UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

POSGRADO EN CIENCIA E INGENIERÍA DE LA COMPUTACIÓN
INSTITUTO DE INVESTIGACIONES EN MATEMÁTICAS APLICADAS Y EN SISTEMAS
SEÑALES IMÁGENES Y AMBIENTES VIRTUALES

IDENTIFICACIÓN DE LOCUTOR PARA UN ROBOT DE SERVICIO UTILIZANDO APRENDIZAJE PROFUNDO

TESIS
Que para optar por el grado de:
Maestra en Ciencia e Ingeniería de la Computación

Presenta:
IVETTE JULIANA VÉLEZ TOBÓN

Director de Tesis:
Dr. CALEB ANTONIO RASCÓN ESTEBANÉ
POSGRADOENCIENCIAEINGENIER•IADELACOMPUTACI•ON

Ciudad de México, Julio de 2018

Resumen

Durante este trabajo se realizó la implementación de un sistema de identificación de locutor que no requiere reentrenar un nuevo modelo con nuevos locutores, basado en verificación utilizando aprendizaje profundo.

Para la realización de este trabajo se propuso un sistema que funciona independientemente del texto, es decir que funciona con cualquier tipo de contenido hablado. Para entrenar un modelo con estas condiciones se utilizó la base de datos LibriSpeech, la cual contiene más de 2000 locutores y es independiente del texto.

Se propusieron dos arquitecturas convolucionales tipo siamés para la verificación automática de locutor: una conocida como Módulo VGG y que fue inspirada en la arquitectura VGG 16, y la otra conocida como ResNet 50 siamés y que fue inspirada en la ResNet 50.

Estos modelos utilizaron como entrada contenido de audio en forma de transformada de Furrier o de espectrogramas de frecuencias, que es una forma de representar audio como imágenes de un sólo canal con información tanto espectral como temporal.

Para realizar las pruebas de verificación del sistema utilizando el modelo entrenado y una base de datos que se puede actualizar, se hicieron pruebas con la sección de datos de prueba de LibriSpeech y con la base de datos SITW que contiene grabaciones de diversos locutores en diferentes condiciones ambientales.

Se realizó el entrenamiento de 21 modelos con diferentes tipos de parámetros y arquitecturas y de éstos se seleccionaron los 3 mejores modelos y se realizaron diversas evaluaciones tanto para evaluación, como para identificación con diferentes parámetros.

Los resultados con la base de datos de LibriSpeech evidencian un desempeño de exactitud hasta del 100 % con el módulo VGG utilizando 32 frecuencias. Con la base de datos SITW (que tiene ambientes de grabación muy diferentes a los ambientes de LibriSpeech) se llega a una exactitud del 65.5 % con la ResNet 1 y una exactitud del 62.7 % con el módulo VGG utilizando 32 frecuencias.

Los resultados de este trabajo permiten concluir que sí es posible implementar un sistema de identificación de locutor mediante verificación que no requiere ser reentrenado con nuevos locutores.

Dedicatoria

A mis padres, hermanas, amigos y a ti mi amor, Ale, gracias a todos. *We did it*. Banda sonora de este momento: *Happy*.

Agradecimientos

Agradezco de manera especial al programa de Posgrado en Ciencia e Ingeniería de la Computación y al personal administrativo, por haberme brindado un apoyo incondicional durante mi tiempo como estudiante de maestría. Deseo agradecer al grupo Golem por haberme hecho partícipe de los proyectos asociados al robot de servicio Golem III para quien está destinada inicialmente la aplicación final de este proyecto.

Agradezco a mis profesores, por su conocimiento, en especial al Dr. Caleb Antonio Rascón Estebané quien no sólo fue un grandioso profesor, sino el director de mi trabajo de tesis, agradezco por su apoyo incondicional, orientación, disponibilidad y consejo en todos los momentos de esta tesis; y al Dr. Gibrán Fuentes Pineda, por su apoyo y presencia durante todo este proyecto, por los conocimientos en aprendizaje de máquina y estar siempre disponible para brindarme las herramientas necesarias para concluir esta investigación.

Agradezco al Consejo Nacional de Ciencia y Tecnología (Conacyt) por el financiamiento de programas de becas en los que está incluido el Posgrado en Ciencia e Ingeniería de la Computación y al que pude acceder durante mi maestría.

Agradezco a mis sinodales por haber aceptado la revisión de este proyecto, y por sus comentarios y sugerencias para este trabajo de tesis.

Agradezco a mis padres y hermanas por su apoyo incondicional y estar siempre ahí para mí, siempre para escuchar, siempre para conversar y quienes se alegran tanto como yo por todos mis aprendizajes.

Agradezco a Alejandro Maldonado Navarro a quien conocí en esta maestría y con quien compartiré mi vida, gracias por tu apoyo, buenas ideas, comentarios, sugerencias, este trabajo también es tuyo.

Agradezco a Dios, por estar viva, por estar aquí.

Índice general

Índice de tablas

Índice de figuras

Capítulo 1

Introducción

El reconocimiento automático de voz ha sido un campo de estudio por décadas, y se ha considerado como un puente para mejorar la comunicación humano-humano y humano-máquina. En el pasado la comunicación humano-máquina mediante audio estaba limitada especialmente por las capacidades computacionales (procesamiento y almacenamiento) que se tenía en la época. Sin embargo con el avance tecnológico en diferentes áreas y la introducción de los dispositivos móviles, el habla se convirtió en una de las modalidades de interacción más fuerte con diferentes herramientas tecnológicas como los teléfonos inteligentes o los sistemas de reconocimiento de voz de las casas inteligentes.

Ahora bien la interacción entre humanos y máquinas ya no sólo se limita a reconocer correctamente las palabras que son pronunciadas, sino que esta interacción se ha vuelto aún más compleja. Se busca que las máquinas puedan tener una interacción con los humanos similar a la que tendría un humano con otro humano, esto implica la identificación correcta del locutor con el cual se está interactuando.

Los robots de servicio se encargan de realizar tareas que implican la interacción en tiempo real con un humano, esta interacción debe realizarse de la forma más natural posible. El proceso de realizar sus tareas implica un conjunto de acciones entre estas podría estar el reconocer a la persona con la que se interactúa. Los sistemas para identificar personas pueden hacer uso de imágenes o de señales de voz. En este trabajo se busca realizar la identificación automática de personas para un robot de servicio utilizando la voz del locutor.

Los sistemas actuales de identificación de locutor realizan un proceso de reentrenamiento cada vez que conocen a un nuevo locutor, lo que implica más tiempo. Esto puede traer como consecuencia que la interacción humano-robot no sea natural. En este trabajo se propone una alternativa que no implica el reentrenamiento con nuevos locutores.

El objetivo final de este trabajo es crear un sistema de identificación de locutor para un robot de servicio, que permita ampliar el estado del arte en sistemas de verificación de locutor con máquinas de aprendizaje profundo, el cual es aún limitado. Y se busca complementar la descripción de la escena auditiva para el beneficio de la interacción humano-robot.

De acuerdo con [Campbell, 1997] existen dos formas de aprender a verificar un locutor. La primera es utilizando textos dependientes, cuyas frases o palabras posibles están restringidas a un vocabulario pequeño. La segunda es utilizando textos independientes de la expresión hablada, cuya forma de interactuar es más flexible, pero con una menor precisión de clasificación. Este trabajo busca identificar al locutor independientemente de la frase hablada.

Para llevar a cabo este proyecto se propone el uso de un modelo de red convolucional profunda siamés de verificación. El entrenamiento se realizó utilizando una base de datos independiente del texto. Una vez entrenado el modelo, fue posible realizar la verificación de locutores, esta consiste en comparar los locutores conocidos que se tienen en una base de datos, con una nueva señal. Cuando se agregaron nuevos locutores no se requirió reentrenamiento.

Los resultados obtenidos muestran que es posible tener una exactitud para identificación superior al 94 % con una base de datos con ambiente de grabación constante y de hasta el 65 % con una base de datos con diferentes ambientes de grabación.

Este documento está ordenado de la siguiente forma: en el capítulo 2 se exponen los conceptos básicos de audio necesarios para este trabajo. En el capítulo 3 se explican los conceptos básicos sobre redes neuronales profundas y las arquitecturas que son pertinentes para este trabajo. En el capítulo 4 se habla de las diferencias entre la verificación de locutor y la clasificación de locutor, además se describen brevemente algunos modelos que han sido utilizados para la verificación de locutor. En el capítulo 5 se da una explicación de la metodología utilizada para realizar los experimentos. En el capítulo 6 se muestran los resultados y la discusión de los mismos. Y por último en el capítulo 7 se evidencian las conclusiones y los trabajos futuros propuestos a partir de este trabajo.

1.1. Objetivos

1.1.1. Objetivo general

El objetivo general de este proyecto es:

- Desarrollar un sistema de identificación de locutor que no requiera reentrenar un nuevo modelo con nuevos locutores, basado en verificación utilizando aprendizaje profundo.

