhancement techniques for spectrogram images," *Symmetry*, vol. 12, no. 1822, 2020

capacidad de clasificar espacios ambientales auditivos.

En el capítulo 2 se desglosan todos los conceptos previos al diseño e implementación del flujo de trabajo. Esto engloba múltiples ciencias aplicadas, como el estudio de aspectos físicos del audio, procesamiento digital de señales, optimización numérica y ciencias de la computación. En el capítulo 3 se describe la arquitectura del diseño implementado para la resolución del problema propuesto: clasificación de sonidos ambientales urbanos. También se profundiza en la selección de algoritmos y parámetros en cada paso propuesto en el diseño. Por último, en los capítulos 4 y 5 se presentan los resultados y conclusiones finales del trabajo.

2 Marco teórico

Como parte de un proceso cognitivo, es necesario estudiar las herramientas necesarias que nos ayudarán a abordar un tema de investigación. A través de este capítulo se plasmarán todos los conceptos necesarios para cumplir con el objetivo de esta tesis, abarcado desde el análisis y entendimiento del sonido en una perspectiva física-matemática, hasta toda la teoría crítica para implementar un modelo de inteligencia artificial, en específico, una red neuronal artificial. También se profundizará en la teoría de técnicas de procesamiento digital de señales, rama fundamental para trasladar el espacio auditivo a una representación numérica computacional.

2.1 Sonido

El sonido es la sensación producida por la detección de ondas acústicas en el oído. Este efecto es producido por vibraciones mecánicas en un medio, como puede ser el aire [1]. La cuerda de una guitarra es un ejemplo de un objeto vibrante, el cual, en movimiento, perturba las moléculas de aire en reposo causando variaciones periódicas (ondas) tal como se muestra en la Figura 2.1.

[1] J. Valenzuela, *Audio Digital: Conceptos Básicos y Aplicaciones*. Santa Monica, CA.: Backbeat Books, 1996

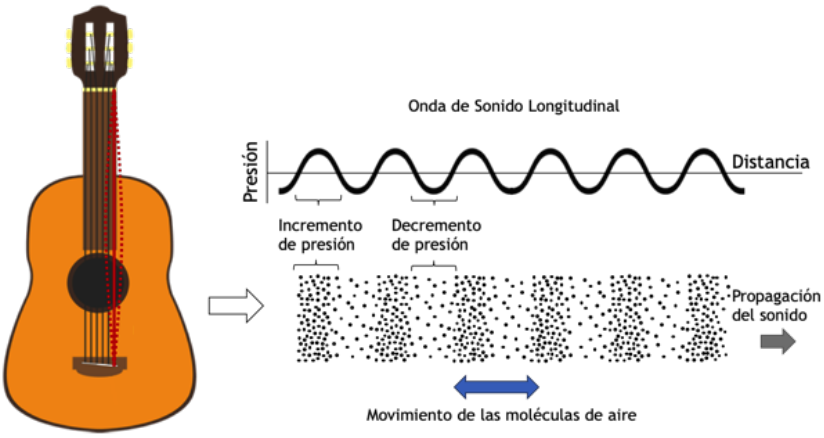


Figura 2.1: Vibraciones ocasionadas por el sonido de una guitarra
Fuente: Science Buddies

El sonido es propagado por moléculas de aire a través de desplazamientos sucesivos; dicho en otras palabras, moléculas de aire chocando unas con otras, propagando así la energía y alejándose gradualmente de la fuente transmisora (una voz, la cuerda de una guitarra, etcétera).

El sonido requiere de un medio elástico para su propagación. Si las moléculas de aire son desplazadas fuera de su posición original por una fuerza externa, estas tienden a regresar a su posición original cuando es removida la fuerza externa. El desplazamiento local de las moléculas de aire se desplazan en la dirección de las perturbaciones en las que están viajando, así que el sonido sigue una forma de transmisión longitudinal [2]. El receptor, por ejemplo, un micrófono en el mismo campo de sonido moverá su diafragma de acuerdo a la presión impuesta en él, completando la cadena de reacciones.

[2] J. Valenzuela, *Audio Digital: Conceptos Básicos y Aplicaciones*. Santa Monica, CA.: Backbeat Books, 1996

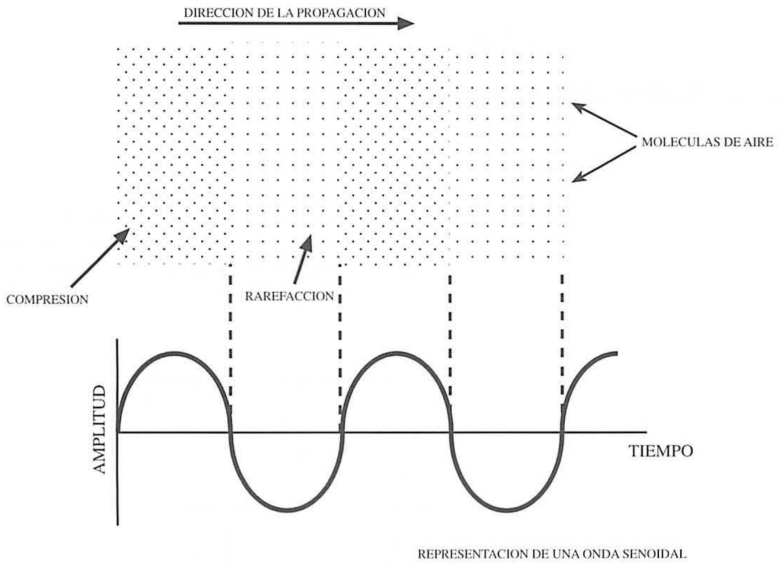


Figura 2.2: Propagación del sonido a través del aire
Fuente: J. Valenzuela

Para explicarlo de otra forma, suponga que con un martillo golpeamos una barra metálica la cual se encuentra fija en un punto en el espacio [3]. Esta acción hará que la barra se doble, teniendo un desplazamiento $x(t)$ en un tiempo t. Debido a la rigidez de la barra, habrá una fuerza F la cual hará que se restaure la posición inicial de la barra, tal como se muestra en la Figura 2.3.

[3] M. G. Christensen, *Introduction to Audio Processing*. Switzerland: Springer, 2019

La fuerza restauradora actuará como un resorte, por lo que el desplazamiento es proporcional:

$$F = -kx(t) \tag{2.1}$$

Donde k es una constante física particular de la barra. De acuerdo a

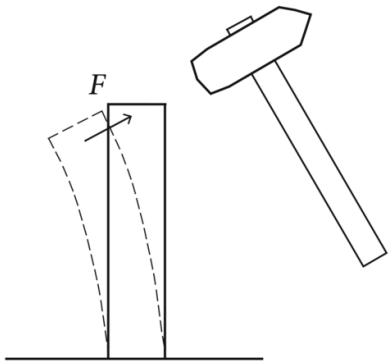


Figura 2.3: Vibraciones ocasionadas por el golpe de un martillo
Fuente: M. G. Christensen

la segunda ley de Newton, esta fuerza es igual a $F = ma$, donde m es la masa de la barra y a la aceleración, por lo tanto tendremos:

$$ma = -kx(t) \tag{2.2}$$

de esta manera, podemos relacionar la aceleración con el desplazamiento $x(t)$ ya que la aceleración es la derivada de la velocidad con la cual la barra se está moviendo. El concepto de velocidad se puede definir de la siguiente manera:

$$v(t) = \frac{dx(t)}{dt} \tag{2.3}$$

definiendo la representación de la aceleración como la derivada de la velocidad tenemos que:

$$a = \frac{dv}{dt} = \frac{d^2x(t)}{dt^2} \tag{2.4}$$

sustituyendo 2.2 en 2.4 obtenemos la siguiente expresión:

$$m\frac{d^2x}{dt^2} = -kx(t) \tag{2.5}$$

reacomodando los términos tendremos que:

$$\frac{d^2x(t)}{dt^2} = -\frac{k}{m}x(t) \tag{2.6}$$

por lo tanto, debido al sistema físico que presenta este modelo, existe una relación entre el desplazamiento $x(t)$, la constante física k y la masa m. Ahora nos podemos hacer la pregunta: ¿qué $x(t)$ obedece esta ecuación? En otras palabras, ¿qué tipo de función veremos al observar el desplazamiento a través del tiempo después de que la barra fue golpeada? La respuesta son las funciones sinusoidales:

$$x(t) = sin(\omega t) \tag{2.7}$$

donde ω es una frecuencia en radianes por segundo. Lo antes descrito se puede notar en la Figura 2.4.

Ahora, continuando el análisis, la derivada de la función sinusoidal en función del tiempo es la siguiente:

$$\frac{d}{dt}sin(\omega t) = \omega cos(\omega t) \tag{2.8}$$

derivando nuevamente obtenemos:

$$\frac{d^2}{dt^2}sin(\omega t) = -\omega^2 sin(\omega t) \tag{2.9}$$

sustituyendo 2.7 y 2.9 en 2.6:

$$-\omega^2 sin(\omega t) = -\frac{k}{m}sin(\omega t) \tag{2.10}$$

y así de esta manera podemos confirmar que el desplazamiento de este ejemplo realmente obedece las leyes de la física. De modo interesante, los senos no son la única solución a este problema, los cosenos también ofrecen una solución a este modelo. Al nosotros pensar el sonido como una perturbación del medio debido a una fuerza, este modelo también aplica en la teoría de sonido.

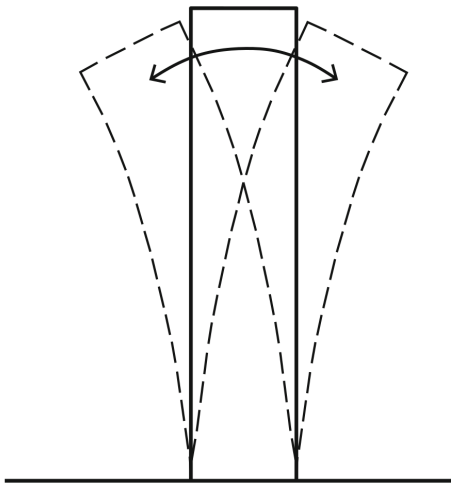


Figura 2.4: Representación del seno en las vibraciones de la barra
Fuente: M. G. Christensen

2.2 *Propiedades del sonido*

Amplitud

La amplitud es el desplazamiento máximo de cualquier partícula en un medio desde su posición de equilibrio. La amplitud es a lo que comúnmente llamamos volumen, y su magnitud se mide en decibelios

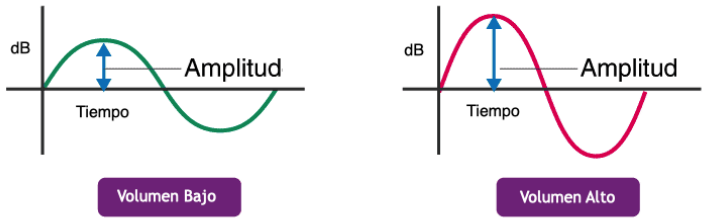


Figura 2.5: Amplitud de una función senoidal
Fuente: BYU'S

(dB). Entre mayor sea el desplazamiento de moléculas en el aire, mayor va a ser la amplitud y mayor será el volumen de esa señal [4]. La Figura 2.5 ilustra este concepto.

[4] J. Valenzuela, *Audio Digital: Conceptos Básicos y Aplicaciones*. Santa Monica, CA.: Backbeat Books, 1996

Frecuencia

La frecuencia es el número de vibraciones por segundo producidas por un objeto oscilante. Su magnitud se mide en ciclos por segundo - Hertz (Hz). Por ejemplo, la frecuencia de la nota de Do Central es de 262 Hz o ciclos por segundo [5]. Asimismo, al recíproco de la frecuencia ($T = 1/F$) se le puede definir como el tiempo que toma un ciclo en completarse. A este fenómeno físico se le conoce como **Periodo (T)**. En la Figura 2.6 se ejemplifica este concepto a través de la escala de Do.

[5] J. Valenzuela, *Audio Digital: Conceptos Básicos y Aplicaciones*. Santa Monica, CA.: Backbeat Books, 1996

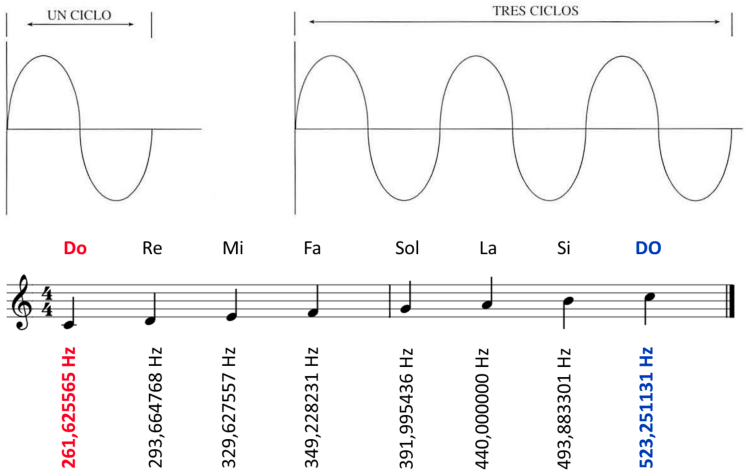


Figura 2.6: Frecuencias de las notas musicales
Fuente: J. Valenzuela

Longitud de onda

La longitud de onda es la distancia recorrida por una onda en un ciclo completo. Es fácil calcular la longitud de onda de un sonido dividiendo la velocidad del sonido por su frecuencia, es decir:

$$\lambda = \frac{V}{F} \tag{2.11}$$

Nota: la velocidad del sonido en el aire es de 335 metros por segundo.

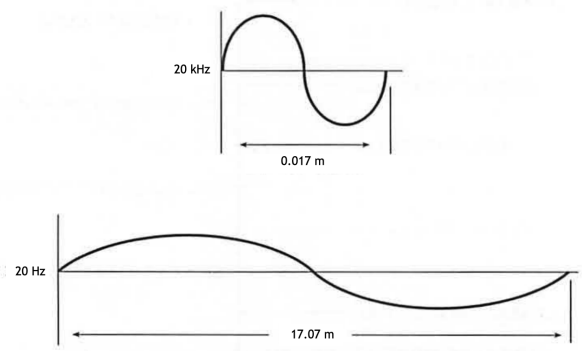


Figura 2.7: Longitud de onda
Fuente: J. Valenzuela

Fase

La fase es la relación de tiempo entre dos o más sonidos (ondas) en un determinado punto de sus ciclos. La fase de un sonido se mide en grados. Se dice que un sonido esta en fase con otro sonido cuando su relación en grados es la misma [6]. Por otro lado, cuando dos o más sonidos están fuera de fase quiere decir que relación en grados es diferente y auditivamente decae o se amplifica la amplitud, es decir, aumenta o disminuyen los decibelios.

[6] J. Valenzuela, *Audio Digital: Conceptos Básicos y Aplicaciones*. Santa Monica, CA.: Backbeat Books, 1996

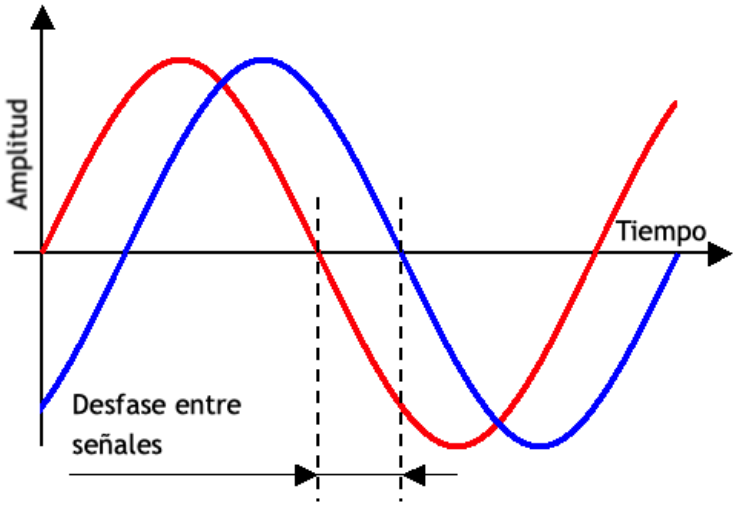


Figura 2.8: Fase de onda
Fuente: Wikipedia

2.3 *Muestro de una señal*

En procesamiento de señales, el muestreo se refiere a la reducción de una señal continua en una serie de valores discretos extraídos de la señal original [7]. La frecuencia/tasa de muestreo es el número de muestras tomadas en una cantidad de tiempo determinada. Una tasa de muestreo alta resulta en menor pérdida de información aunque impactando en el rendimiento computacional, mientras que una tasa baja resulta en mayor pérdida de información, pero permitiendo tener un mayor rendimiento computacional.

[7] "How to apply machine learning and deep learning methods to audio analysis." https://medium.com/comet-ml. Accessed: 2023-03-02

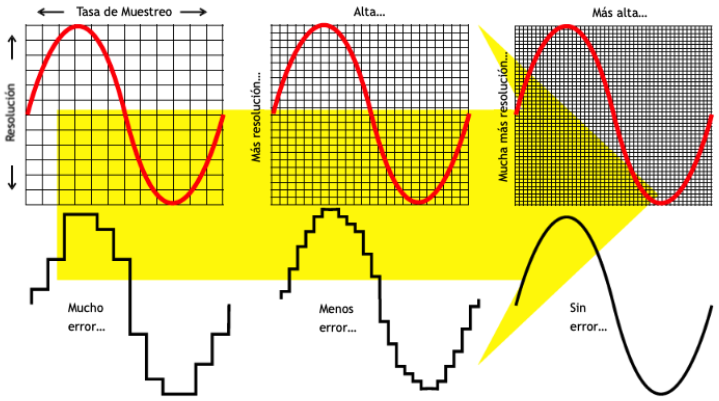


Figura 2.9: Fase de onda
Fuente: Desconocida

2.4 *Espectro de potencia*

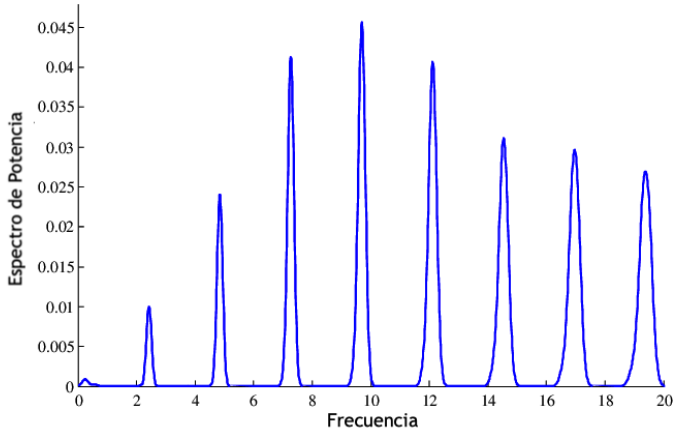
La función de densidad del espectro de potencia muestra las variaciones de potencia (energía) como una función dependiente de la frecuencia. En otras palabras, muestra en que frecuencias las variaciones de energía son fuertes y en cuáles las variaciones son débiles [8].

[8] "The data processing studio." https://www.cygres.com/OcnPageE/Glosry/SpecE.html. Accessed: 2023-03-03

Viendo al espectro, se puede deducir cuánta energía está contenida en cada frecuencia que compone la señal. El espectro de potencia responde a la pregunta "¿cuánta energía está contenida en las frecuencias que componen la señal?". La gráfica del espectro de potencia se realiza con la densidad contra frecuencia (Figura **??**).

2.5 *Transformada de Fourier*

La transformada de Fourier es una técnica matemática que se utiliza para descomponer una señal en sus componentes de frecuencia. Se le acredita el desarrollo de este método al físico y matemático francés Jean Baptiste Joseph Fourier. En 1807, Fourier presentó un artículo



al Instituto de Francia acerca del uso de ondas sinusoidales para representar la distribución del calor. El artículo presentaba contenido controversial ya que afirmaba que cualquier señal periódica continua podría ser representada como la suma de las de las ondas sinusoidales adecuadamente seleccionadas.

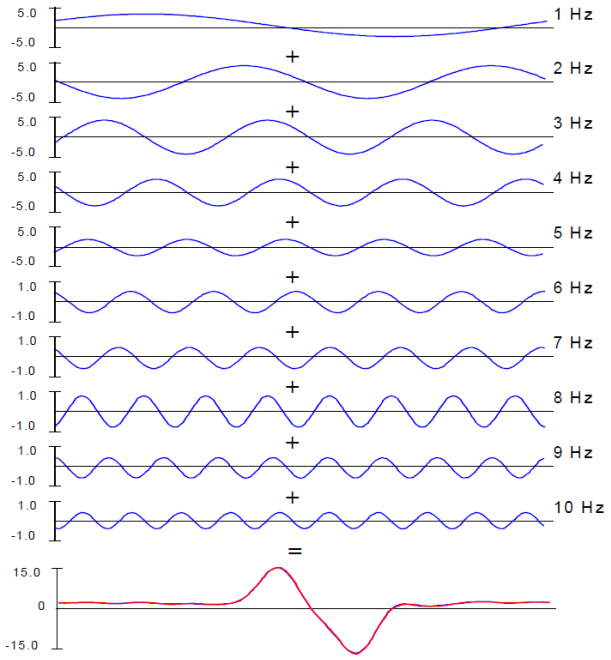


Figura 2.10: Descomposición de Fourier
Fuente: R. Howard

La Figura 2.10 ilustra el concepto del cual se basa el artículo de Fourier, siendo la señal color rojo el resultado final de la sumatoria de las señales de color azul.

Una señal puede ser continua o discreta, y puede ser periódica o aperiódica. La combinación de estas dos variables genera las siguientes cuatro categorías [9] (tal como se ilustra en la imagen 2.11):

[9] S. W. Smith, *The Scientist and Engineer's Guide to Digital Signal Processing*. San Diego, CA.: California Technical Publishing, 1998

- **Aperiódica - Continua:** Estas señales se extienden tanto en el infinito positivo como en el infinito negativo sin repetir un patrón periódico. Al procesamiento de este tipo de señales se le conoce como transformada de Fourier.
- **Periódica - Continua:** Aquí los ejemplos incluyen, senos, ondas cuadradas o cualquier otra onda que se repita en un patrón regular desde el infinito negativo hasta el infinito positivo. La versión de esta transformada se llama series de Fourier.
- **Aperiódica - Discreta:** Estas señales sólo están definidas en puntos discretos desde el negativo hasta el infinito positivo y no se repiten a si mismas en una forma periódica. Se le conoce como transformada Fourier en tiempo discreto.
- **Periódica - Discreta:** Estas señales son discretas y se repiten a si mismas de manera periódica desde el negativo hasta el infinito positivo. A esta clase de transformada a veces se le llama series discretas de Fourier, pero es más conocida como transformada discreta de Fourier.

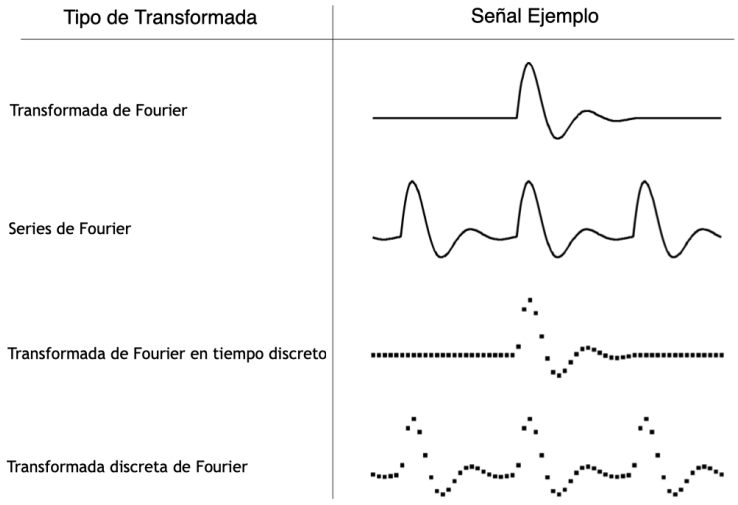


Figura 2.11: Categorías de la transformada de Fourier
Fuente: S. W. Smith

En este trabajo haremos uso de la transformada periódica y discreta, es decir, la transformada discreta de Fourier, la cual se define de la siguiente manera [10]:

[10] S. T. Kong, Q. and A. M. Bayen, *Python Programming and Numerical Methods*. Berkeley, CA.: Academic Press, 2020

$$X_k = \sum_{n=0}^{N-1} x_n e^{-i2\pi kn/N} \tag{2.12}$$

donde:

- i = Unidad imaginaria
- N = Número de muestras
- n = Iterador de muestras
- k = Iterador de frecuencias
- x_n = El valor de seno en la muestra n
- X_k = La transformada discreta de Fourier que incluye información tanto de amplitud como de fase.

También es importante mencionar que, la transformada rápida de Fourier (FFT, por sus siglas en inglés), es un algoritmo de procesamiento de señales que permite calcular la transformada de fourier discreta de una señal de manera eficiente.

Una manera de evaluar periodicidad en una señal es aplicando la transformada de Fourier. Especialmente, podemos aplicar la DFT o la DCT al logaritmo del espectro de potencia obteniendo así una representación conocida como cepstrum [11]. El eje de las x en el espacio del cepstrum es conocido como quefrency y es representado en segundos. Cepstrum es un anagrama de ”spectrum”, mientras que quefrency es un anagrama de ”frequency”. Los valores resultantes de esta transformación es una secuencia numérica la cual caracteriza la señal en un espacio de tiempo.

[11] “Introduction to speech processing.” https://wiki.aalto.fi/display/ITSP/. Accessed: 2023-03-19

En el cepstrum, las frecuencias bajas contienen información de que tan lento los rasgos cambian en el log-PSD. Otra gran utilidad del cepstrum es información relacionada con la estructura armónica del log-PSD. En la Figura 2.12 se muestra el flujo para realizar el cálculo del cepstrum.

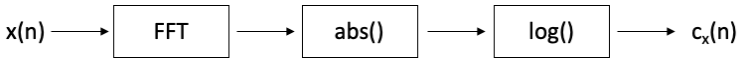


Figura 2.12: Algoritmo para el cálculo del cepstrum

Para mejorar la representación del cepstrum podemos incluir información de percepción humana auditiva al modelo. Esto es posible gracias a la escala de Mel.

2.6 *Escala de Mel*

Estudios han demostrado que los humanos no perciben frecuencias en una escala lineal. Somos mejores para detectar diferencias en frecuencias más bajas que en frecuencias más altas. Por ejemplo, podemos distinguir fácilmente la diferencia entre 500 y 1000 Hz pero difícilmente podremos distinguir una diferencia entre 10,000 y 10,500 Hz, aunque la distancia entre los dos pares sea la misma [12].