1.1.2. Objetivos específicos

Los objetivos específicos para este proyecto son:

- Entrenar un modelo de verificación de locutor utilizando una red convolucional profunda haciendo uso de una base de datos con ambiente de grabación constante para la la identificación automática de locutor.
- Realizar la identificación de locutores con el sistema entrenado utilizando una sección de datos de prueba una base de datos con ambiente de grabación constante.
- Realizar la identificación de locutores con el sistema entrenado utilizando una base de datos grabada en diferentes ambientes.
- Evaluar la precisión de los sistemas tanto como verificadores como clasificadores.

1.2. Presentación del problema

Los identificadores de locutores, actualmente presentan una limitación para poder ser implementados en robots de servicio y que funcionen en línea, debido a que se requiere reentrenar los sistemas cuando llega un nuevo locutor. Esto implica tiempos de los cuales no dispone el robot, ya que se esperan reacciones inmediatas.

El problema que se desea abordar en esta tesis es el de implementar un sistema de identificación de locutor que sea rápido para aprender nuevos locutores, ya que el uso de los sistemas tradicionales y propuestos hasta ahora, no son aptos para ser implementados sobre un robot de servicio, por el tiempo de reentrenamiento que requerirían con nuevos locutores.

1.3. Motivación

La motivación principal de este trabajo, es que si existiera un identificador de locutor que funcionara de forma rápida y natural en los robots de servicio, la interacción humano-robot, se podría asemejar más a la interacción humano-humano. Para llevar a cabo esta idea se desea utilizar un modelo que no requiera reentrenamiento con nuevos locutores.

Se conoce que las redes siameses puede realizar la verificación de imágenes, también que se han implementado sistemas para identificación de caracteres escritos que no requieren reentrenamiento con estas, por lo que se busca realizar un sistema de identificación de locutor utilizando verificación con redes siameses usando audio o alguna de sus representaciones como son los espectrogramas de frecuencia.

1.4. Hipótesis

Se puede crear un sistema para la identificación de locutor basado en una red siamés que verifique la similitud de dos espectrogramas generados de dos señales de audio con una exactitud de identificación superior al 80 % y con un tiempo de respuesta inferior a un segundo, para un robot de servicio.

Mas detalladamente, la hipótesis propone que se puede crear un sistema para la identificación de locutor basado en verificación que no requiera reentrenarse con nuevos locutores.

Haciendo uso de una red siamés se puede aprender a determinar la similitud de dos espectrogramas generados a partir de dos señales de audio, de manera que cuando se tenga un audio de un locutor conocido, y haya un audio de una persona a identificar, se comparen las dos señales y se determine si es o no el mismo locutor.

Se cree que un sistema de este tipo puede tener una exactitud de identificación superior al 80 % que es el estándar actual de verificación para sistemas que reentrenan con cada nuevo locutor, para una base de datos en condiciones de grabación constantes, y con una respuesta inferior a segundo que es lo máximo que espera un locutor que se tarde un robot de servicio en reconocerlo.

Todo esto está enmarcado en el contexto de robot de servicio, por lo que se espera que la interacción se realice con una cantidad limitada de locutores.

Capítulo 2

Audio

La comunicación mediante audio ha sido una herramienta utilizada entre los humanos desde hace muchos años, esta comunicación poco a poco se ha convertido en un puente importante no sólo para los humanos, sino también para la interacción humano-máquina. En el pasado, una de las razones por las cuales no fue explotada este tipo de comunicación, es que los sistemas computacionales no eran los suficientemente robustos para que pasaran la barrera de usabilidad, además de que otros sistemas de comunicación como teclados y ratones eran mucho más eficientes [Yu and Deng, 2015]. Sin embargo los avances tecnológicos han potenciado los métodos alternativos de comunicación y los sistemas computacionales ya no son la barrera que limita esta forma de interacción.

Los avances tecnológicos y la introducción de los dispositivos móviles en la vida cotidiana, cambiaron la forma en la que se realiza la comunicación humano-máquina, siendo el reconocimiento de voz una de las principales formas de interacción [Yu and Deng, 2015]. Esta interacción ha sido posible gracias a la potencia computacional disponible en la actualidad y el acceso a datos masivos recopilados a través de años, lo que ha propiciado un gran avance en los reconocedores de voz actuales. Un ejemplo de este tipo de datos se puede observar en las bases de datos disponibles para reconocimiento de voz y de locutor, en el pasado bases de datos como TIMIT [John S. Garofolo, 1993] que sólo estaba compuesta por una cantidad limitada de frases dichas por locutores en un entorno controlado, han evolucionado a bases de datos como la de Voxceleb [Nagrani et al., 2017] donde los datos se recopilan directamente de videos de youtube, en muchos tipos de ambientes y sin limitantes en el tipo de texto a utilizar.

Para poder realizar el reconocimiento automático de voz, o en este caso la identificación de un locutor a través de verificación, es necesario definir el tipo de señal con la que se va

a trabajar, qué es una señal de audio, qué tipos de características tiene y cómo se pueden representar.

En este capítulo se definen los conceptos de sonido, señal de audio y señal digital de audio. Además se introducen los temas de transformada de Fourier, espectrograma y características de audio como otras formas de representar y caracterizar señales auditivas. Particularmente en el tema de características se abordan aquellas que, de acuerdo con la revisión de literatura, han sido utilizadas para verificar y clasificar señales de audio y adicionalmente se nombran cuáles son sus ventajas y desventajas.

2.1. Conceptos básicos

2.1.1. Sonido

El sonido audible puede ser definido como "una perturbación en la atmósfera " [Western Electric CO., 1969]. Esta perturbación genera un movimiento de onda que se propaga desde la fuente a una velocidad de 1 075 pies por segundo que equivale a 343 metros por segundo. La onda propagada audible para un humano comprende las frecuencias de 20 a 20 000 ciclos por segundo o *Hz*.

Estos movimientos de onda u ondas sonoras, "son producidas por la vibración de alguna fuente, como un diapasón o las cuerdas vocales humanas" [Western Electric CO., 1969]. El nivel de vibración que tenga esta fuente determina la frecuencia del sonido y el tono. La frecuencia del sonido viene dada por la velocidad de vibración de la fuente, si la fuente vibra rápidamente genera un tono alto, mientras que si vibra lentamente produce un tono bajo.

2.1.2. Señal de audio

Las señales de audio se pueden describir como "una función del nivel de presión sonora variable en el tiempo" [Lerch, 2012]. Ahora bien para convertir este nivel de presión en voltaje se debe hacer uso de algún tipo de herramienta de grabación de sonidos que pueda capturar el tipo de frecuencias con las que se desea trabajar, como por ejemplo un micrófono. Se debe de tener claro el tipo de grabación que se desea llevar a cabo, por ejemplo si se desea grabar sonidos de alta frecuencia, como la emisión sonora de los murciélagos, se requiere de equipo de grabación que pueda capturar frecuencias superiores e inferiores al rango de $20Hz$ a $20kHz$. Si se desea grabar voz, el micrófono a utilizar debe capturar frecuencias entre $20Hz$

y $20kHz$. Las señales de audio están definidas para todos los tiempos y por lo tanto son señales continuas en el tiempo.

Una señal de audio también puede definirse como una señal continua cuyos voltajes son exactamente iguales a los de una señal sonora y cuyas frecuencias suelen estar en el rango audible para los humanos de 20 a $20,000Hz$.

2.1.3. Señal digital de audio

En el dominio digital no se puede tratar con señales continuas. Por lo tanto, una señal análoga y continua de audio debe discretizarse tanto en amplitud como en tiempo, la discretización de amplitudes se conoce como cuantificación y la discretización en tiempo se conoce como muestreo, estos dos conceptos se explican brevemente en las secciones 2.1.3.1 y 2.1.3.2. La señal resultante es una señal de audio digital con una serie de valores de amplitud cuantificados [Lerch, 2012].

2.1.3.1. Muestreo

El muestreo se refiere a la discretización en momentos específicos del tiempo de una señal [Lerch, 2012]. Para las señales de audio la distancia entre los puntos en los cuales se va a discretizar la señal es igual, es decir esta distancia T_s en segundos entre los puntos del muestreo es equidistante en el contexto de las señales de audio y puede derivarse directamente de la frecuencia de muestreo F_s por la ecuación 2.1.

$$T_s = 1/F_s \tag{2.1}$$

La señal sólo se puede reconstruir sin ambigüedades cuando ciertos requisitos para la señal de audio y las frecuencias de muestreo se cumplen según lo definido por el teorema de muestreo (Teorema 1).

Teorema 1 (Teorema de muestreo)**.** *"Una señal muestreada solo puede reconstruirse sin pérdida de información si la frecuencia de muestreo es mayor que el doble de la frecuencia más alta en la señal de audio muestreada" [Lerch, 2012] (ecuación 2.2).*

$$F_s > 2F_{max} \tag{2.2}$$

2.1.3.2. Cuantificación

La cuantificación de una señal de audio se refiere a discretización de las amplitudes de la señal [Lerch, 2012]. La cuantificación significa que cada amplitud de la señal se redondea a un conjunto predefinido de valores de amplitud permitidos. Normalmente, el rango de amplitud de entrada entre la amplitud máxima y la mínima se supone que es simétrica alrededor de cero y se divide en M pasos.