[12] "The mel spectrogram." https://medium.com/analytics-vidhya/understanding-the-mel-spectrogram-fca2afa2ce53. Accessed: 2023-03-03

En 1937, Stevens, Volkmann y Newmann propusieron una unidad tal que distancias tonales iguales sonarán igualmente distantes para el oyente sin importar la frecuencia en análisis (tal como se muestra en la imagen 2.13), a esto se llama escala Mel. La ecuación de Mel se describe de la siguiente manera:

$$m = 2595 * \log_{10}(1 + \frac{f}{700}) \tag{2.13}$$

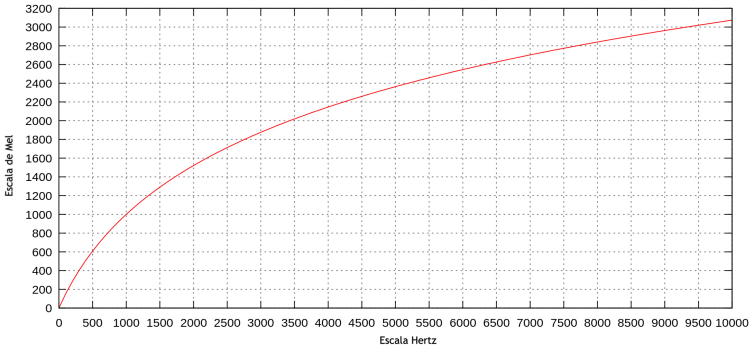


Figura 2.13: Escala de Mel
Fuente: Wikipedia

A través de los coeficientes cepstrales de Mel (MFCCs, por sus siglas en inglés), obtendremos una representación matemática de la señal de audio, la cual será utilizada para el análisis y la síntesis de la misma, logrando así la extracción de información sobre sus características acústicas.

2.7 *Filtros digitales*

La forma más básica para procesamiento de audio es a través del concepto de filtrado, y es gracias a este proceso es que se pueden realizar acciones interesantes sobre las señales de audio [13].

[13] M. G. Christensen, *Introduction to Audio Processing*. Switzerland: Springer, 2019

Los filtros digitales se presentan en una forma conocida como ecuaciones en diferencias, las cuales, por ejemplo, pueden ser sumas de

versiones retardadas de la entrada. Cada una de estas versiones retardadas es escalada a un número real a la que podemos llamar como coeficiente de filtrado.

Mientras que la salida de un filtro digital está determinada por las ecuaciones diferenciables, la salida de un filtro análogo está determinada como ecuaciones diferenciables e integrables. La matemática en los filtros análogos es mucho más avanzada, por lo que es más complicado realizar algunas acciones en comparación a los filtros digitales.

Un filtro digital toma la entrada como una secuencia de números, llamada x_n, donde n es el índice del tiempo, y modifica dicha entrada para producir otra secuencia de números, otra señal y_n, la cual representa la salida, tal como se representa en la Figura 2.14.

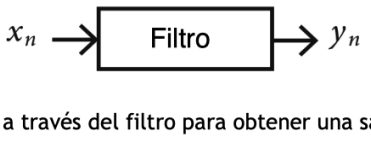


Figura 2.14: Diagrama de filtrado
Fuente: M. G. Christensen

2.8 *Data augmentation*

La técnica de incremento de datos (data augmentation, en inglés) es el proceso de aumentar artificialmente el número de datos, tomando como fábrica fuente información existente en el conjunto de datos inicial. Esto implica agregar alteraciones menores a los datos para generar información nueva amplificando así el conjunto de datos inicial [14].

Algunos modelos de inteligencia artificial dependen de grandes cantidades de datos para ser entrenados y así evitar el sobre-ajuste, sin embargo, los datos correctamente etiquetados del "mundo real" pueden ser limitados en cantidad. Mejorando la cantidad y diversidad de los datos de entrenamiento, el data augmentation se ha vuelto una técnica inevitable en modelos de aprendizaje de máquina [15].

Las diferencias principales entre datos sintéticos y data augmentation son:

- **Datos Sintéticos:** Son generados artificialmente sin utilizar datos reales como referencia. Comúnmente estos datos son producidos por redes neuronales generativas adversarias.
- **Data Augmentation:** Son datos derivados del conjunto de datos

[14] "The essential guide to data augmentation in deep learning." https://www.v7labs.com/blog/data-augmentation-guide. Accessed: 2023-03-26

[15] S. Yang, W. Xiao, M. Zhang, S. Guo, J. Zhao, and F. Shen, "Image data augmentation for deep learning: A survey," 2022

original con algún tipo de transformación, con el objetivo de aumentar la diversidad y cantidad de datos.

Específicamente para procesamiento de audio, algunas de las técnicas más recurrentes de data augmentation son las siguientes [16]:

[16] V.-V. Eklund, "Data augmentation techniques for robust audio analysis," Master's thesis, Tampere, Finland, 2019

- **Ruido Aditivo (additive noise):** Como su nombre lo sugiere, additive noise es la sumatoria de la señal original con ruido agregado. El ruido puede ser, por ejemplo, ruido gaussiano, ruido blanco o simplemente añadir una grabación de fondo.
- **Cambio de Tono (pitch shifting):** Todas las componentes frecuenciales de una señal de audio son modificados en tonalidades arriba o abajo, haciendo así ya sea más grave o agudo el sonido inicial.
- **Estiramiento del tiempo (time stetching):** La duración de la señal es escalada por un coeficiente manteniendo el tono original de la muestra.
- **Compresión de rango dinámico (dynamic range compression):** La señal de audio es procesada de manera tal que los sonidos de baja amplitud son amplificados y los sonidos de alta alta amplitud son atenuados.

2.9 *Redes neuronales*

Una red neuronal artificial es un sistema distribuido de procesamiento en paralelo construido a base de unidades simples (neuronas) que cuentan con la capacidad de almacenar y facilitar conocimiento [17]. Las redes neuronales se han desarrollado como generalizaciones matemáticas del procesamiento cognitivo humano, basándose de las siguientes suposiciones [18]:

[17] S. Haykin, *Neural Networks and Learning Machines*. Hoboken, New Jersey: Prentice-Hall, 1999

[18] L. Fausett, *Fundamentals of Neural Networks*. Hoboken, New Jersey: Prentice-Hall, 1994

- El procesamiento de la información ocurre en múltiples elementos llamados neuronas.
- Las señales son transferidas entre neuronas a través de conexiones.
- Cada conexión tiene un peso asociado, el cual, típicamente multiplica la señal transmitida.
- Cada neurona aplica una función de activación (usualmente no lineal) a la señal de entrada para determinar el valor de la señal de salida.

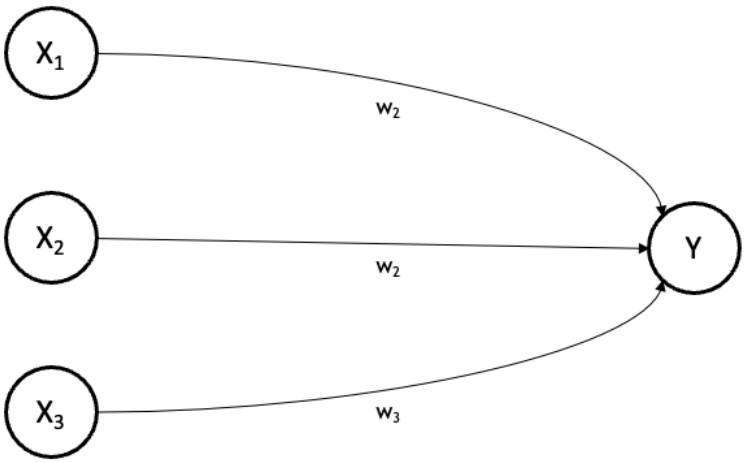


Figura 2.15: Red neuronal simple

Las redes neuronales se caracterizan por su patrón de conexiones entre neuronas (llamado arquitectura), su método para determinar los pesos en las conexiones (llamado algoritmo de entrenamiento/aprendizaje) y su función de activación.

En las conexiones neuronales, los pesos representan información que será utilizada por la red para resolver algún problema. Por ejemplo, una señal x_j que se encuentra cruzando a través de un enlace j el cual a su vez se encuentra conectado a la neurona k será multiplicada por el peso w_{kj}.

Cada neurona tiene un estado interno, llamado nivel de activación, el cual es una función matemática que será aplicada a la señal de entrada. El resultado de este proceso será enviado a otras neuronas. Por ejemplo, considera la neurona Y ilustrada en la Figura 2.15 la cual recibe entradas de las neuronas X_1, X_2 y X_3. Los pesos en las conexiones del conjunto de neuronas X a la neurona Y son W_1, W_2 y W_3 respectivamente. La señal de entrada, y_{in} a la neurona Y será la suma de las señales ponderadas, representándose dicho comportamiento de la siguiente manera:

$$y_{in} = w_1 x_1 + w_2 x_2 + w_3 x_3 \tag{2.14}$$

La activación de la neurona Y está dada por una función matemática. Esta función limita el valor de salida a un rango de valores finitos, típicamente en problemas de clasificación supervisada siendo la amplitud de valores entre [0, 1] o [-1, 1]. Por ejemplo, una función que entrega este tipo de rango de valores es la regresión logística, la cual se define de la siguiente manera:

$$f(x) = \frac{1}{1 + e^{-x}} \tag{2.15}$$

aunque también pueden existir funciones de diferentes rangos. Por ejemplo, la función ReLu (rectified linear unit, en inglés) incorpora un rango de [0, ínf], siendo expresada a través de la siguiente ecuación:

$$f(x) = \text{máx}(0, x) \tag{2.16}$$

esto quiere decir que valores negativos serán mapeados a un valor de cero, mientras que aquellos mayores a cero conservarán su valor.

Aunado al modelo antes descrito, podemos incorporar un elemento conocido como **bias** el cual será denotado como b_y. El bias tiene el efecto de incrementar o disminuir y_{in} dependiendo si el valor es positivo o negativo [19].

[19] S. Haykin, *Neural Networks and Learning Machines*. Hoboken, New Jersey: Prentice-Hall, 1999

Por lo tanto, en términos matemáticos podemos describir a la neurona Y a través de un par de ecuaciones:

$$y_{in} = b + \sum_{j=1}^{m} w_{kj} x_j \tag{2.17}$$

$$y_{out} = \phi(y_{in}) \tag{2.18}$$

donde ϕ representa la función de activación.

Ahora supóngase que la neurona Y tiene conexiones posteriores, neuronas Z_1 Y Z_2 con pesos v_1 y v_2 respectivamente, tal como se muestra en la Figura 2.16. Aunque la neurona Y manda la misma señal y a cada una de las unidades Z, los valores recibidos por las neuronas Z_1 Z_2 serán diferentes ya que cada señal es escalada por los pesos v_1 y v_2 [20].

[20] L. Fausett, *Fundamentals of Neural Networks*. Hoboken, New Jersey: Prentice-Hall, 1994

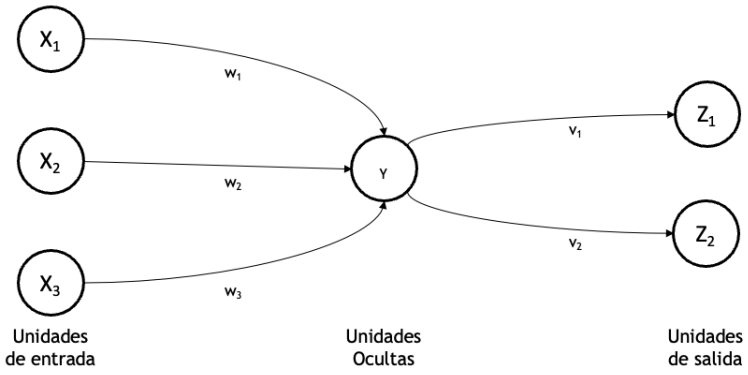


Figura 2.16: Red neuronal con una capa oculta

Aunque la red neuronal 2.16 es bastante simple, la presencia de una capa oculta aunado a la integración de una función de activación no lineal brinda la capacidad de poder resolver muchos más problemas

que en el modelo presentado en la imagen 2.15. Por otro lado, esta última propuesta de red neuronal resulta más complicada de entrenar.

Las Redes Neuronales de Propagación hacia Adelante

Las Redes Neuronales de Propagación hacia Adelante, también conocidas como perceptrones multicapa, son el modelo más representativo en el área de aprendizaje profundo, las cuales son una variante del modelo del pereceptrón original propuesto por Rosenblatt en 1950 [21].El objetivo de estas redes es el aproximar el valor resultante de una función f^*. Por ejemplo, en un problema de clasificación, la función $f = f^*(x)$ mapea una entrada x a una categoría y. Dicho de manera general, una red neuronal de propagación hacia adelante define una función de mapeo $y = f(x, \theta)$ y aprende el valor de los parámetros θ que resultan de la mejor aproximación de la función previamente definida [22].

[21] H. e. a. Ramchoun, "Multilayer perceptron: Architecture optimization and training," *International Journal of Interactive Multimedia and Artificial Intelligence*, vol. 4, no. 1, 2016

[22] B. Y. Goodfellow I. and C. A., *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016

A estos modelos se le acuñe el término de red ya que típicamente son representadas como un conjunto compuesto de diferentes funciones. Por ejemplo, podemos tener tres funciones f^1, f^2 y f^3 conectadas en cadena, formando así:

$$f(x) = f^3(f^2(f^1(x))) \tag{2.19}$$

esta estructura de cadena es la más comúnmente utilizada en redes neuronales. En este caso, f^1 se le conoce como la primer capa de la red, f^2 la segunda capa y así de manera consecutiva. La longitud de esta cadena de funciones define la **profundidad** de la red. Se conoce como capa de entrada a la primer capa de la red y está compuesta por un conjunto de neuronas que reciben los datos de entrada. La capa de entrada es responsable de recibir los datos de entrada y transformarlos en una forma que la red neuronal pueda entender y procesar. A la última capa se le conoce como capa de salida. Durante el entrenamiento de la red neuronal, dirigimos a que $f(x)$ sea lo más aproximado a $f^*(x)$. Cada ejemplar contenido en los datos de entrenamiento x están acompañados por un valor o etiqueta $y \mapsto f^*(x)$.

Los datos utilizados durante el entrenamiento especifican directamente el valor esperado en la capa de salida, mientras que el comportamiento de las capas intermedias no están directamente relacionadas con dicho comportamiento, el algoritmo de entrenamiento debe decidir como utilizar esas capas para producir el valor esperado en la capa de salida [23]. Se le conoce como capas ocultas a las capas que se encuentran entre la capa de entrada y la capa de salida debido a que no están conectadas directamente a los valores de entrada o salida de la red.