2.1.4. Huellas dactilares de audio

Los sonidos pueden ser representados por características que son conocidas como huellas dactilares de audio (*audio fingerprints*), estas huellas "hacen referencia a las firmas compactas (extraídas de una señal de audio)" [Malekesmaeili and Ward, 2014], las cuales permiten distinguir entre diferentes sonidos como son canciones, voces u otro tipo de sonidos audibles.

En la sección 2.4 se habla de algunas características de audio que han sido utilizadas para la identificación y verificación de locutor, éstas tienen ventajas y desventajas, y en este trabajo se busca que la extracción de características o la huella dactilar de audio sea extraída por la propia red convolucional, en vez de hacer uso de características artesanales de audio.

2.1.5. La naturaleza del habla

El habla, al igual que cualquier otra señal auditiva, es una secuencia de ondas que se transmiten a lo largo del tiempo a través de un medio y se caracteriza por algunas propiedades, incluida la intensidad y la frecuencia [Kasabov, 1996]. El habla es percibida por el oído interno en los humanos, activando las oscilaciones de pequeños elementos en los medios del oído interno, cuyas oscilaciones se transmiten a una parte específica del cerebro para su posterior procesamiento. Algunos investigadores utilizan los antecedentes biológicos para desarrollar sistemas de reconocimiento automático del habla, o para extraer las características de la voz con las cuales los humanos suelen realizar la identificación de locutores, como por ejemplo el tracto vocal, los sonidos nasales generados por los humanos, entre otros. Pero otros investigadores toman otros enfoques como el análisis frecuencial del sonido más allá de la forma biológica como es generada.

Ahora bien existen diferentes formas de representar el audio, algunas de las más comunes son:

- Escala de tiempo, cuya representación se denomina representación de forma de onda.

- Escala de frecuencia, cuya representación se llama espectro.
- Tanto en una escala de tiempo como de frecuencia, que es el espectrograma de la señal de voz.

Otras formas de representar el audio son los MFCC y MFB descritos en la sección 2.4, o los espectrogramas de MFCC y MFB, sin embargo en este trabajo no se utilizan este tipo de representaciones.

2.1.6. Variabilidad en el habla

El habla es una señal con una alta variabilidad y es afectada por la frecuencia de conversación, el contexto y las condiciones acústicas [Lee et al., 1993]. La variabilidad del habla no sólo depende de los factores biológicos como el tamaño del tracto vocal, la disposición de las cuerdas vocales, la edad y el sexo entre otros factores, sino que también se relaciona con la pronunciación de las palabras de acuerdo con su contexto e incluso el lugar de aparición en una oración [Kasabov, 1996]. También ser puede relacionar con el país de origen, la región, el acento del idioma que se habla e incluso si el idioma a reconocer no es la lengua materna del hablante. El mismo hablante puede mostrar variabilidad en su discurso dependiendo de si se trata de una situación formal e informal, de su estado anímico y del idioma que esté utilizando.

La tarea fundamental del reconocimiento de voz es entonces encontrar qué variaciones son relevantes para el reconocimiento de voz.

2.1.7. VAD

Un detector de actividad de voz (*VAD*) o detector de actividad hablada (*SAD*), busca localizar los segmentos de voz de una señal de audio determinada [Benyassine et al., 1997]. Las señales de voz a pesar de ser continuas no tienen contenido representativo en cada instante del tiempo, por ejemplo los locutores en idioma español realizan una pequeña pausa después del signo de puntuación coma, o una pausa ligeramente más larga tras el signo de puntación punto, estos pequeños trozos de silencio no contienen actividad de voz real para realizar el reconocimiento de la identidad de un locutor. Un buen *VAD* evita todos los trozos de la voz que no contienen información de voz relevante para la tarea.

El *VAD* tiene diferentes formas de ser abordado dependiendo de la aplicación en la cual es requerido. Se puede utilizar un *VAD* estático para aquellas señales que se conocen de

antemano por completo, es decir para aplicaciones fuera de línea. El *VAD* en este caso funciona como un umbral, donde se supone que los tramos de las señales con valores de energía superiores al umbral contienen actividad de voz. También se puede utilizar un *VAD* dinámico o adaptativo para aquellas señales que cambian a través del tiempo. Normalmente este tipo de *VAD* se usa para aplicaciones en línea, en las cuales el ambiente puede variar constantemente.

2.2. Transformada de Fourier

Las señales auditivas pueden ser representas en el dominio de la frecuencia o haciendo uso de los espectrogramas de frecuencia. Para llevar a cabo este tipo de representaciones es necesario realizar una transformación del dominio del tiempo al dominio de la frecuencia, tal transformación requiere hacer uso de la transformada de Fourier.

El análisis de Fourier busca representar una señal como la suma de funciones trigonométricas, o de una forma más general, como la suma de series de funciones periódicas más simples [Anand, 2010]. Series muy similares a la series de Fourier fueron utilizadas por d'Alembert, Euler, Bernoulli y Gauss, los cuales propusieron el uso de series trigonométricas, sin embargo en sus aplicaciones el periodo de la función a representar era conocido. Fue hasta 1807 que Fourier propuso que funciones arbitrarias también podían ser representadas por series trigonométricas.

2.2.1. Transformada rápida de Fourier

La transformación de una señal en el dominio del tiempo al dominio de la frecuencia puede realizarse utilizando la transformada de Fourier la cual hace uso de la serie de Fourier que está definida en 2.2.1, ésta representa una función periódica como la suma de oscilaciones con frecuencias espaciadas a una razón fija y amplitudes complejas [Anand, 2010].

Definición 2.2.1. Serie de Fourier La serie de Fourier de una función f, por partes continua de $[-P, P]$ y teniendo un periodo $2P$ está definida como:

$$f(t) = \sum_{i=-\infty}^{\infty} c_n e^{in\frac{\pi}{P}t} \tag{2.3}$$

donde:

$$c_n = \frac{1}{2P}\int_{-P}^{P} f(t)e^{-in\frac{\pi}{P}t}dt \tag{2.4}$$

La serie de Fourier representa la función periódica f como una suma de oscilaciones con frecuencias $\frac{n\pi}{P}$ y amplitudes complejas c_n. Así es posible tener una expresión integral que represente cualquier función f, como una suma de funciones periódicas. Se define:

$$\hat{f}(P,\omega) = \int_{-P}^{P} f(t)e^{-i\omega t}dt \tag{2.5}$$

De la ecuación 2.4, $c_n = \frac{1}{2P}\hat{f}(P, \frac{n\pi}{P})$, y la ecuación 2.3 puede ser escrita como:

$$f(t) = \frac{1}{2\pi}\sum_{i=-\infty}^{\infty} \hat{f}(P,\omega_n)e^{i\omega_n t}\frac{\pi}{P}, \omega_n = \frac{n\pi}{P} \tag{2.6}$$

Dado que $\Delta\omega_n = \omega_{n+1} - \omega_n = \frac{\pi}{P}$. Cuando $P \to \infty$ y definiendo

$$\hat{f}(\omega) = \lim_{P\to\infty} \hat{f}(P,\omega) = \int_{-\infty}^{\infty} f(t)e^{-i\omega t}dt, \omega \in \mathbb{R} \tag{2.7}$$

Del aplicar el límite se obtiene que:

$$f(t) \sim \frac{1}{2\pi}\int_{-\infty}^{\infty} \hat{f}(\omega)e^{i\omega t}d\omega \tag{2.8}$$

Ahora bien, la señal de audio al ser transformada no es continua, sino discreta ya que ha pasado por un proceso de muestreo y cuantificación, por lo que se requiere el uso de la transformada de Fourier discreta (DFT), que es "el mapeo espectral más simple de la transformada de Fourier realizada en el dominio digital" [O'Shaughnessy, 2008]. La DFT es una aproximación de la transformada de Fourier continua (CFT) para el caso de funciones discretas. La CFT se define en 2.2.2 mientras que la DFT se define en 2.2.3

Definición 2.2.2. Transformada de Fourier Continua Para una función f en $\mathbb{R}$ tal como en 2.9

$$\int_{-\infty}^{\infty} |f(t)|dt < \infty \tag{2.9}$$

La integral $\hat{f}(\omega)$ que se define en 2.10 converge absolutamente y se define como la transformada de Fourier continua o integral de Fourier de f.

$$\hat{f}(\omega) = \int_{\mathbb{R}} f(t)e^{-i\omega t}dt \tag{2.10}$$

Definición 2.2.3. Transformada de Fourier Discreta La transformada de Fourier Discreta (DFT) para una señal x se puede definir como en 2.11

$$X(\omega_k) = \sum_{n=0}^{N-1} x(t_n)e^{-i\omega_k t_n}, k = 0, 1, ..., N-1 \tag{2.11}$$

La construcción de la DFT usa el hecho de que los términos sinusoidales en la DFT forman una base ortogonal en el espacio $\mathbb{C}^n$.

Sustituyendo $t_n = nT, \omega_k = \frac{2\pi k}{NT}$, se llega a una forma más común de la DFT, como se observa en la ecuación 2.12.

$$X(\omega_k) = \sum_{n=0}^{N-1} x(n)e^{-2\pi i \frac{nk}{N}}, k = 0, 1, ..., N-1 \tag{2.12}$$

Para realizar de manera eficiente el cálculo de la transformada discreta de Fourier, se hace uso de la transformada rápida de Fourier, que elimina operaciones redundantes en su cálculo.

La transformada rápida de Fourier (FFT) se ha convertido en un algoritmo muy eficiente para calcular la transformada discreta Fourier (DFT) [Heideman et al., 1985]. En 1965 J. W. Cooley y J. W. Tukey publicaron su algoritmo de la FFT como un medio para calcular la DFT, el nuevo algoritmo FFT podía calcular la DFT usando un número de operaciones proporcionales a $NlogN$, mientras que el cálculo tradicional de la DFT requería operaciones aritméticas N^2.

El algoritmo de Cooley y Tukey sigue siendo el más comúnmente utilizado aunque a Gauss se le atribuye la primera implementación de la Transformada Rápida de Fourier (FFT) en 1805, durante una interpolación de mediciones orbitales [Anand, 2010].

2.3. Espectrograma

Otra forma de representar una señal de audio es mediante el uso de espectrogramas de frecuencia, en el cual las propiedades acústicas de los eventos de sonido se pueden visualizar en una "imagen" que está representada tanto en frecuencia como en tiempo [Prasad, 2009].

El uso de espectrogramas se remonta hasta el año 1968 donde [Asano et al., 1968] desarrollaron un proceso automatizado llamado "análisis de ventana en movimiento", el cual

produce una visualización de amplitudes y/o fases como funciones del periodo y la velocidad de grupo de ondas a analizar. Estos resultados se calcularon mediante el análisis de Fourier de porciones de ventana adecuadas para sismogramas.

Para generar la imagen anteriormente descrita, se computan espectros de la transformada de Fourier en segmentos de ventana sobrelapados en intervalos pequeños sucesivos. El análisis espectral que se obtiene mediante la transformada de Fourier produce una función de frecuencia y un espectro de fase. Sin embargo "el espectro de fase no es perceptualmente significativo" [Prasad, 2009]. Así, a partir del espectro de magnitud se produce una visualización gráfica del contenido de frecuencia y tiempo de la señal o espectrograma.

Un ejemplo gráfico de cómo se genera un espectrograma de frecuencia se muestra en la figura 2.1, en la figura se tiene una señal sinusoidal de $100ms$ con una frecuencia de $100Hz$, de esta se toma una ventana de $50ms$ a la cual se le aplica la transformada de Fourier, esta transformación representa un pixel del espectrograma de frecuencia que se va a generar, los valores de la transformada indican la intensidad del color en el espectrograma. Después se toma otra ventana de $50ms$ para generar otro pixel y así sucesivamente, hasta recorrer toda la señal de audio. La parte compartida entre las ventanas se conoce como traslape o sobrelape y la cantidad de muestras o de tiempo que se tiene entre una ventana y otra se conoce en este trabajo como paso.

El espectrograma muestra propiedades espectro-temporales de los eventos acústicos [Prasad, 2009]. La duración de la ventana con la cual se crea el espectrograma, dice el compromiso entre la resolución frecuencial y la resolución temporal de eventos o transiciones que pueden variar rápidamente en el tiempo. Una ventana de alta duración permite realizar una transformación frecuencial con mayor número de puntos, con lo que se puede tener mayor información en frecuencia acerca sonido. Una ventana con baja duración permite ver los cambios frecuenciales a través del tiempo con mayor precisión. Este compromiso dependerá de la necesidad frecuencial o temporal de la aplicación a desarrollar.

2.4. Características de audio

El audio puede ser representado mediante diversas características. A continuación se nombran brevemente algunas características y de qué forma se utilizan para la verificación e identificación de locutor. Esta sección se relaciona con la sección 4.3 donde se detalla que métodos, arquitecturas y características han sido utilizadas para el reconocimiento de locutores.

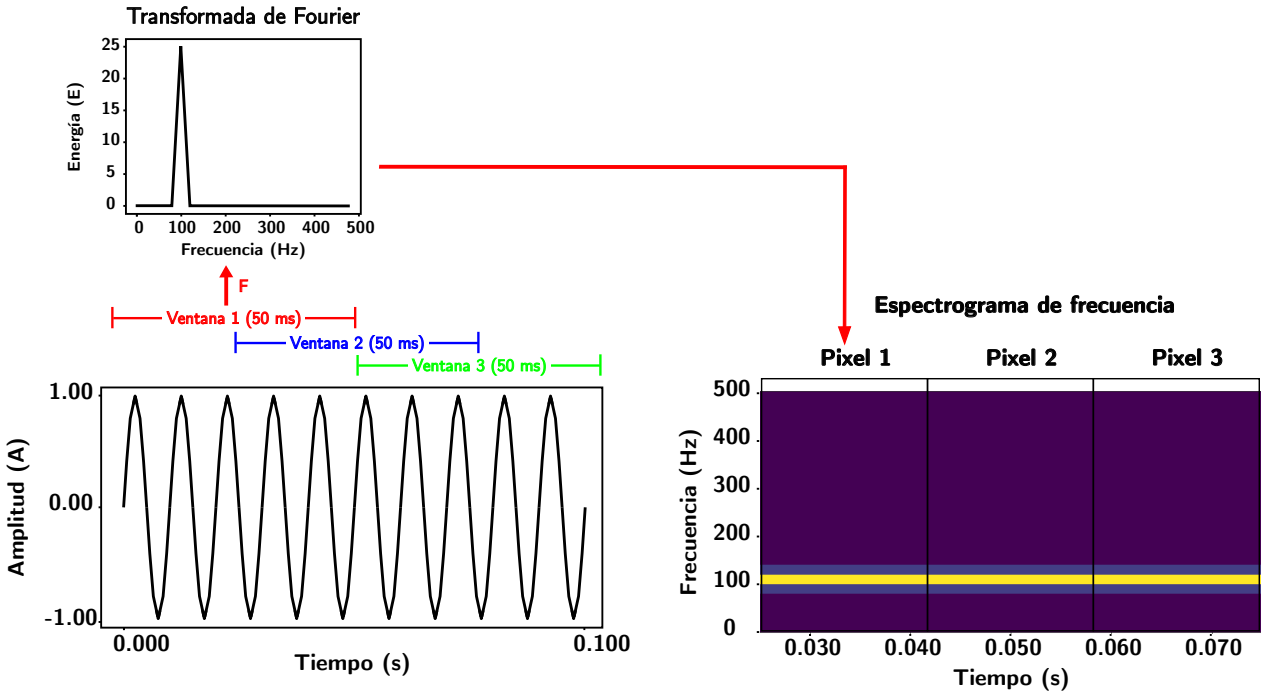

Figura 2.1: Diagrama que representa la generación de un espectrograma de frecuencia.

2.4.1. *MFCC* y *MBF*

Dos métodos comunes para el reconocimiento automático de voz es el uso de los coeficientes cepstrales de frecuencia de Mel (*MFCC*) y el uso de los Bancos de filtros de Mel (*MFB*) [O'Shaughnessy, 2008]. Para calcularlos primero se obtiene un espectro de FFT para cada ventana de voz, para el cual se toma el logaritmo de la amplitud espectral (convirtiendo a decibelios y descartando la fase espectral), después se aplica un conjunto de filtros triangulares espaciados de acuerdo con los pesos de la escala perceptual de Mel. El resultado de este paso genera los *MFB*. Los valores generados en este paso, dada la naturaleza de los filtros de Mel, están correlacionados. Para los *MFCC* el último paso es aplicar la transformada de vectores espectrales de Mel, la cual decorrelaciona sus componentes. Para el reconocimiento de hablantes esta transformada es aproximada por la transformada discreta del coseno. Éste último paso es lo que diferencia a los *MFCC* de los *MFB*. Los coeficientes generados proveen el vector espectral para la evaluación.

La ventaja de estas características o formas de representar el audio es que no se requiere tomar decisiones difíciles para determinar que características utilizar con respecto a otras técnicas como por ejemplo la búsqueda de formantes o la estimación F0 en las cuales hay riesgo de error.

La desventaja de estos métodos es que si la cantidad de filtros a utilizar es muy baja, la cantidad de información espectral que se puede obtener también es baja ya que sólo se representa en una cantidad fija de coeficientes. En el caso de reconocimiento automático de locutores al agrupar las frecuencias con el uso de filtros se puede perder la firma espectral de la persona. Otra desventaja es que cuando el habla se corrompe por ruido, se llenan los valles espectrales entre los armónicos y los formantes, por esto los *MFCC* y *MFB* se deterioran, este deterioro de los armónicos y los formantes puede hacer perder la firma espectral de los locutores, ya que son los armónicos y los formantes lo que permite diferenciar entre locutores, además de que se degradan las frecuencias principales en las cuales habla el locutor y con las cuales se puede identificar.

2.4.2. *LPC*

Los coeficientes de predicción lineal (LPC) se basan en un modelo lineal de producción del habla [Bimbot et al., 2004]. El modelo generalmente utilizado es un modelo de promedio móvil autoregresivo ($ARMA$) simplificado en un modelo autoregresivo (AR). El aparato de producción del habla se describe como una combinación de 4 módulos: la fuente glótica, el tracto vocal, el tracto nasal y los labios. Cada uno de estos módulos puede representase por un filtro. Los cuatro filtros se combinan para caracterizar la señal creando un filtro global, del cual se determinan los coeficientes del filtro global que serán conocidos como *LPCs*.