[23] B. Y. Goodfellow I. and C. A., *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016

Estos modelos son llamados "de propagación hacia adelante" ya que son un tipo de red neuronal artificial en la que la información fluye en una sola dirección, desde la capa de entrada a través de una o más capas ocultas hasta la capa de salida, sin ciclos o retroalimentación. Cuando a estas redes se les extiende la funcionalidad de recibir retroalimentación se les concede el nombre de "redes neuronales recurrentes".

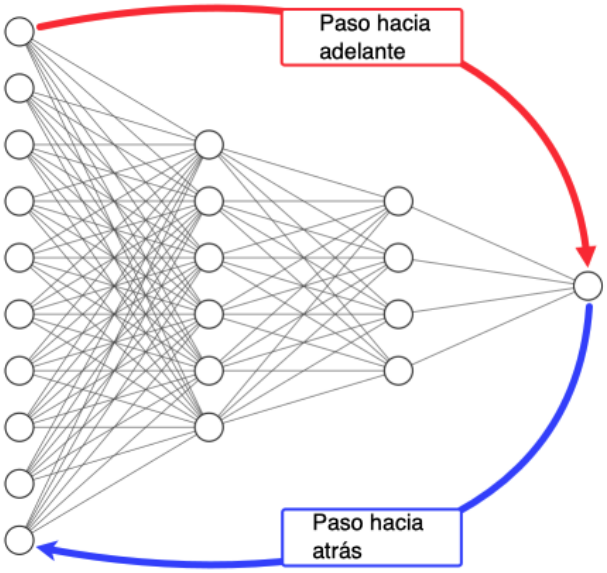


Figura 2.17: Backpropagation
Fuente: Neptune.AI

Propagación hacia atrás o "backpropagation" en inglés es un concepto de suma importancia en las redes neuronales de propagación hacia adelante. Es un algoritmo de entrenamiento el cual ajusta los pesos de las conexiones entre neuronas [24]. Durante el backpropagation, el error entre el valor de salida predicho y el valor real es calculado y propagado hacia atrás a través de la red para ajustar los pesos de las conexiones. Este proceso es iterado múltiples veces (épocas) hasta lograr que la salida de la red sea lo suficientemente precisa.

[24] "Backpropagation in a nutshell everything you need to know." https://towardsdatascience.com/. Accessed: 2023-04-01

ADAM Como Optimizador

En 2015, el algoritmo de estimación adaptativa del momentum (ADAM, por sus siglas en inglés) fue propuesto por Diederik P. Kingma and Jimmy Lei Ba. en su artículo de conferencia "ADAM: A Method For Stochastic Optimization", donde ellos mismos definen a ADAM como "un método eficiente de optimización estocástica que sólo requiere gradientes de primer orden con bajos requerimientos de memoria" [25]:

- **Optimización estocástica:** Es el proceso de optimizar una función objetivo con la presencia de procesos aleatorios.

[25] "Complete guide to adam optimization." https://towardsdatascience.com/complete-guide-to-adam-optimization-1e5f29532c3d. Accessed: 2023-04-09

- **Gradientes de primer orden:** ADAM sólo requiere la primer derivada para operaciones de optimización por gradientes.

En física, **momentum** se define como un cuerpo en movimiento, por lo tanto, SGD (stochastic gradient descent, por sus siglas en inglés) con momentum incorpora los gradientes de iteraciones anteriores para agilizar la convergencia del algoritmo.

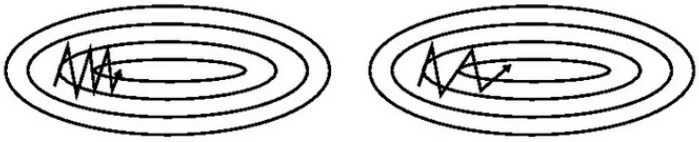


Figura 2.18: SGD con o sin momentum
Fuente: G. B. Orr

El SGD con momentum se calcula a través de la media móvil del gradiente utilizando el promedio móvil exponencial, método que se usa comúnmente para suavizar series de tiempo. Esto queda expresado a través de la siguiente ecuación:

$$w_{t+1} = w_t - \alpha V_t \tag{2.20}$$

siendo:

$$V_t = \beta V_{t-1} + (1-\beta)\frac{\partial L}{\partial w_t} \tag{2.21}$$

donde w_{t+1} s el valor de la función en el tiempo $t+1$, β es un factor de peso, $\frac{\partial L}{\partial w_t}$ es el gradiente de L, la función de pérdida a minimizar y α la tasa de aprendizaje, hiperparámetro encargado de determinar el tamaño del "paso" entre cada iteración. Los hiperparámetros son parámetros ajustables que permiten controlar el proceso de entrenamiento de un modelo.

Implementando esta metodología, podremos encontrar "tendencias" las cuales nos puedan ayudar a agilizar los tiempos de convergencia en comparación al gradiente descendente, tal como se muestra en la imagen 2.18, donde del lado izquierdo se ilustra el gradiente descendente tradicional y del lado derecho con momentum.

Ahora bien, los algoritmos del gradiente adaptativo (AdaGrad, por sus siglas en inglés) y la propagación de la raíz cuadrada del valor medio de los cuadrado (RMSProp, por sus siglas en inglés) son considerados como modelos **adaptativos**, lo cual quiere decir que sus tasas de aprendizaje pueden variar entre cada iteración del optimizador. Ambos están basados igualmente en el SGD. En el algoritmo AdaGrad, la velocidad de la tasa de aprendizaje depende de los cambios observados en los gradientes de la variable observada: entre más alto sea la tasa de

cambio, menor será la tasa de aprendizaje, mientras tanto, RMSProp basa su aceleración de convergencia en función de la penalización: si existe un parámetro que haga oscilar a la función objetivo, entonces debemos penalizarlo [26].

El algoritmo ADAM involucra lo mejor de las metodologías antes descritas: momentum y algoritmos adaptativos. Básicamente, ADAM controla la tasa de aprendizaje de manera tal que existe una actualización mínima cuando estamos cerca de un máximo de la función objetivo, mientras que realiza una tasa de cambio mayor cuando nos encontramos estancados en un mínimo local.

[26] "Complete guide to adam optimization." https://towardsdatascience.com/complete-guide-to-adam-optimization-1e5f29532c3d. Accessed: 2023-04-09

2.10 *Métricas en machine learning*

Aunque existen diferentes algoritmos de clasificación, todos comparten principios similares al momento de ser evaluados. En un problema de clasificación **supervisada**, cada dato de prueba presentará dos valores: valor esperado o conocido (etiqueta) y valor de salida del modelo. Por esta razón, al momento de evaluar un modelo de clasificación, podemos asignar su resultado en cualquiera de las siguientes categorías [27]:

- **Verdadero Positivo (T_P):** el valor esperado es positivo y la predicción también es positiva.
- **Verdadero Negativo (T_N):** el valor esperado es negativo y la predicción también es negativa.
- **Falso Positivo (F_P):** el valor esperado es negativo pero la predicción es positiva.
- **Falso Negativo (F_N):** el valor esperado es positivo pero la predicción es negativa.

[27] M. Hossin and M. Sulaiman, "A review on evaluation metrics for data classification evaluations," *IJDKP*, vol. 5, no. 2, 2015

Estas cuatro categorías son la base para la mayoría de las métricas de evaluación de un modelo de clasificación. En la tabla 2.1 se muestran las métricas más populares y la manera en que son construidas.

Métrica	Ecuación	Enfoque de Evaluación
Exactitud	$\frac{t_p+t_n}{t_p+f_p+t_n+f_n}$	Tasa de predicciones correctas sobre el total de instancias evaluadas
Tasa de Error	$\frac{f_p+t_n}{t_p+f_p+t_n+f_n}$	Tasa de predicciones incorrectas sobre el total de instancias evaluadas
Sensibilidad	$\frac{t_p}{t_p+t_n}$	Fracción de positivos clasificados correctamente
Especificidad	$\frac{t_n}{t_n+f_p}$	Fracción de negativos clasificados correctamente
Precisión (p)	$\frac{t_p}{t_p+f_p}$	Positivos clasificados correctamente sobre el total de datos clasificados como positivos
F1-Score	$\frac{2pr}{p+r}$	Representa la media armónica entre el sensibilidad y la precisión

Cuadro 2.1: Métricas para un modelo de clasificación supervisada

Un punto fundamental a considerar al momento de evaluar un clasificador es que no es recomendable utilizar únicamente la exactitud como métrica. La razón de esto es porque un conjunto de datos puede estar desbalanceado. Por ejemplo, si un modelo está diseñado para predecir fraudes bancarios de un conjunto de datos donde el 95 % de la información no está etiquetada como fraude, entonces las predicciones de fraude pueden estar altamente sesgadas [28].

[28] "Evaluation metrics." https://spark.apache.org/docs/latest/mllib-evaluation-metrics.html. Accessed: 2023-04-13

3 Metodología

Una vez estudiada la teoría de señales y después de haber analizado la fuente y razón de los datos, se implementaron las siguientes actividades para la realización de esta tesis:

- Lectura y análisis inicial de los datos
- Data augmentation como técnica de diversificación
- Extracción de rasgos característicos de audio utilizando los coeficientes cepstrales en las frecuencias de Mel (ver sección 2.6).
- Modelado predictivo de clasificación de audio utilizando redes neuronales

3.1 Análisis inicial

Se utilizó el conjunto de datos **"UrbanSound8k"**, el cual nace a partir de la necesidad de investigar de manera precisa la relación de los sonidos urbanos y su taxonomía, área de investigación poco profundizada en comparación a otras ramas de investigación como lo es el procesamiento de lenguaje natural, música y bioacústica, etcétera [1].

[1] J. C. . B. J. P. Salamon, J., "A dataset and taxonomy for urban sound research," *ACM international conference on Multimedia*, vol. 22, pp. 1041–1044, 2014

El conjunto de datos se conforma de 8,732 sonidos, cada uno con una duración menor a 4 segundos y etiquetado con su respectiva clase (10 en total). Las clases contenidas en los datos son las siguientes (en la Figura 3.1 se muestra una gráfica de tiempo contra amplitud por cada clase):

- Ladridos de perros (a)
- Bocina de un coche (b)
- Música callejera (c)
- Sirenas de ambulancias (d)
- Martillos neumáticos (e)
- Niños jugando (f)

- Aire acondicionado (g)
- Disparos por armas de fuego (h)
- Motores (i)
- Perforaciones por taladro (j)

La descarga de los datos se hizo directamente de la fuente publicada por el autor de los mismos. Esta información es ofrecida de manera gratuita para uso no comercial bajo los términos del "Creative Commons Attribution Noncommercial License [2]".

[2] "Creative commons attribution noncommercial license." http://creativecommons.org/licenses/by-nc/3.0/. Accessed: 2023-03-05

Una vez descargada la información es necesario explorar sobre los datos para conocer y familiarizarse con los sonidos contenidos en el paquete. Se graficó una sola muestra de audio de cada clase, buscando así notar diferencias y similitudes entre las señales utilizando las librerías **librosa** y **matplotlib**. En la Figura 3.1 se muestra un ejemplo de lo antes mencionado, graficando una muestra aleatoria por clase.

Tal como se puede observar, a través de una simple gráfica de amplitud contra tiempo se pueden notar patrones característicos de cada clase. Por ejemplo, el ruido generado por el aire acondicionado suele ser constante y sin variaciones de amplitud, mientras que los sonidos de disparos por un arma de fuego suelen ocasionar un sonido instantáneo y con poca duración a través del tiempo.

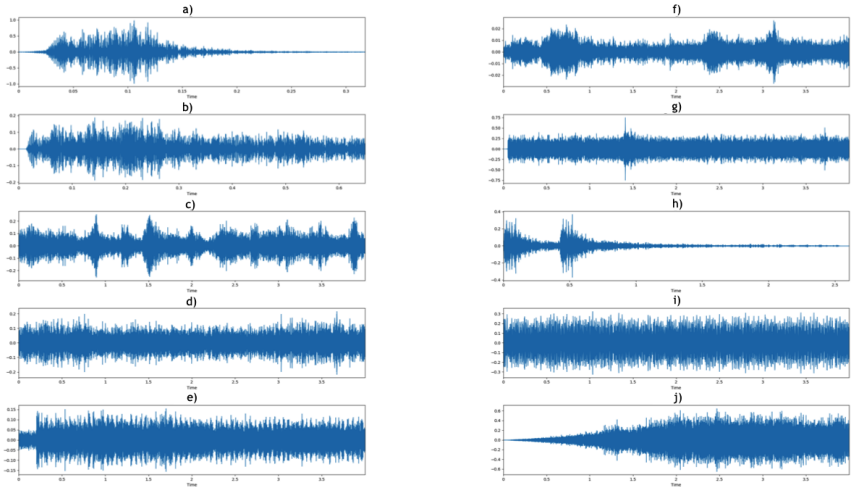


Figura 3.1: Gráfica de las señales por clase

3.2 *Procesamiento de audio*

Frecuentemente nos podemos encontrar con una cantidad limitada de datos de entrenamiento disponibles, lo que puede hacer que los modelos de aprendizaje automático se sobreajusten a los datos de entrenamiento y tengan un rendimiento deficiente en datos de prueba no vistos. El data augmentation ayuda a abordar este problema al generar nuevas muestras de datos a partir de las muestras de datos de entrenamiento existentes, aumentando la cantidad de datos disponibles para el entrenamiento y reduce la probabilidad de sobreajuste.

Se replicó la experimentación de Justin Salamon y Juan Pablo Bello [3] publicada en el artículo *"Deep Convolutional Neural Networks and Data Augmentation for Environmental Sound Classification"* .

[3] J. Salamon and J. P. Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," *CoRR*, vol. abs/1608.04363, 2016

En términos de implementación, se utilizó la librería MUDA (Musical Data Augmentation) [4]. MUDA implementa funciones para procesar y manejar data augmentation en señales de sonido. Después de aplicar alguna deformación, el audio modificado y sus anotaciones pueden ser almacenados en formato JSON con especificación de música anotada (JAMS, por sus siglas en inglés).