El principio del análisis de LPC es estimar los parámetros de un filtro AR en una porción de ventana de una señal de voz. Luego la ventana se mueve y se calcula una nueva estimación. Para cada ventana, se estima un conjunto de coeficientes (llamados coeficientes predictivos o coeficientes LPC). Finalmente, se puede estimar una envolvente de espectro para la ventana actual a partir de los coeficientes predictivos.

2.4.3. Espectrograma

Los espectrogramas de frecuencia, permiten representar el audio como una imagen de un sólo canal que nos proporciona información tanto frecuencial como temporal. Para crear un espectrograma, varios parámetros pueden ser modificados, los cuales permiten evidenciar o resaltar información.

- La cantidad de segundos de audio con los cuales se va a generar el espectrograma. Esto impacta directamente en el ancho de la imagen, a mayor cantidad de segundos, mayor

es el ancho de la imagen.

- La ventana indica la cantidad de segundos sobre los que se va a trazar cada fila de píxeles del espectrograma. Normalmente esta ventana suele ser del orden de los milisegundos (ms), esta ventana requiere el uso de una función matemática cuyos valores no sean cero en un intervalo definido, tal como son las funciones Hann, Hamming, Tuckey entre otras.

- El paso es la cantidad de información que se mueve la ventana para su siguiente evaluación de frecuencias, esto se relaciona con el traslape de la señal, por ejemplo si se usa una ventana de 25 ms y un paso de 10 ms se están traslapando 15 ms de la señal, en algunas implementaciones de espectrogramas de frecuencias se configura el traslape en vez del paso.

- La cantidad de puntos que la FFT va a generar. Esto indica la resolución frecuencial que va a tener el espectrograma, a mayor cantidad de puntos mayor resolución espectral y también mayor será el alto de la imagen.

2.5. *GMM* y *HMM*

Previo a los años 2000, los *GMM* (modelos de mezcla de gaussianas) y *HMM* (modelos ocultos de Markov) fueron muy utilizados para el reconocimiento de locutores, tanto con identificación como con verificación.

La gran desventaja de estos dos métodos es que buscan modelar al locutor, incluso cuando se utilizan para verificación, requieren que el locutor sea conocido por el modelo, para extraer de forma correcta su representación. Esto es indeseable para este trabajo, en el cual se busca que el sistema de identificación de locutor no requiera reentrenar un nuevo modelo con nuevos locutores, por lo que ni los *GMM* ni los *HMM* son aptos para este proyecto.

Otra desventaja de estos modelos es que suelen usar características artesanales como son los *MFCC*, los *MFB* y los *LPC*, cuyas características se degradan rápidamente con ruido de fondo, lo que también es indeseable para este trabajo, ya que se busca que se pueda identificar locutores aún cuando las condiciones ambientales se modifiquen.

Capítulo 3

Aprendizaje profundo

Desde el año 2006 el aprendizaje profundo ha emergido como una nueva área de investigación del aprendizaje automático [Deng and Yu, 2014], [Hinton et al., 2006], [Bengio, 2009], ningún intento exitoso (a excepción de las redes convolucionales) había sido registrado antes de ese año. Usualmente se lograban crear redes de dos o tres capas, entrenar con más capas traía resultados pobres [Bengio, 2009].

Se considera que el progreso comenzó en el año 2006 debido a las publicaciones de Hinton y sus colaboradores [Hinton et al., 2006] donde introducen un algoritmo de entrenamiento *greddy*, el cual entrena una capa a la vez, explotando un aprendizaje no supervisado para cada capa.

En este capítulo se explican brevemente conceptos de aprendizaje automatizado y arquitecturas de redes convolucionales que son relevantes para abordar este trabajo.

3.1. Aprendizaje automático

El aprendizaje automático es una rama de la inteligencia artificial que tiene como objetivo permitir que las máquinas realicen tareas mediante el uso de software inteligente, este software inteligente se basa en métodos estadísticos de aprendizaje. Según [Mohammed et al., 2017] el objetivo del aprendizaje es construir un modelo que tome la entrada y produzca un resultado deseado.

La rama del aprendizaje automático enfrenta tareas que serían complicadas de resolver con programas hechos a la medida diseñados por humanos programadores. “El aprendizaje es nuestro medio para alcanzar la capacidad de realizar la tarea” [Goodfellow et al., 2016],

si necesitamos por ejemplo, que un robot ande en bicicleta, tenemos la opción de enseñarle a pedalear o la opción de especificar manualmente como pedalear.

El aprendizaje automatizado ha sido aplicado exitosamente a diferentes tareas, tales como clasificación, regresión, transcripción, traducción, entre otras. Estas tareas se describen en términos de cómo el sistema de aprendizaje automatizado debe procesar un ejemplo.

Existen muchas tareas que pueden ser abordadas con aprendizaje automático, en este trabajo se aborda la tarea de clasificación. Según la descripción de [Goodfellow et al., 2016], en esta tarea el sistema debe especificar a qué categoría k pertenece una entrada. Para resolver este problema el algoritmo de aprendizaje usualmente produce una función $f : \mathbb{R}^n \rightarrow \{1, ..., k\}$. El modelo asigna una entrada descrita por el vector $\mathbf{x}$ a una categoría identificada por un código numérico y. Un ejemplo de esta tarea de clasificación se observa en los sistemas de reconocimiento de objetos, donde la entrada es una imagen y la salida es un código numérico que identifica el objeto de la imagen.

Los algoritmos de aprendizaje automatizado pueden ser clasificados principalmente en supervisados y no supervisados dependiendo el tipo de experiencia que se le permita tener durante el proceso de aprendizaje [Goodfellow et al., 2016]. Esta experiencia se compone de ejemplos, estos se pueden ver como una colección de características que han sido cuantitativamente medidas de un objeto o un evento, un ejemplo se representa mediante un vector $\mathbf{x} \in \mathbb{R}^{\mathbf{n}}$ donde cada entrada x_i del vector es una característica [Goodfellow et al., 2016].

Los algoritmos son entrenados mediante bases de datos compuestas de cierta cantidad de ejemplos, estos ejemplos pueden o no contener una etiqueta, usualmente estas indican la categoría a la que pertenece el ejemplo (en términos de clasificación). El aprendizaje supervisado requiere que los ejemplos tengan sus etiquetas, mientras que el aprendizaje no supervisado no lo necesita.

El término supervisado se origina desde el punto de vista de que el objetivo y se provee por un instructor, quien muestra al sistema de aprendizaje qué hacer. En el aprendizaje no supervisado no hay instructor y el algoritmo debe aprender a darle sentido a los datos sin esta guía [Goodfellow et al., 2016].

3.1.1. Aprendizaje supervisado

De acuerdo con [Mohammed et al., 2017], el objetivo del aprendizaje supervisado es inferir o función o mapeo de los datos etiquetados de entrenamiento. Para este tipo de entrenamiento se tienen datos completamente etiquetados que constan de una vector de entrada $\mathbf{x}$ y un vector

de salida **y** de etiquetas, donde una etiqueta de **y** corresponde a su respectivo dato de entrada **x** y juntos forman un ejemplo de entrenamiento.

Este tipo de aprendizaje se considera supervisado, ya que se le proveen ejemplos etiquetados para el entrenamiento, es decir, características asociadas a una etiqueta [Goodfellow et al., 2016].

3.1.2. Aprendizaje no supervisado

De acuerdo con [Mohammed et al., 2017], en el aprendizaje no supervisado no se tienen etiquetas para los datos de entrenamiento, es decir que no se tiene un vector de etiquetas **y** para una serie de vectores de entrada **x**. La idea es encontrar una estructura oculta en esta información.

3.2. Red neuronal artificial

Una forma de aprendizaje automático es el uso de redes neuronales.

De acuerdo con [Kasabov, 1996], una red neuronal artificial (ANN) es un modelo computacional inspirado biológicamente que procesa elementos (conocidos como neuronas) a través de algoritmos de entrenamiento, la neuronas se conectan a través de conexiones que tienen coeficientes vinculados y que son conocidas como pesos, esto constituye la estructura neuronal. Estos modelos se conocen como modelos conexionistas debido a las conexiones entre ellos. La "memoria" del sistema está asociada a los pesos que conectan las neuronas.

Siguiendo con la definición de [Kasabov, 1996], el primer modelo matemático de una neurona fue propuesto por McCulloch y Pitts en 1943. Era un dispositivo binario que utilizaba entradas binarias, salida binaria y un umbral de activación fijo. En general, un modelo de neurona artificial se basa en los siguientes parámetros que describen una neurona:

- Conexiones de entrada (o entradas) : $x_1, x_2...x_n$. Hay pesos ligados a las conexiones de entrada: $w_1, w_2...w_n$; una entrada a la neurona llamada sesgo ($bias$) y generalmente se representa como una entrada separada llamada x_0, pero por simplicidad se trata como una entrada sujeta a un valor constante.

- Función de entrada f, calcula la señal de entrada neta agregada a la neurona $u = f(\mathbf{x}, \mathbf{w})$, donde **x** y **w** son los vectores de entrada y pesos correspondientes: f es gene-

ralmente la función suma:

$$u = \sum_{i=1}^{n} x_i \cdot w_i$$

- Una función de activación s calcula el nivel de activación de la neurona $a = s(u)$.
- Una función de salida calcula el valor de la señal de salida emitida a través de la salida de la neurona (el axón): $o = g(a)$; la señal de salida generalmente se supone que es igual al nivel de activación de la neurona, es decir $o = a$.