[4] "Musical data augmentation." https://muda.readthedocs.io/en/stable. Accessed: 2023-03-16

A través de MUDA, se deformó el conjunto de datos utilizando cuatro técnicas de procesamiento de audio, aumentando de 8,732 datos a 174,640 favoreciendo así la suficiencia y diversidad en los datos. Las variables de deformación fueron las siguientes:

- **Estrechamiento del tiempo:** Incrementar o decrementar la velocidad con la que se reproduce un audio sin cambiar la tonalidad (pitch) del mismo. Cada muestra fue deformada utilizando los siguientes factores de estrechamiento: {0.81, 0.93, 1.07 y 1.23}
- **Desplazamiento de tonalidad :** Un semitono en música es la distancia más pequeña entre dos notas en una escala. Este procesamiento consiste en aumentar o disminuir la frecuencia/tono del audio procesado. Cada muestra fue modificada a través de los siguientes valores (representados en semi tonos): {-3.5, -2.5, -2, -1, 1, 2, 2.5 y 3.5}
- **Compresión de rango dinámico:** Es una operación de procesamiento de audio la cual reduce los sonidos de alta amplitud o amplifica aquellas frecuencias con baja amplitud. A través de MUDA, se utilizaron cuatro estándares de compresión: {estándar de música, películas, habla y radio}
- **Ruido de fondo:** Se mezcla la muestra con un audio de fondo,

esto con la finalidad de agregar ruido a la secuencia original y tener un conjunto de datos más cercanos a la realidad. Cada audio fue mezclado con cuatro escenas auditivas: {trabajadores urbanos, tráfico urbano, personas en ambientes urbanos y sonidos de parques urbanos}.

Un punto medular de esta tesis son los datos extraídos mediante el procesamiento digital de señales, ya que son utilizados como entrada para el modelado predictivo.

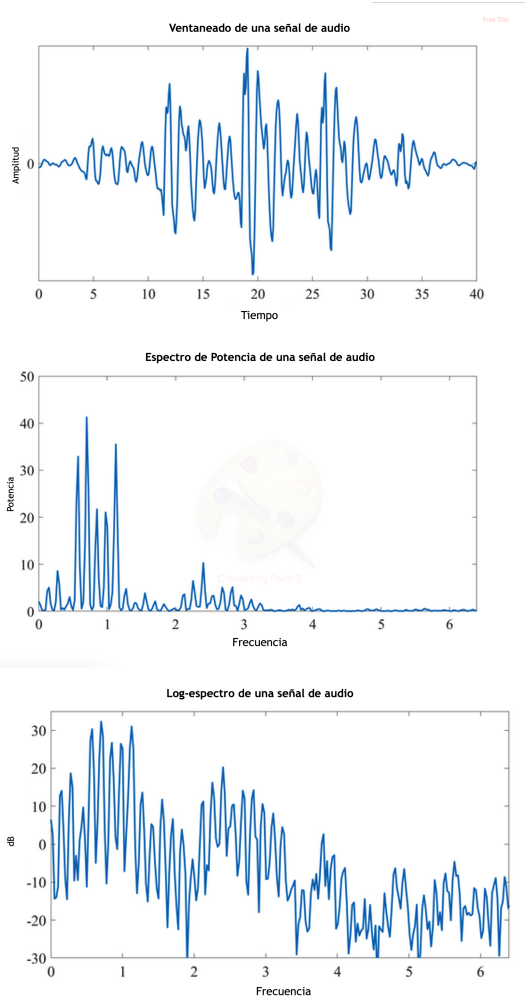


Figura 3.2: Espectro de Potencia
Fuente: T. Bäckström et al.

El espectrograma es una representación bastante útil en el procesamiento digital de audio al permitirnos la observación del comportamiento del sonido a través del tiempo, aunque también presenta ciertos inconvenientes, particularmente, no es un método eficiente de representación numérica en términos de coeficientes [5]. Típicamente se busca tener información de la señal la cual puede ser manipulada en forma de coeficientes. Es por ello que aplicamos la metodología de los

[5] T. e. a. Bäckström, *Introduction to Speech Processing*. 2 ed., 2022

coeficientes cepstrales en las frecuencias de Mel.

El espectro de potencia (PSD) de una ventana de tiempo contiene información fundamental en ese espacio de tiempo. Si además convertimos la escala a dB del PSD a través de una transformación logarítmica, obtendremos una representación sencilla de interpretar y aproximando el comportamiento de la señal al sentido auditivo del humano [6]. En la Figura 3.2 se puede observar lo antes mencionado.

[6] "Introduction to speech processing." https://wiki.aalto.fi/display/ITSP/. Accessed: 2023-03-19

Modificando el flujo mostrado en la sección 2.5 (Figura 2.12), podremos agregar la escala de Mel de la siguiente manera:

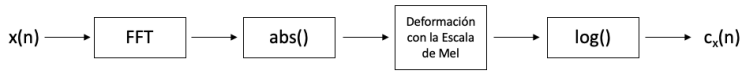


Figura 3.3: Algoritmo para el cálculo de MFCCs

Por lo tanto, el algoritmo final para obtener los MFCCs queda de la siguiente manera:

- Segmentación de la señal
- Aplicar una función de ventana sobre el segmentado procesado, esto ayuda a eliminar el fenómeno de **fuga espectral**. En general, las señales de audio no son periódicas, concepto que no cumple con la suposición de la FFT. La discordancia entre ambas postulaciones conlleva a errores durante la transformación de la señal. A este este efecto se le conoce como fuga espectral (imagen 3.4). Las funciones de ventaneo nos ayudan a contrarrestar este efecto
- Cálculo de la transformada de Fourier
- Transformar los datos de acuerdo a la escala de Mel. Esto se logra a través de aplicar un banco de filtros el cual consiste en M filtros triangulares los cuales están espaciados linealmente de acuerdo a la escala de Mel, siendo el centro de cada filtro proveniente de la fórmula de la escala de Mel [7]
- Aplicar una función logarítmica a los coeficientes resultantes después de haber aplicado el banco de filtros
- Eliminar correlación en los datos a través de la transformada discreta del coseno

[7] G. Zigelboim and I. Shallom, "A comparison study of cepstral analysis with applications to speech recognition," *International Conference on Information Technology: Research and Education*, 2006

La librería **librosa** cuenta con la función para hacer el cálculo de los MFCCs en una línea de código, aunque se desarrolló el flujo paso a paso para dar a conocer al lector el proceso más detalladamente.

Al tener los datos muestreados y posteriormente por segmentos, la información ya es discreta, perdimos datos de nuestra señal análoga. La

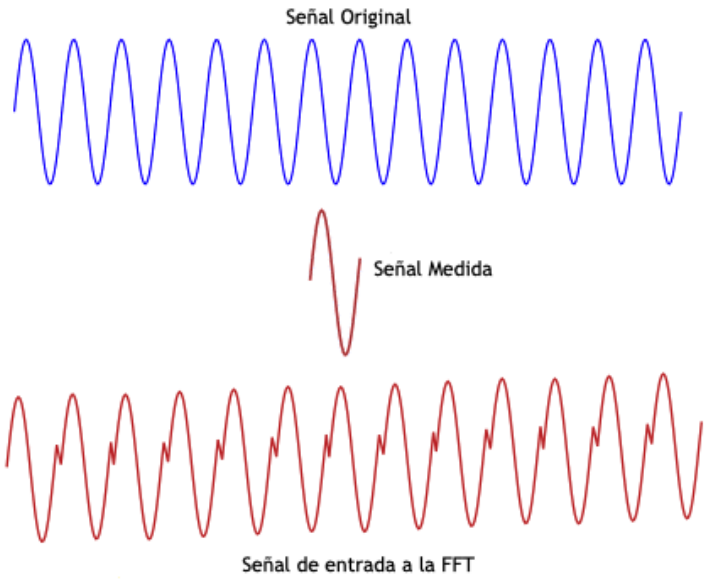


Figura 3.4: Fuga espectral
Fuente: M. Viswanathan

periodicidad se logra a través de la segementación. Por último, debemos asegurar continuidad, es por ello que se debe aplicar la función de ventana de Hamming. En la imagen 3.5 se muestra un ejemplo de este proceso en uno de los sonidos del conjunto de datos.

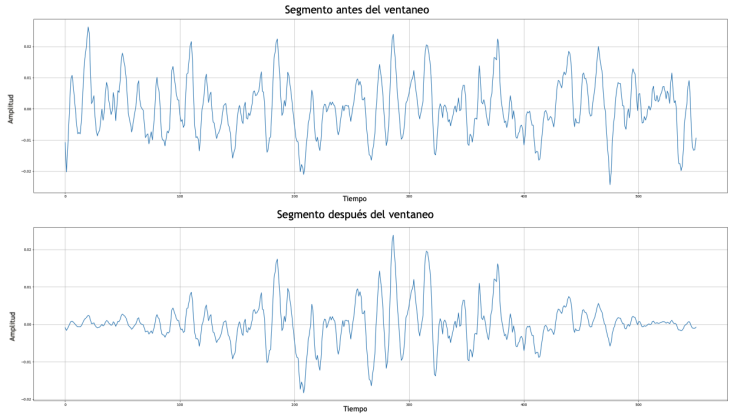


Figura 3.5: Ventana de Hamming

El siguiente paso consiste en hacer el cálculo del espectro de potencia. Como el término lo sugiere, el espectro de potencia representa la proporción del total del poder de la señal contribuido por cada componente de frecuencia.

Para lograr lo antes mencionado, se realizó una FFT en cada frame de cada audio para obtener el espectro de frecuencias de cada señal del conjunto de datos. Se utilizaron n puntos en cada frame para realizar

la FFT: si n es menor que la longitud del frame, la entrada es truncada, si n es mayor, se aplica un relleno con ceros.

Posterior a realizar la FFT sobre la señal, se procede a hacer el cálculo del espectro de potencia de la siguiente manera:

$$PS(f) = |X(f)|^2 \tag{3.1}$$

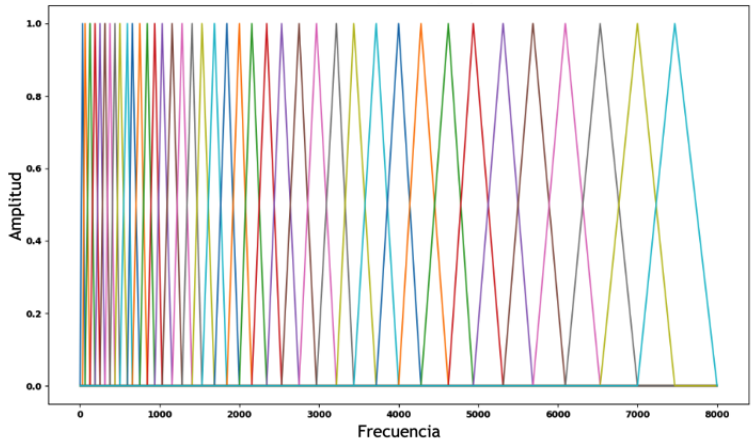


Figura 3.6: Banco de filtros utilizando 40 filtros triangulares

Finalmente, el siguiente paso consiste en completar nuestro objetivo, el cálculo de los MFCCs. Estudios han demostrado que los humanos no perciben frecuencias en una escala lineal. Somos mejores para detectar diferencias en frecuencias más bajas que en frecuencias más altas.

En este paso buscamos calcular el filtro en la escala de Mel y después pasar nuestras señales a través de él. Cada filtro en el banco de filtros es triangular (un filtro triangular es un tipo de filtro utilizado en el procesamiento de señales que tiene una respuesta de frecuencia en forma de triángulo. Es un filtro pasa-bajas que atenúa las componentes de alta frecuencia de la señal mientras permite pasar las componentes de baja frecuencia), comportamiento que elimina la respuesta lineal de la señal. Esto se puede observar en la Figura 3.6.

Por último, se aplica una transformación discreta del coseno para eliminar la correlación de nuestro resultado, y así obtener los coeficientes que serán usados por nuestro algoritmo predictivo.

3.3 Modelado predictivo

Para el proceso de clasificación se seleccionó una red neuronal de propagación hacia adelante. La librería empleada para la implementación fue **Keras** [8], módulo que basa los modelos de redes en neuronales en la clase **Sequential**, la cual apila un grupo de capas definiendo así la arquitectura de la red neuronal.

[8] "Keras api reference." https://keras.io/api/. Accessed: 2023-04-01

Los modelos de inteligencia artificial, en su generalidad, se caracterizan por su alto nivel de parametrización, razón por la cual es fundamental conocer los efectos que tendrá el ajuste de cada variable en el resultado final. Buscamos contar con un modelo robusto y preciso evitando el sobre-entrenamiento. Los parámetros tratados en este trabajo fueron:

- **Número de neuronas:** Keras nos permite definir el número de neuronas que queremos implementar en cada capa de la red neuronal. Este valor tiene un efecto tal que, aumentando el número de neuronas de manera desconsiderada, el modelo será sobre-entrenado, además de elevar el costo computacional. El incremento de complejidad de las fronteras de decisión es otro de los efectos de esta práctica [9]. Se recomienda que, para la capa de entrada, se defina el número de neuronas igual al número de dimensiones del conjunto de datos, mientras que para las capas ocultas existen ciertas reglas generales, de las cuales, se puede optar por alguna de las siguientes opciones[10]:
  - Debe estar contenido entre la cardinalidad de neuronas de la capa de entrada y el de la capa de salida.
  - Debe ser igual a 2/3 del tamaño de la capa de entrada más el tamaño de la capa de salida.
  - Debe ser menor al doble del número de neuronas de la capa de entrada.
- **Número de capas:** Aumentando el número de capas en nuestra arquitectura obtendremos una mayor capacidad de representación. Esto quiere decir que la red será capaz de modelar funciones más complejas, teniendo como efecto negativo el sobre-entrenamiento.
- **Capas "dropout":** La regularización es una técnica la cual hace pequeñas modificaciones al algoritmo de aprendizaje de manera tal que se mejore la generalización de un problema. Esto da como resultado mejoras en el desempeño del modelo para datos no conocidos [11]. Las capas dropout es una tipo de regularización el cual, en cada iteración (época), selecciona de manera aleatoria algunas neuronas y las remueve de la arquitectura. El hiper-parámetro de esta

[9] B. Y. Goodfellow I. and C. A., *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016

[10] "How do determine the number of layers and neurons in the hidden layer?." https://medium.com/geekculture/introduction-to-neural-network-2f8b8221fbd3. Accessed: 2023-04-03

[11] "An overview of regularization techniques in deep learning (with python code)." https://www.analyticsvidhya.com/blog/2018/04/. Accessed: 2023-04-06

variable es la probabilidad con la que cada neurona será eliminada del modelo. Este método se aplica en cada paso feedforward, por lo tanto, implementándose en cada lote (batch, en inglés) una vez por época.

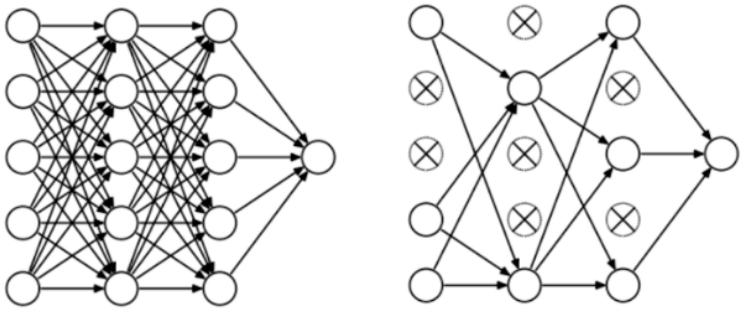


Figura 3.7: Dropout
Fuente: Z. Wang et al.

- **Función de Activación:** Función que será empleada en las neuronas de la red. Para las capas ocultas se sugiere usar por defecto la función ReLU (Rectified Linear Unit), mientras que para la capa de salida, al enfrentarnos a un problema de clasificación multiclase, **Softmax** es la función ideal, la cual presenta similitudes a la regresión logística, pero con la diferencia de que la función Softmax (ver Figura 3.8) está diseñada para trabajar en escenarios multiclase [12]. A diferencia de la función de la tangente hiperbólica, la cual mapea los valores de entrada a un rango entre -1 y 1, la función Softmax mapea los valores de entrada a una distribución de probabilidad, asegurando que la suma de los valores posteriores a la transformación será igual a uno, logrando así representar cada valor como una probabilidad de pertenencia a cada clase.

[12] B. Y. Goodfellow I. and C. A., *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016

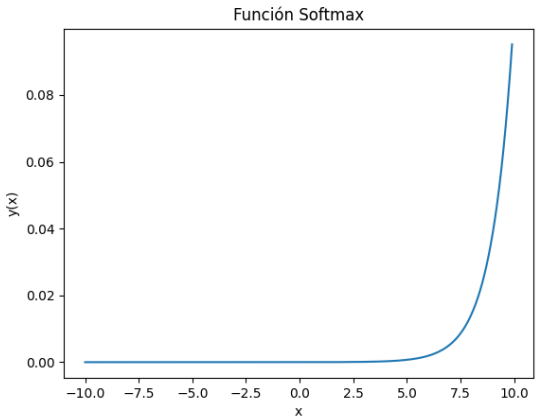


Figura 3.8: Función Softmax

- **Función de Pérdida:** Su objetivo es optimizar una función de costo

(generalmente minimizar) durante la etapa de entrenamiento. Siendo este un problema de clasificación multiclase, la recomendación es utilizar una función de pérdida probabilística, permitiéndonos así seleccionar aquel valor que presente mayor pertenencia a cierta clase. En la librería Keras, la función **categorical crossentropy** cumple con las características antes mencionadas.

- **Optimizador:** Son algoritmos o métodos utilizados para minimizar o maximizar la función de pérdida. Investigaciones sugieren que el algoritmo **ADAM** ha demostrado mejor desempeño sobre otros optimizadores, como el AdaGrad, SGD, RMSP, etcétera [13].
- **Lotes:** Las redes neuronales son entrenadas con la mayor cantidad de observaciones posibles. Cuando un conjunto de datos es demasiado grande o los recursos computacionales son limitados, puede llegar a consumir mucho tiempo utilizar toda la información para la actualización del gradiente durante la etapa de entrenamiento. Como solución se propusieron los lotes, los cuales son porciones de los datos. La desventaja de implementar esta metodología es la calidad del modelo, ya que como efecto lateral, la calidad de generalización de datos no conocidos se puede ver degradada.

Se hizo supervisión del modelo en cada iteración utilizando los valores de las métricas de precisión y error de la función de pérdida, mientras que para analizar el comportamiento global del modelo, las métricas empleadas fueron la exactitud (accuracy), precisión, sensibilidad (recall) y score F1.

[13] "Adam." https://optimization.cbe.cornell.edu/index.php?title=Adam. Accessed: 2023-04-04

4 Resultados

En este capítulo, se presentan los resultados finales de la investigación y se discuten los diferentes enfoques que se utilizaron para diseñar la implementación del modelo predictivo. Se realizaron diversas pruebas modificando poco a poco la arquitectura de la red neuronal hasta encontrar la configuración óptima.

4.1 Perceptrón de una sola capa

Se inició con un enfoque ingenuo, definiendo una arquitectura lo más simple posible con el objetivo de observar el comportamiento de la red en dicho escenario. Para ello, se ejecutó como primer intento un **perceptrón** (ver capítulo 2.9) de una sola capa con una función de activación **sigmoide**.

Se definió el perceptrón con un total de neuronas en la capa de entradas igual al doble del total de dimensiones (coeficientes de Mel). El número total de coeficientes que retorna el algoritmo MFCC está relacionado con la longitud del audio procesado y el ancho de ventana utilizado. Al ser todos los sonidos de tamaño símil y al haber sido todos procesados con los valores por defecto de la librería **Librosa**, cada audio entrega un arreglo de longitud igual a 128.

Para la división de datos de entrenamiento y validación se usó una tasa 70-30 respectivamente, además de usar un algoritmo de estratificación con barajeamiento/mezcla, el cual buscará tener un balance en la densidad de clases en ambos subconjuntos, además de hacer validación cruzada con una lógica estocástica: se seleccionan muestras de manera aleatoria en cada iteración (fold) para generar un subconjunto de datos de entrenamiento y validación buscando tener la misma densidad de datos en cada clase. Se realizaron diez iteraciones seleccionando un valor de semilla distinto entre cada iteración, además de realizar $k = 10$ número de folds. Al término de la ejecución, se seleccionó aquella iteración que presentó mejores valores de ajuste en términos de **exactitud**, esto debido a que nuestro conjunto de datos se encuentra balanceado

en relación a la densidad de datos por clase, además que todas las clases tienen el mismo peso de importancia, razón por la que tener un dato mal clasificado no es crítico como lo pudiese llegar a ser un modelo de clasificación de fraudes bancarios.

El desempeño del pereceptrón utilizando este primer enfoque fue deficiente, mostrando una baja capacidad de generalización en los datos de validación:

Tipo de Dato	Exactitud	Precisión	Sensibilidad	F1 Score
Entrenamiento	0.569	0.599	0.544	0.556
Validación	0.565	0.595	0.540	0.552

Cuadro 4.1: Métricas: modelo del perceptrón

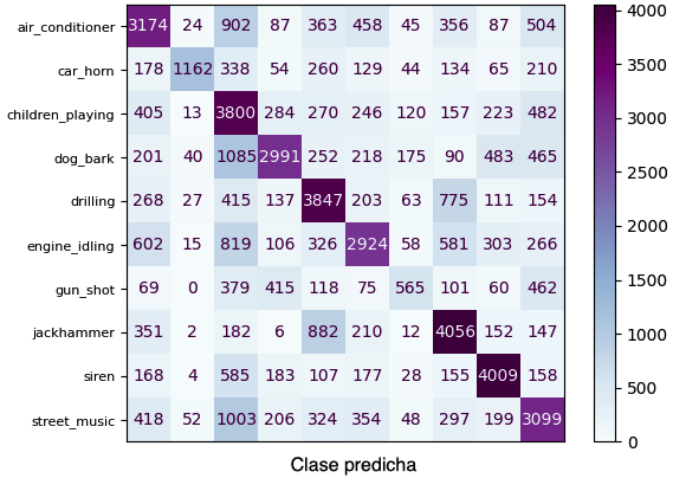


Figura 4.1: Matriz de confusión: perceptrón

Similarmente, este bajo desempeño se puede observar gráficamente en la matriz de confusión mostrada en la Figura 4.1. Los resultados nos indican que nuestros datos no viven en un espacio linealmente separable a través de una metodología rígida, como lo es perceptrón, razón por la que podemos optar y explorar en modelos que tengan mayor flexibilidad en la definición de las fronteras de decisión.

4.2 *Perceptrón multicapa*

El siguiente paso consistió en agregar capas ocultas a nuestro modelo inicial. Siguiendo las recomendaciones del capítulo 3, se implementó la siguiente arquitectura, la cual también se muestra en la Figura 4.2:

- Tres capas: capa de entrada, oculta y de salida.

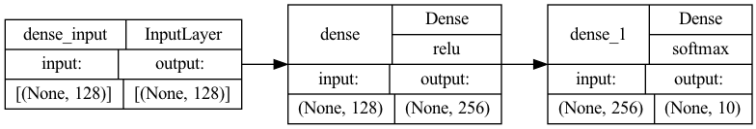


Figura 4.2: Representación gráfica de la arquitectura del modelo

- Número de neuronas:
  - Capa de entrada: Total de dimensiones del conjunto de datos - 128.
  - Capa oculta: Doble del total de dimensiones - 256.
  - Capa de salida: Número de clases - 10.
- Optimizador: ADAM con tasa de aprendizaje igual a 0,001.
- Función de pérdida: Entropía cruzada categórica.
- Función de activación: ReLU.
- Función de salida: Softmax.
- Número de épocas: 500.
- Tamaño de cada lote: 128 (no se observó degradación del modelo al usar un número menor: se experimentó con n igual a 1, 32 y 128).

A su vez, se utilizó la función **EarlyStopping**, forzando al modelo a detener su entrenamiento en caso de que no observar mejora en las métricas después de cierto número de épocas. Se implementó como métrica de paro los valores de error pertenecientes a la función de pérdida para el subconjunto de validación, con un límite de 30 épocas y un delta de error de 0,01. Los resultados fueron los siguientes:

Tipo de dato	Exactitud	Precisión	Sensibilidad	F1 Score
Entrenamiento	0.978	0.979	0.976	0.978
Validación	0.940	0.942	0.938	0.940

Cuadro 4.2: Métricas: primer modelo feedforward

Como se puede analizar a partir de los valores de la tabla 4.2, el modelo presenta valores elevados en todas las métricas, aunque claramente existe sobre-entrenamiento. Este comportamiento ocurre cuando el modelo no es lo suficientemente robusto para generalizar datos no conocidos, lo cual es un gran problema, ya que esperamos poder clasificar de manera correcta aquellos datos ajenos a nuestro conjunto de datos original.

Otra manera de observar el sobre-entrenamiento de nuestro modelo es a través de las gráficas de exactitud (Figura 4.3) y pérdida (Figura

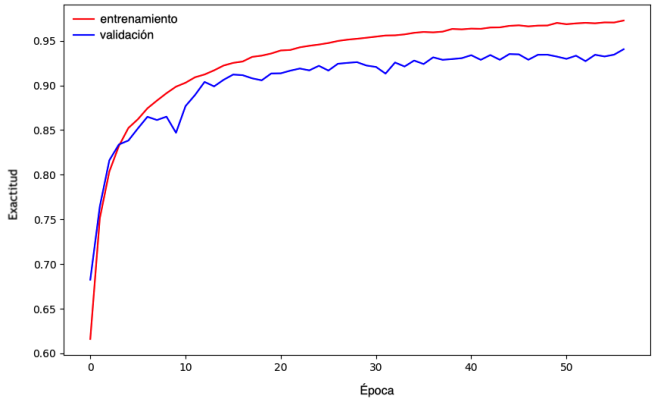


Figura 4.3: Exactitud: primer modelo feedforward

4.4). Se espera que en ambas gráficas, las curvas de entrenamiento-validación se asemejen lo más posible entre sí. Para este ejemplo, el sobre-entrenamiento se nota claramente en la gráfica de error de la función de pérdida, en donde para los datos de validación, el optimizador no logra generalizar lo suficiente, estancando el error en valores próximos a 0,4.

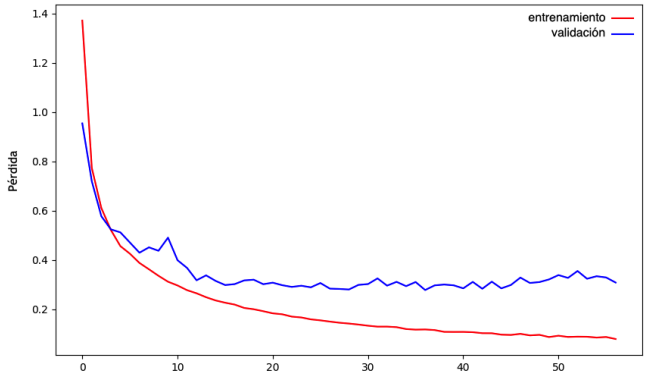


Figura 4.4: Función de pérdida: primer modelo feedforward

De igual manera, podemos observar el desempeño del modelo a través de la curva característica de operación del receptor (ROC, por sus siglas en inglés). Esta gráfica nos permite analizar la compensación entre la tasa de verdaderos positivos y la tasa de falsos positivos. Además, si se calcula el área bajo la curva (conocida como **AUC**, por sus siglas en inglés), obtendremos un valor que nos ayudará a concluir que tan robusto fue nuestro modelo. El rango de valores del AUC es de 0 a 1, siendo 0 el peor escenario (ningún dato fue correctamente clasificado) y 1 una clasificación perfecta.