Los valores de entrada y salida de una neurona pueden ser binarios, 0, 1; bivalentes, - 1, 1; continuos, [0, 1]; o números discretos en un intervalo definido.

Las funciones de activación más usadas son:

- La función umbral estrictamente limitada (*The hard-limited threshold function*). Si el valor de entrada neto u de la neurona está por encima de cierto umbral, la neurona se activa (toma el valor de 1); de lo contrario permanece inactiva (toma el valor de 0).
- La función de umbral lineal (*The linear threshold function*). El valor de la activación aumenta linealmente con el aumento de la señal de entrada u, pero después de un cierto umbral, la salida se satura a un valor.
- La función sigmoidea (*sigmoid*). Esta es cualquier función de transformación no lineal $g(u)$ en forma de S que se caracteriza por lo siguiente:
  - Limitada, es decir sus valores están restringidos entre dos límites.
  - Monótonamente creciente, es decir el valor de $g(u)$ nunca disminuye cuando u aumenta.
  - Continua, por lo tanto es diferenciable en todas partes en su dominio.

  Diferentes tipos de funciones sigmoideas han sido utilizadas en la práctica. La mayoría de ellas se componen de la función logística, de una forma general esta puede ser escrita de la siguiente manera: $a = 1/(1 + e^{-c \cdot u})$, donde c es una constante.

  La razón por la cual la función logística se ha usado como una función de activación neuronal, es que para realizar aprendizaje en redes neuronales muchos algoritmos usan la derivada de la función de activación y ésta tiene una derivada simple $\partial g/\partial u = a(1-a)$. Las alternativas a la función logística como funciones en forma de S son la logística

bipolar $h(u) = (1-e^{-u})/(1+e^{-u})$ y la tangente hiperbólica $tanh(u) = (e^{u}-e^{-u})/(e^{u}+e^{-u})$

- Función gaussiana (forma de campana, *Gaussian function*). Una señal de salida de una neurona puede ser representada por un único potencial estático, o por un pulso, el cual ocurre (toma el valor de 1) o no ocurre (toma el valor de 0)

- Rectificador. Es conocida como función rampa y es análoga a la rectificación de media onda en electrónica. Se define como: $f(u) = max(0, u)$. Las unidades que emplean el rectificador son conocidas como unidad lineal rectificada o ReLU (*Rectified Linear Unit*). Una aproximación "suave"del rectificador es la función analítica $f(u) = ln(1 + e^{u})$, conocida como función *softplus*.

Aunque una sola neurona puede realizar ciertas funciones simples de procesamiento de información, el poder la computación neuronal proviene de la conexión de neuronas en redes.

Una red neuronal artificial es un modelo computacional definido por cuatro parámetros:

1. Tipo de neuronas (también llamadas nodos, ya que una red neuronal se asemeja a un grafo).

2. Arquitectura conexionista: la organización de las conexiones entre las neuronas. Las neuronas pueden estar completamente conectadas o parcialmente conectadas. De acuerdo con la ausencia o presencia de conexiones de retroalimentación en una red, se distinguen dos tipos de arquitecturas, la arquitectura *Feed-Forward* y la arquitectura retroalimentada (*Feedback*); en la primera no hay conexiones desde la salida a las neuronas de entrada, la red no mantiene recuerdo de sus valores de salida previos y del estado de activación de sus neuronas; mientras que en la segunda hay conexiones desde la salida a las neuronas de entrada, la red mantiene un recuerdo de sus estados previos y el siguiente estado depende no sólo de las señales de entrada, sino de los estados previos de la red.

3. Algoritmo de aprendizaje. La característica más atractiva de las redes neuronales es su capacidad de aprender. Se entrena una red neuronal para que la aplicación de un

conjunto X de vectores de entrada produzca el conjunto deseado (o que sea consistente) de vectores de salida Y, o la red aprenda sobre las características internas y las estructuras de datos de un conjunto X. El conjunto X usado para entrenar una red se llama conjunto de entrenamiento. Los elementos $\mathbf{x}$ de X se llaman ejemplos de entrenamiento. El proceso de entrenamiento se refleja en el cambio de los pesos de conexión de la red. Durante el entrenamiento, los pesos de la red deberían converger gradualmente a valores tales que cada vector de entrada $\mathbf{x}$ de los datos de entrenamiento establecidos provoquen un vector de salida $\mathbf{y}$ deseado y producido por la red. La capacidad de aprendizaje de una red neuronal se logra aplicando un algoritmo de aprendizaje (entrenamiento). Los algoritmos de entrenamiento se clasifican principalmente en tres grupos: supervisados, no supervisados y de aprendizaje reforzado.

4. Algoritmo de recuerdo: La otra característica general de las redes neuronales artificiales, similar a la capacidad del cerebro humano, es la generalización. Esto sucede cuando, después de haber entrenado una red neuronal, se presenta un nuevo vector de entrada $\mathbf{x}'$ a la red y se activa un procedimiento de "recuerdo". La red producirá una salida $\mathbf{y}'$ que es similar a la salida y_i de los ejemplos de entrenamiento si $\mathbf{x}'$ es similar al vector de entrada x_i. El principio general es que estímulos similares causan reacciones similares. La generalización puede tomar varias iteraciones para calcular estados consecutivos de la red. Eventualmente, la red entra en un estado de equilibrio, que es cuando la red no cambia su estado durante las siguientes iteraciones, es decir se "congela" en este estado.

El funcionamiento de una red neuronal cuando se suministra un vector de entrada $\mathbf{x}$, se puede ver como una función de mapeo $f : \mathbb{X} \to \mathbb{Y}$, donde $\mathbb{X}$ es el estado del espacio de entrada (dominio) y $\mathbb{Y}$ es el estado del espacio de salida (rango) de la red. La red simplemente mapea vectores de entrada $\mathbf{x} \in \mathbb{X}$ en vectores de salida $\mathbf{y} \in \mathbb{Y}$. El funcionamiento de una red generalmente se basa en cálculos de números reales de un vector o matriz. La matriz de pesos representa el "conocimiento", la memoria a largo plazo del sistema, mientras que la activación de las neuronas representa el estado actual, la memoria de corto plazo del sistema.

3.2.1. Retropropagación

Las redes con más de una capa se pusieron en práctica solo cuando los algoritmos de aprendizaje fueron desarrollados para estas, uno de ellos fue el algoritmo de retropropagación (*Back-Propagation*) [Kasabov, 1996].

Según [Goodfellow et al., 2016], cuando se usa una arquitectura *feedforward* para una red neuronal, las entradas $\mathbf{x}$ proveen la información inicial que se propaga hacia las capas ocultas y finalmente produce $\widehat{\mathbf{y}}$. A esto se le conoce como propagación hacia adelante. Durante el entrenamiento la propagación hacia adelante puede continuar hasta producir un costo escalar.

El algoritmo de retropropagación permite que la información de costo fluya hacia atrás a través de la red para computar el gradiente. El algoritmo de retropropagación se refiere al método para calcular el gradiente, mientras que otro algoritmo, tal como el descenso del gradiente es usado para realizar el aprendizaje mediante este gradiente [Goodfellow et al., 2016].

3.2.2. Gradiente descendente

Siguiendo las definiciones de [Kasabov, 1996] la regla de descenso del gradiente puede utilizarse para encontrar pesos óptimos de conexión $W_{i,j}$ que minimicen el error global E. Un cambio de un peso $\Delta W_{i,j}$ en un ciclo $(t+1)$ está en la dirección del gradiente negativo del error E (ecuación 3.1).

$$\Delta W_{i,j}(t+1) = -\alpha(\partial E/\partial W_{i,j}(T)), \tag{3.1}$$

donde α es la tasa de aprendizaje. La regla del gradiente asegura que después de un número de ciclos, el error E llegará a un valor mínimo. Un error global para todos los ejemplos de entrenamiento se puede calcular como se indica en la ecuación 3.2.

$$E = \sum_{p} \sum_{j} Err_j^p, \tag{3.2}$$

donde el error de un ejemplo p, es decir Err_j^p podría ser calculado, por ejemplo como un error cuadrático medio, como se observa en la ecuación 3.3.

$$Err_j^p = (y_j^p - x_j^p)^2/2 \tag{3.3}$$

3.2.3. Entropía cruzada

Para medir el error de un algoritmo de aprendizaje, debemos diseñar una medida cuantitativa. Usualmente esta es específica para la tarea llevada a cabo por el sistema y es conocida como error, éste se calcula mediante una función de pérdida. Existen diferentes funciones pa-

ra realizar este proceso, cada una se aplica según la tarea que se desee realizar, sin embargo unas funcionan mejor que otras para ciertas tareas.

Por ejemplo, si la tarea a realizar es una regresión, la función de pérdida que se debe usar es el error cuadrático medio (MSE). Por otra parte, si la tarea a realizar es una clasificación, la función de pérdida que resulta ser mejor es la entropía cruzada.

Según [Ramos et al., 2018], "la teoría de la información afirma que la información obtenida en un proceso inferencial está determinada por la reducción de la entropía, que mide la incertidumbre sobre una determinada variable desconocida a la luz del conocimiento disponible". En un problema de clasificación, la entropía representa la incertidumbre que tiene quien debe de tomar decisiones sobre el valores de la variable de hipótesis Θ.

Podemos decir entonces que a mayor probabilidad de que un evento ocurra, menor incertidumbre tendremos y viceversa. Por ejemplo, si tenemos una moneda cargada con 80 % de probabilidades para cara y 20 % para cruz, tendremos más incertidumbre o mayor entropía en el 20 % para cruz que en cara.

Continuando con la definición de [Ramos et al., 2018] en un problema de clasificación, la incertidumbre sobre Θ está dada por las probabilidades previas $P(\theta_1) = 1 - P(\theta_2)$, aunque las características aún no se observen. Con este conocimiento disponible, la entropía de la hipótesis está determinada por la ecuación 3.4.

$$H_p(\Theta) = -\sum_{i\epsilon 1,2} P(\theta_i) log_2 P(\theta_i) \tag{3.4}$$

Los trabajos de clasificación como el aquí propuesto hacen uso de la entropía cruzada debido a que esta medida ayuda a encontrar los pesos correctos de la red de una forma más rápida en comparación con otras medidas de error (ej. MSE). El uso de la entropía cruzada en este caso, permitirá saber la incertidumbre que existe al clasificar dos señales como iguales o diferentes.

3.3. Red neuronal convolucional

Una red neuronal convolucional (CNN) proporciona invarianza de cambio en el tiempo y en el espacio lo que ha permitido tener un alto desempeño en el reconocimiento de imágenes y que ha permitido demostrar que se puede mejorar la precisión de DNN puras en algunas tareas [Yu and Deng, 2015].

Las redes convolucionales están diseñadas para procesar datos que se presentan en forma de matrices. Algunos de estos tipos son: 1D para señales y secuencias; 2D para imágenes o espectrogramas de audio y 3D para imágenes volumétricas o video. Existen cuatro ideas fundamentales detrás de las redes convolucionales que permiten aprovechar las propiedades naturales de las señales y son: conexiones locales, pesos compartidos y muchas capas [LeCun et al., 2015].

De acuerdo con la definición de [LeCun et al., 2015], la arquitectura de una red convolucional comúnmente está estructurada como una serie de etapas. Las primeras capas suelen ser capas convolucionales y capas de submuestreo (*pooling*). Las unidades en una capa convolucional se organizan en mapas de características, dentro de las cuales cada unidad se conecta a parches locales en los mapas de características de la capa previa a través de un conjunto de pesos llamados bancos de filtros. El resultado de esta se pasa a través de una función no lineal como una ReLU. Todas las unidades en un mapa de características comparten el mismo banco de filtros. El motivo de la arquitectura es que en los arreglos matriciales de datos, los grupos locales de valores a menudo están altamente correlacionados y que las estadísticas locales de imágenes y otras señales son invariables a la ubicación. La operación matemática que se realiza para el filtrado y que genera mapas de características es una convolución discreta, por esto estas redes son conocidas como redes convolucionales.

Siguiendo la definición de [LeCun et al., 2015] la función de la capa convolucional es detectar conjunciones locales de características de la capa anterior y la función de la capa de submuestreo es fusionar características semánticamente similares en una.

Las redes neuronales profundas explotan la propiedad de que muchas señales naturales son jerarquías de composición, en las que las características de nivel superior se obtienen componiendo las de nivel inferior.

Como se describió anteriormente las redes convolucionales utilizan la función de convolución como se describe en la ecuación 3.5.

$$u(t) = (\mathbf{x} * \mathbf{w})(t) = \sum_{i=-\infty}^{\infty} x(i) \cdot w(t-i) \tag{3.5}$$

Donde $\mathbf{x}$ es el vector de entrada, mientras que $\mathbf{w}$ es el filtro. La salida $\mathbf{u}$ es el mapa de características.

El uso de un filtro de 2 dimensiones $K(i,j)$ en un imagen $I(i,j)$ describe un mapa de

características $S(i, j)$ como se muestra en la ecuación 3.6.

$$S(i,j) = (K * I)(i,j) = \sum_m \sum_n I(i-m, j-n)K(m,n) \tag{3.6}$$

3.4. Arquitectura

Existen diversas arquitecturas de redes neuronales convolucionales en la literatura, algunas de ellas probadas en diversos ámbitos, otras sólo probadas con propósitos específicos.

En esta sección se describirán las arquitecturas VGG 16 y ResNet 50 que son las que se ocupan en este trabajo.

3.4.1. VGG 16

La red *VGG 16* desarrollada por el *Visual Geometry Group* del departamento de ciencia de la ingeniería, de la Universidad de Oxford y que se describe en [Simonyan and Zisserman, 2014], es una red convolucional profunda que en el momento de ser propuesta buscaba estudiar el efecto de la profundidad de la red convolucional en el reconocimiento de imágenes a gran escala utilizando filtros de convolución de 3 x 3. Se propusieron dos redes con 16 y 19 capas respectivamente y fueron presentadas en el *ImageNet Challenge 2014* donde obtuvieron el primer y segundo lugar.

A continuación se da una descripción más detalla de la red que aparece en [Simonyan and Zisserman, 2014]. Para el entrenamiento la red convolucional usan imágenes RGB de 224×224. El único preprocesamiento que se hace es restar el valor RGB medio, calculado en el conjunto de entrenamiento de cada píxel. La imagen pasa a través de una pila de capas convolucionales, donde se usan filtros de 3×3. El paso de convolución (*convolution stride*) es de 1 píxel. Se utiliza un relleno espacial (*padding*) para conservar la resolución espacial después de realizar la convolución. Se realiza de igual forma una generalización por valor máximo (*max pooling*) utilizando una ventana de 2×2 y un paso de 2 en algunas capas.

Después de la pila de capas convolucionales (que tienen diferente profundidad en diferentes arquitecturas del tipo VGG), le siguen tres capas completamente conectadas (*fully-connected* o también llamadas densas en inglés *dense*). Las dos primeras tienen 4096 neuronas y la tercera tiene 1000 neuronas una para cada clase, que es la clasificación requerida por el ILSVRC. Todas las capas ocultas están equipadas con rectificadores no lineales (ReLu).

En la figura 3.1 se observa un diagrama de la red convolucional *VGG 16*.

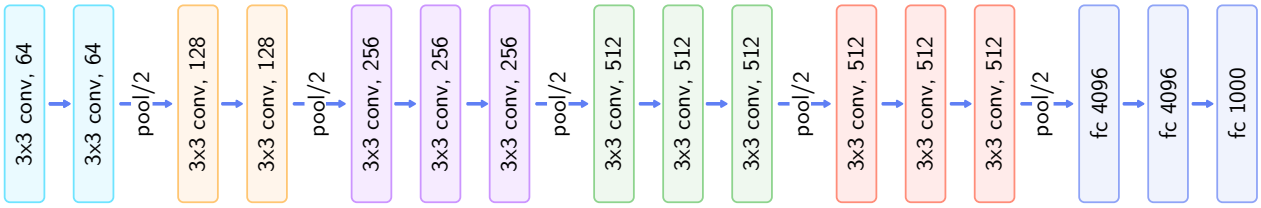

Figura 3.1: Diagrama de una red convolucionales VGG 16.

Otra particularidad de esta red convolucional, es la forma como fueron inicializados sus pesos, ya que una inicialización incorrecta puede detener el aprendizaje debido a la inestabilidad del gradiente en redes profundas. Para inicializar los pesos, los autores realizan el entrenamiento de una red convolucional más pequeña, con sólo 11 capas, cuyos pesos fueron inicializados aleatoriamente. Luego para entrenar la arquitectura de la VGG 16 se utilizan los pesos de las primeras 4 capas convolucionales y de las últimas 3 capas completamente conectadas de la red VGG 11 y el resto de las capas se inicializan aleatoriamente. Para la inicialización aleatoria se muestrean pesos de una distribución normal con media 0 y varianza 10^{-2}. Los sesgos se inicializan con valor de 0.

3.4.2. ResNet 50

Las redes profundas residuales a las que pertenece la ResNet 50 fueron desarrolladas por el grupo de *Microsoft Research* y se describen en [He et al., 2015]. El objetivo de estas redes fue encontrar una forma de entrenar redes neuronales muy profundas, ya que en redes muy profundas se evidencia el problema de desvanecimiento (*vanishing*) o explosión (*exploding*) del gradiente. Para esto utilizaron el aprendizaje residual, en su artículo dan una evidencia empírica de que estas redes residuales son más fáciles de optimizar y pueden obtener una mayor precisión con una profundidad mucho mayor que sus antecesoras. El uso de estas redes le permitió al grupo de *Microsoft Research* obtener el primer lugar en las tareas detección y localización del ILSVRC 2015 y de detección y segmentación de COCO 2015.

A continuación se da una descripción más detalla de la red que aparece en [He et al., 2015]. Para este tipo de redes se abordó el problema de la degradación del gradiente introduciendo un marco de aprendizaje residual profundo. En vez de esperar que las capas se ajusten a un mapeo deseado, se deja que se ajusten a un mapeo residual. Toda la red puede ser entrenada utilizando el algoritmo de Descenso Estocástico del Gradiente (*SGD*).