Cada clase tendrá su propia curva ROC y su propio valor AUC. A pesar del sobre-entrenamiento, se puede notar observando la Figura 4.5 que nuestro modelo presentó una clasificación bastante decente, retornando valores elevados de AUC para todas las clases.

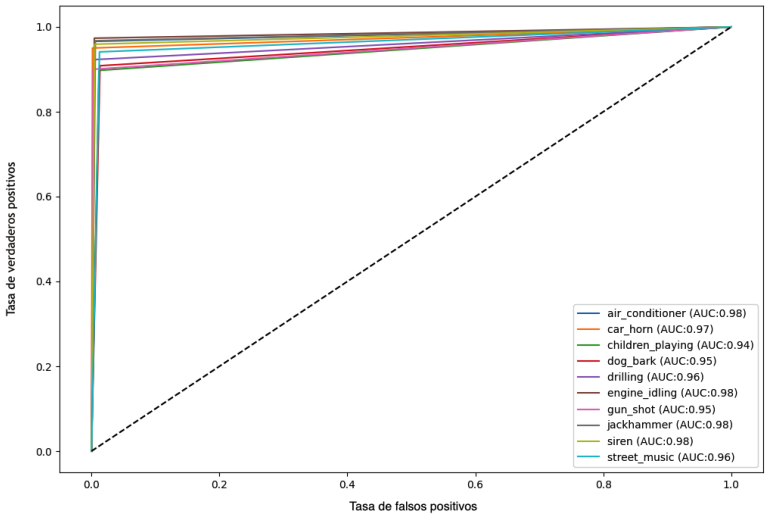


Figura 4.5: Curva ROC: primer modelo feedforward

Al graficar la matriz de confusión (Figura 4.6), podemos reafirmar las conclusiones realizadas a través de la curva ROC.

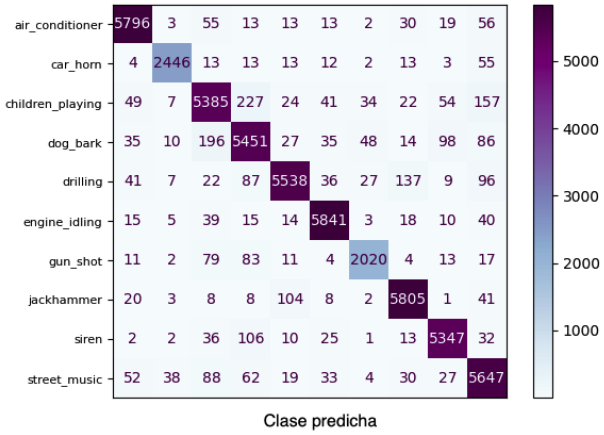


Figura 4.6: Matriz de confusión: primer modelo feedforward

Ahora, nuestro siguiente paso es reducir el sobre-entrenamiento, esto nos ayudará a contar con un modelo más robusto y lo suficientemente

confiable al momento de clasificar datos ajenos al conjunto de datos de entrenamiento.

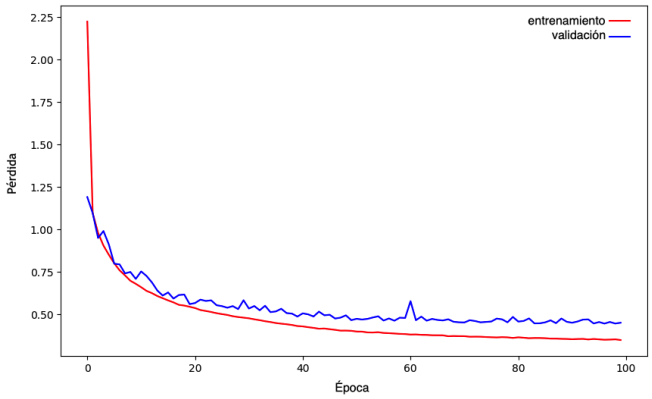


Figura 4.7: Función de pérdida: reducción de neuronas en la capa oculta

El primer intento para evitar el sobre-entrenamiento consistió en la reducción del número de neuronas en la capa oculta. Se optó por fijar el número de neuronas a un total de un medio de la cardinalidad de dimensiones del conjunto de datos, esto con el objetivo observar el desempeño del modelo y analizar los valores resultantes de dicha arquitectura, además que no se notó mejoría utilizando valores superiores o iguales al número de dimensiones.

Siguió persistiendo sobre-entrenamiento, denotando las siguientes observaciones: la curva de validación en la función de pérdida presentó un comportamiento asintótico decreciente (ver Figura 4.7), estancándose en un valor superior en comparación modelo del perceptrón. Por otro lado, observando la curva ROC se detectó una caída en los valores AUC para todas las clases (ver Figura 4.8), impactando directamente en el desempeño global del modelo.

4.3 *Perceptrón multicapa con regularización L1*

Otro método de reducción del sobre-entrenamiento en una red neuronal es el **decaimiento de pesos** o mejor conocido como **regularización de pesos**. A través de este método, podemos agregar un término adicional a nuestra función de pérdida (conocido como bias), permitiendo tener mayor flexibilidad al momento de minimizar la función.

La magnitud de aumento que tendrá nuestro bias está definido por un hiper-parámetro de penalización λ, donde el rango de valores está contenido entre cero e infinito positivo: si seleccionamos un valor igual

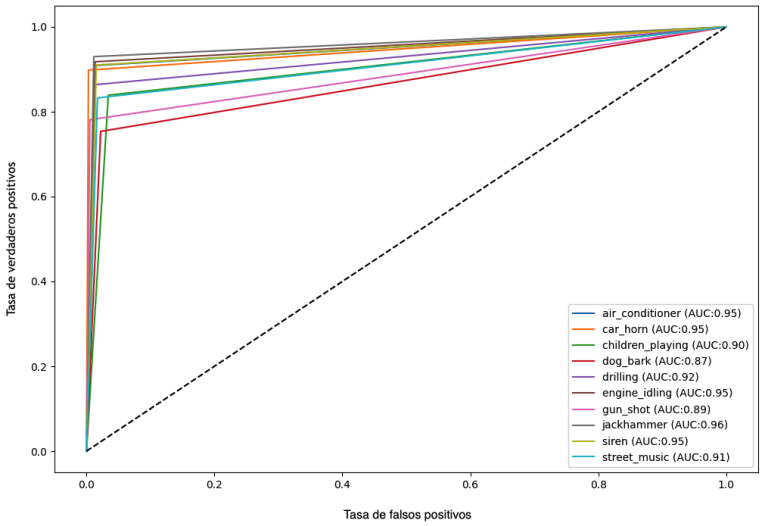


Figura 4.8: Curva ROC: reducción de neuronas en la capa oculta

a cero, no se aplicará regularización a la función de pérdida, mientras si seleccionamos un valor alto para λ, el bias será mayor.

Los métodos principales de regularización de pesos es la regresión Lasso (L1) y la regresión Ridge (L2) [1]:

$$L1 := d_1(y, x, w) = f(y, x, w) + \lambda \sum_{i=1}^{n} ||w|| \tag{4.1}$$

$$L2 := d_1(y, x, w) = f(y, x, w) + \lambda \sum_{i=1}^{n} w^2 \tag{4.2}$$

donde $f(y, x, w)$ es la función de pérdida y w los pesos de la red.

A través de Keras podemos aplicar estos métodos de regularización ya sea en el bias o en el kernel (pesos excluyendo al bias). Se notó un bajo impacto al incorporar regularización en el bias, y por parsimonia del modelo, se hizo ajuste únicamente en el kernel. La técnica que presentó mejores resultados fue la regresión Lasso: se utilizó el doble del total de dimensiones en la capa oculta debido a los bajos resultados de AUC obtenidos en el experimento anterior. Se trabajaron diferentes valores para el hiper-parámetro de peso de la regularización, definiendo finalmente la mejor asignación en $\lambda = 0{,}01$.

Los resultados obtenidos muestran una notable mejora en el sobre-entrenamiento, además de mantener métricas similares a la primer arquitectura analizada (ver Figura 4.9). Tanto en la Figura 4.14 como

[1] J. Kasche and F. Nordström, "Regularization methods in neural networks," tesis de licenciatura, Uppsala, Suecia, 2020

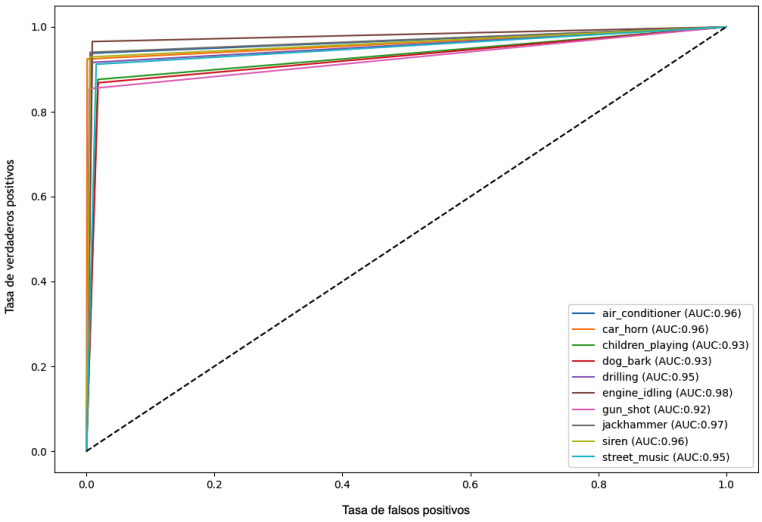


Figura 4.9: Curva ROC: modelo feedforward con regularización L1

en la Figura 4.13 se observa una mayor superposición entre las curvas de ajuste, además de notar un comportamiento más asintótico en la curva de pérdida de validación. A través de estos valores se comprueba el potencial de la regularización en los pesos de la red, sin embargo, aunque en menor grado, en nuestro modelo sigue presentando sobre-entrenamiento.

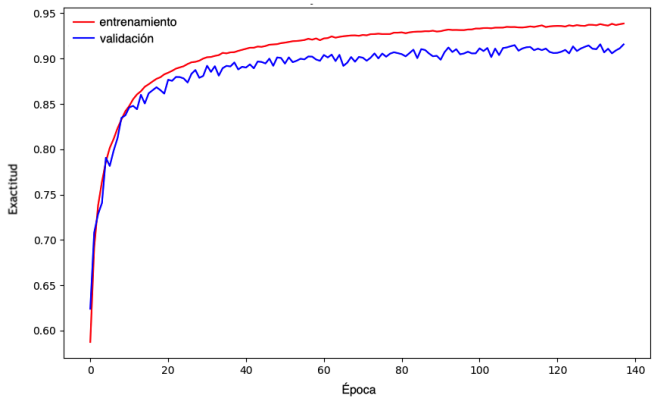


Figura 4.10: Exactitud: modelo feedforward con regularización L1

4.4 *Perceptrón multicapa con dropout*

Por último, se optó por explorar la regularización a través de las capas dropout (ver capítulo 3). Utilizando esta técnica, se realizaron

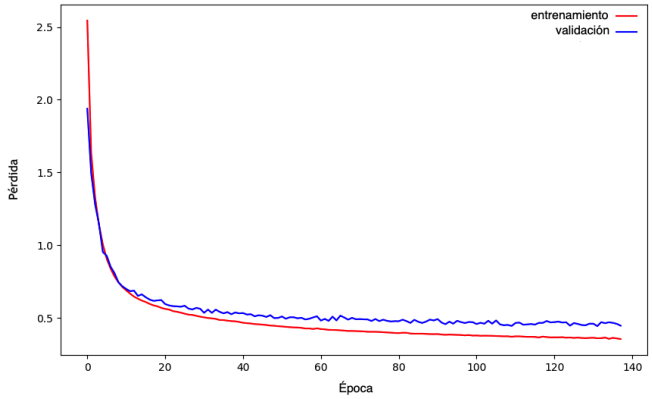


Figura 4.11: Función de pérdida: modelo feedforward con regularización L1

múltiples experimentos para encontrar la combinación de hiperparámetros adecuada para resolver nuestro problema.

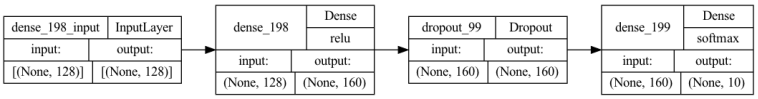


Figura 4.12: Representación gráfica de la arquitectura del modelo final

La arquitectura final implementada consistió en los siguientes elementos (representación gráfica mostrada en la Figura 4.12):

- Cuatro capas: capa de entrada, oculta, dropout y de salida.
- Número de neuronas:
  - Capa de entrada: Total de dimensiones del conjunto de datos - 128.
  - Capa oculta: Total de dimensiones del conjunto de datos multiplicado por 1.25 - 160.
  - Capa dropout: Probabilidad de desactivación igual a 0,15.
  - Capa de salida: Número de clases - 10.
- Optimizador: ADAM con tasa de aprendizaje igual a 0,001.
- Función de pérdida: Entropía cruzada categórica.
- Función de activación: ReLU.
- Función de salida: Softmax.
- Número de épocas: 500.
- Tamaño de cada lote: 128 .
- Función de término: función de validación de pérdida con una paciencia de 30 épocas y un delta de error de 0,01.

Los resultados obtenidos a través de esta arquitectura son los siguientes:

Tipo de Dato	Exactitud	Precisión	Sensibilidad	F1 Score
Entrenamiento	0.939	0.943	0.937	0.939
Validación	0.913	0.917	0.909	0.913

Cuadro 4.3: Métricas: modelo final

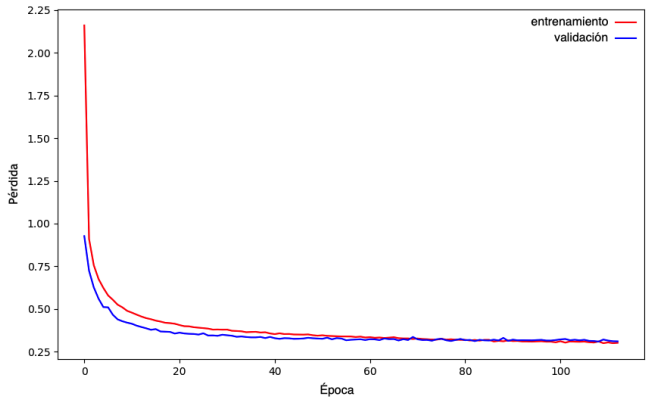


Figura 4.13: Función de pérdida: modelo final

Como se puede observar en comparación al primer modelo (tabla 4.2), se redujo notablemente el sobre-entrenamiento con esta nueva arquitectura, corroborando este comportamiento a través de las gráficas de exactitud (4.14) y pérdida (4.13).

La gráfica de pérdida resalta una notable mejora en la generalización del modelo, mostrando que el optimizador entrega resultados bastante símiles tanto para datos conocidos como aquellos ajenos a los datos de entrenamiento.

Ahora, algo interesante a observar de la gráfica de exactitud es que los datos de validación muestran mayor desempeño que los datos de entrenamiento. Esta tendencia es consecuencia del haber agregado la capa dropout.

La red neuronal construye diferentes arquitecturas en cada paso "feedforward" a través de la capa dropout. Esto quiero decir que se genera una "sub-red" por cada lote que es procesado por el modelo [2]. Al ser el dropout una técnica probabílistica, existe la posibilidad de construir 2^n (contabilizando todas las capas) número de subredes durante el entrenamiento del modelo. En la Figura 4.15 se muestra un ejemplo de este comportamiento, aplicando dropout solo a la capa de

[2] "Chapter 7: Regularization (part 2)." https://shanzhenren.github.io/csci-699-replnlp-2019fall/. Accessed: 2023-04-09

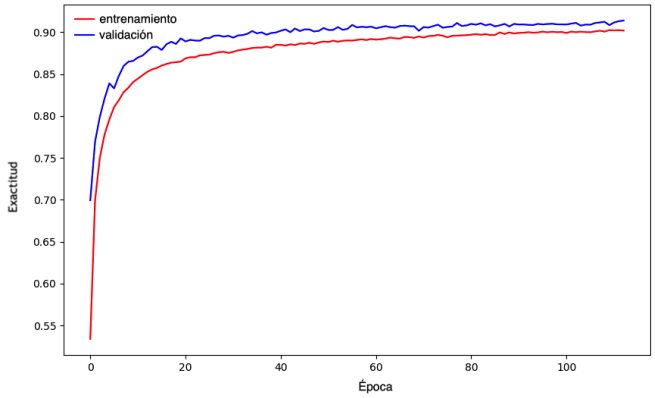


Figura 4.14: Exactitud: modelo final

entrada y oculta, obligando a siempre tener las unidades de salida, y por lo tanto, disminuyendo el número de combinaciones posibles.

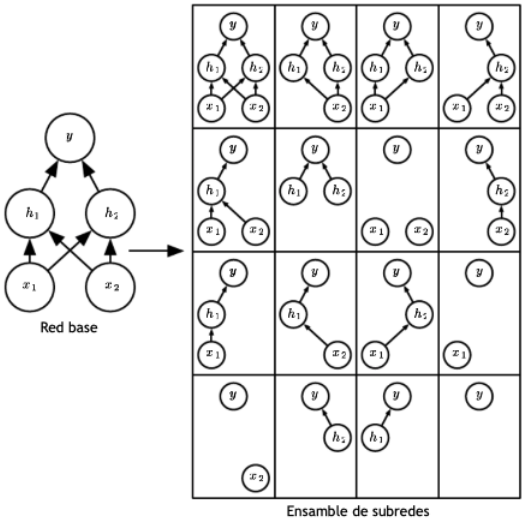


Figura 4.15: Subredes de una arquitectura neuronal
Fuente: D. Lee

Al final, cuando todas las épocas son ejecutadas, el modelo realiza un promedio de todos los pesos calculados en cada paso feedforward, ya que al haber obtenido diferentes submodelos durante el entrenamiento, tendremos diferentes pesos de conexión entre neuronas. Esto quiere decir que muy probablemente el modelo terminará con la misma arquitectura que la inicial, sólo que los pesos de conexión fueron calculados en cada lote usando regularización, y durante el término del entrenamiento, promediados.

Entonces, durante el entrenamiento de la red, Keras realiza cálculos de la función de pérdida en cada lote, y al final, cuando todo el conjunto

de datos es procesado (una época), hace un promedio de las pérdidas observadas. Esto hace conflicto al momento de hacer validación, ya que se espera contar con todas las neuronas iniciales, razón por la cual la capa dropout es apagada durante la validación y los pesos utilizados son el promedio de los valores conocidos hasta ese momento de la ejecución, por lo tanto, siendo bastante factible el hecho de encontrar menor desempeño en los datos de entrenamiento.

Aún con lo antes mencionado se puede observar que el mejor resultado se logró utilizando la capa dropout, entregando un modelo robusto ante datos no conocidos.

5 Conclusiones

A través de la investigación realizada en este trabajo se logró comprender, diseñar e implementar una metodología lo suficientemente robusta cumpliendo con el objetivo de clasificar sonidos en ambientes urbanos, área que se encuentra en constante desarrollo y que implica un reto computacional debido a la dificultad implícita en los datos fuente aunado a la complejidad en la implementación de los diversos algoritmos los cuales abarcan diferentes disciplinas de conocimiento. Se exploraron las redes neuronales como modelo predictivo de clasificación supervisada, encontrando resultados satisfactorios en la resolución del problema planteado.

El procesamiento digital de audio es un área compleja, ya que el contexto del problema a resolver conlleva a diversas técnicas a implementar para contar con información útil. En específico, en este trabajo se buscó encontrar una manera de trasladar el espacio auditivo a alguna representación numérica prolífera para un modelo de clasificación. Se demostró la capacidad y potencial que presentan los coeficientes cepstrales en las frecuencias de Mel, ya que son una herramienta que logra de manera eficiente hacer esta traducción numérica, además de entregar información valiosa y poco redundante, disminuyendo así la incertidumbre en los datos a emplear durante el modelado predictivo. El mayor reto y que está fuera del alcance es la manera en que fueron recabados los sonidos, ya que existen múltiples factores que pueden degradar su calidad, como la distancia entre el emisor y receptor de sonido, traslape de sonidos de diferentes clases en el mismo audio, la calidad del aparato receptor (SNR), etcétera.

El conjunto de datos utilizado cuenta con sonidos bastante limpios en términos de distinción auditiva en cada clase, por lo que se exploraron técnicas que ayudaron diversificando el espacio de datos, además de simular un contexto más realista. El data augmentation nos permitió implementar esta filosofía a través de múltiples técnicas de procesamiento digital de audio, lo cual enriqueció de gran manera la información a utilizar para alimentar nuestro modelo predictivo.

Referente a la implementación del modelo de inteligencia artificial, se notó un bajo rendimiento en los resultados obtenidos por el perceptrón simple. Esto nos permitió hacer un análisis exploratorio para definir la técnica a implementar, teniendo como siguiente paso aumentar la complejidad en las fronteras de decisión. Se optó continuar con el perceptrón multicapa, en donde se presentaron diversos retos, siendo el sobre-entrenamiento el principal de ellos, ya que no sólo es buscar que metodología nos ayuda a reducir este comportamiento, sino que también implica encontrar la combinación de hiper-parámetros óptima, proceso que es altamente dependiente del contexto del problema a resolver.

Se experimentó con diferentes procedimientos, teniendo el mejor resultado al agregar una capa de dropout. Esto tiene bastante sentido debido a la manera que opera esta técnica, ya que ofrece la posibilidad de explorar múltiples arquitecturas de la red neuronal y al final promediar todos los escenarios construidos durante el entrenamiento, dando como resultado pesos ponderados en los axiomas de la red. Sin embargo, esto también fue posible ya que nuestra red no tiene mucha profundidad, permitiendo así tener mayor control sobre los resultados esperados.

En general, se lograron obtener buenos resultados, pero sería interesante poner a prueba una red neuronal con datos más complejos, ya que no sólo aumentaría la complejidad del diseño del modelado predictivo, sino que también se tendrían que implementar más técnicas de preprocesamiento de audio digital. Otro punto interesante a desarrollar a futuro son los aplicativos de esta propuesta, ya que se podrían ofertar productos que mejoren la calidad de vida de los habitantes en una urbe.

Bibliografía

[1] R. A. et al., "Sound classification and processing of urban environments: A systematic literature review," *Sensors*, vol. 22, no. 22, 2022.

[2] A. Bansal and K. N., "Environmental sound classification: A descriptive review of the literature," *Intelligent Systems with Applications*, vol. 16, 2022.

[3] F. H. Al-Hattab, Y. and A. Shafie, "Rethinking environmental sound classification using convolutional neural networks: optimized parameter tuning of single feature extraction," *Neural Computing and Applications volume*, vol. 33, 2021.

[4] J. C. . B. J. P. Salamon, J., "A dataset and taxonomy for urban sound research," *ACM international conference on Multimedia*, vol. 22, pp. 1041–1044, 2014.

[5] J. Salamon and J. P. Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," *CoRR*, vol. abs/1608.04363, 2016.

[6] D. J. et al., "Environmental sound classification using convolution neural networks with different integrated loss functions," *Expert Syst*, vol. 39, 2021.

[7] M. J. and S. Su, "Efficient classification of environmental sounds through multiple features aggregation and data enhancement techniques for spectrogram images," *Symmetry*, vol. 12, no. 1822, 2020.

[8] J. Valenzuela, *Audio Digital: Conceptos Básicos y Aplicaciones*. Santa Monica, CA.: Backbeat Books, 1996.

[9] M. G. Christensen, *Introduction to Audio Processing*. Switzerland: Springer, 2019.

[10] "How to apply machine learning and deep learning methods to audio analysis." https://medium.com/comet-ml. Accessed: 2023-03-02.

[11] "The data processing studio." https://www.cygres.com/OcnPageE/Glosry/SpecE.html. Accessed: 2023-03-03.

[12] S. W. Smith, *The Scientist and Engineer's Guide to Digital Signal Processing*. San Diego, CA.: California Technical Publishing, 1998.

[13] S. T. Kong, Q. and A. M. Bayen, *Python Programming and Numerical Methods*. Berkeley, CA.: Academic Press, 2020.

[14] "Introduction to speech processing." https://wiki.aalto.fi/display/ITSP/. Accessed: 2023-03-19.

[15] "The mel spectrogram." https://medium.com/analytics-vidhya/understanding-the-mel-spectrogram-fca2afa2ce53. Accessed: 2023-03-03.

[16] "The essential guide to data augmentation in deep learning." https://www.v7labs.com/blog/data-augmentation-guide. Accessed: 2023-03-26.

[17] S. Yang, W. Xiao, M. Zhang, S. Guo, J. Zhao, and F. Shen, "Image data augmentation for deep learning: A survey," 2022.

[18] V.-V. Eklund, "Data augmentation techniques for robust audio analysis," Master's thesis, Tampere, Finland, 2019.

[19] S. Haykin, *Neural Networks and Learning Machines*. Hoboken, New Jersey: Prentice-Hall, 1999.

[20] L. Fausett, *Fundamentals of Neural Networks*. Hoboken, New Jersey: Prentice-Hall, 1994.

[21] H. e. a. Ramchoun, "Multilayer perceptron: Architecture optimization and training," *International Journal of Interactive Multimedia and Artificial Intelligence*, vol. 4, no. 1, 2016.

[22] B. Y. Goodfellow I. and C. A., *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[23] "Backpropagation in a nutshell everything you need to know." https://towardsdatascience.com/. Accessed: 2023-04-01.

[24] "Complete guide to adam optimization." https://towardsdatascience.com/complete-guide-to-adam-optimization-1e5f29532c3d. Accessed: 2023-04-09.

[25] M. Hossin and M. Sulaiman, "A review on evaluation metrics for data classification evaluations," *IJDKP*, vol. 5, no. 2, 2015.

[26] "Evaluation metrics." https://spark.apache.org/docs/latest/mllib-evaluation-metrics.html. Accessed: 2023-04-13.

[27] "Creative commons attribution noncommercial license." http://creativecommons.org/licenses/by-nc/3.0/. Accessed: 2023-03-05.

[28] "Musical data augmentation." https://muda.readthedocs.io/en/stable. Accessed: 2023-03-16.

[29] T. e. a. Bäckström, *Introduction to Speech Processing*. 2 ed., 2022.

[30] G. Zigelboim and I. Shallom, "A comparison study of cepstral analysis with applications to speech recognition," *International Conference on Information Technology: Research and Education*, 2006.

[31] "Keras api reference." https://keras.io/api/. Accessed: 2023-04-01.

[32] "How do determine the number of layers and neurons in the hidden layer?." https://medium.com/geekculture/introduction-to-neural-network-2f8b8221fbd3. Accessed: 2023-04-03.

[33] "An overview of regularization techniques in deep learning (with python code)." https://www.analyticsvidhya.com/blog/2018/04/. Accessed: 2023-04-06.

[34] "Adam." https://optimization.cbe.cornell.edu/index.php?title=Adam. Accessed: 2023-04-04.

[35] J. Kasche and F. Nordström, "Regularization methods in neural networks," tesis de licenciatura, Uppsala, Suecia, 2020.

[36] "Chapter 7: Regularization (part 2)." https://shanzhenren.github.io/csci-699-replnlp-2019fall/. Accessed: 2023-04-09.