Para hacer uso del mapeo residual, se considera $H(x)$ como un mapeo subyacente para ser ajustado por una pocas capas apiladas. La hipótesis es que múltiples capas no lineales

pueden aproximar de manera asintótica funciones complicadas, entonces es equivalente hipotetizar que pueden aproximar de forma asintótica las funciones residuales, es decir, $H(x) - \mathbf{x}$ (suponiendo que la entrada y la salida son de las mismas dimensiones). Entonces, en lugar de esperar que las capas apiladas se aproximen a $H(x)$, se deja que estas se aproximen a una función residual $F(x) := H(x) - \mathbf{x}$. Así la función original se convierte en $F(x) + \mathbf{x}$.

Se adopta el aprendizaje residual para cada cierta cantidad de capas apiladas. Formalmente se considera un bloque de construcción descrito en la ecuación 3.7.

$$y = F(\mathbf{x}, W_i) + \mathbf{x}, \tag{3.7}$$

donde $\mathbf{x}$ es la entrada y $\mathbf{y}$ la salida del grupo de capas consideradas y $F(\mathbf{x}, W_i)$ representa el mapeo residual a ser aprendido.

Para el entrenamiento de las redes residuales hicieron uso de imágenes RGB de 224×224, esto debido a que se tomó el tamaño del lado más corto de las imágenes. Para las imágenes con tamaño mayor a este se hicieron cortes aleatorios que contuviera una imagen de 224×224. Se adopta la normalización por lotes después de cada convolución y antes de la activación, una normalización por lotes básicamente mantiene los valores normalizados según la media y la varianza de los datos [Ioffe and Szegedy, 2015]. Los pesos se inicializan desde cero. Se usa un SGD con un tamaño de minilotes de 256 muestras. Se inicia con una tasa de aprendizaje de 0.1 que se divide por 10 cuando el error llega a una meseta. No se usa desconexión de neuronales (*dropout*), esta práctica omite de forma aleatoria a ciertas neuronas, esto ayuda a reducir el sobreajuste, evitando adaptaciones en los datos de entrenamiento [Hinton et al., 2012]. En el artículo [He et al., 2015] se especifica que hacen entrenamientos utilizando redes planas y que las redes residuales tienen la misma arquitectura que las redes planas, sólo que, las conexiones de mapeo idéntico (*shortcut connections*) son agregadas cada cierta cantidad de capas.

Existen dos tipos de bloques para las redes residuales descritas en [He et al., 2015] que se pueden observar en la figura 3.2. El bloque de construcción (*building block*) que consta sólo de dos capas convolucionales con filtros de tamaño 3×3 y el bloque de construcción de cuello de botella (*bottleneck building block*)que consta de 3 capas convolucionales, la primera y tercera capa con filtros de 1×1 y la segunda con un filtro de 3×3. La conexión de mapeo idéntico va de la entrada a la salida de los bloques de construcción o bloques de construcción de cuello de botella.

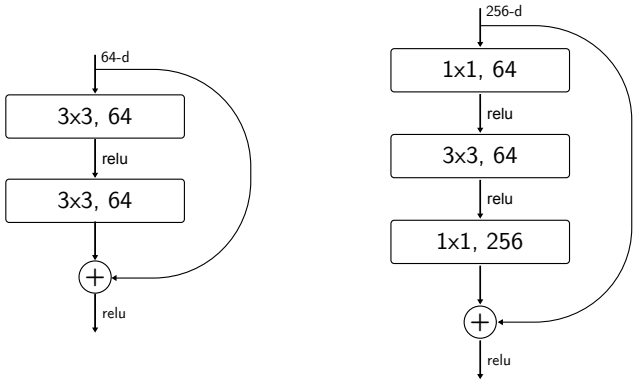

Figura 3.2: Diagrama del bloque de construcción y del bloque de construcción de cuello de botella.

Los bloques de construcción son utilizados por las redes residuales ResNet 18 y ResNet 34 las cuales cuentan con 18 y 34 capas respectivamente, los bloques de construcción de cuello de botella se usan en las redes residuales ResNet 50, ResNet 101 y ResNet 152 que son arquitecturas con más de 34 capas.

En la figura 3.3 se puede observar un diagrama de la red ResNet 50, en esta red se tienen 3 bloques de cuello de botella para el grupo 1, 4 para el grupo 2, 6 para el grupo 3 y 3 para el grupo 4. Las redes ResNet 101 y ResNet 152 tienen la misma distribución sólo que con mayor cantidad de bloques de cuello de botella por grupo. Para la ResNet 101 se tienen 3 bloques de cuello de botella para el grupo 1, 4 para el grupo 2, 23 para el grupo 3 y 8 para el grupo 4. Para la ResNet 152 se tiene 3 bloques de cuello de botella para el grupo 1, 8 para el grupo 2, 36 para el grupo 3 y 3 para el grupo 4.

3.5. Red siamés

Una red siamés es usualmente utilizada para realizar la tarea de verificación, esta tarea consiste en determinar si dos patrones de entrada son iguales o no. Este tipo de red tiene dos entradas para comparar dos patrones y una salida cuyo valor indica la similitud entre ambas entradas [Bromley et al., 1993] .

La idea principal de una red siamés es buscar una función que mapee patrones de entrada a un espacio, tal que una simple distancia en este espacio se aproxime a la distancia semántica del espacio de entrada [Bromley et al., 1993].

Una arquitectura siamés de acuerdo con la definición de [Hadsell et al., 2006], consta de dos copias de la función f las cuales comparten el conjunto de parámetros $\mathbf{W}$ y un módulo

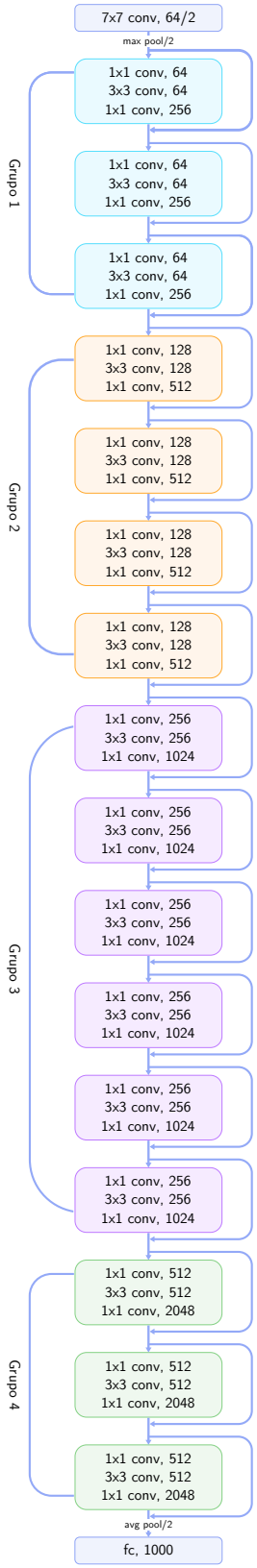

Figura 3.3: Diagrama de una red residual ResNet 50.

de costo. La entrada del sistema son un par de imágenes x_1,x_2 y una etiqueta y. Las imágenes pasan a través de las funciones, produciendo dos salidas $f(x_1)$ y $f(x_2)$. El módulo de costo genera la distancia $D(f(x_1), F(x_2))$. La función de pérdida combina D con la etiqueta y para producir la pérdida.

Los pesos **W** se actualizan utilizando un gradiente estocástico. Los gradientes se pueden calcular a través del algoritmo de retropropagación, el costo y las dos instancias de f. La salida de este tipo de redes puede ser una medida de distancia, de similitud o una clasificación binaria. En la figura 3.4 se puede observar un diagrama simplificado de una red siamés.

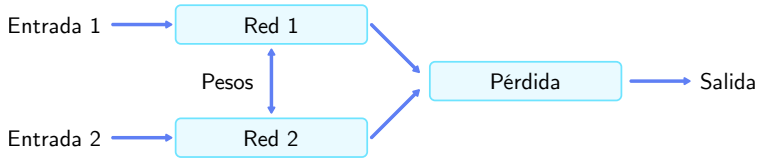

Figura 3.4: Diagrama simplificado de una red siamés.

El sistema es entrenado con parejas de muestras del grupo de entrenamiento. Es posible observar que una red siamés está compuesta principalmente por dos elementos. El primero se puede ver como un módulo encargado de extraer características de las entradas. El segundo elemento puede ser observado como un módulo de verificación, que puede utilizar una función de similitud sobre los dos conjuntos de características extraídas para determinar la similitud entre ambas. La implementación de estos dos módulos pueden variar mucho, sin embargo es importante conservar la misma idea.

Con respecto al módulo de verificación, en el trabajo de [Bromley et al., 1993] se propone el coseno del ángulo entre los dos vectores de características extraídos como medida de distancia. Trabajos más actuales como [Hadsell et al., 2006] proponen una función de pérdida contrastiva entre las salidas del sistema.

En este trabajo se exploran diferentes arquitecturas para la extracción de características y para determinar la similitud de los vectores resultantes. La verificación se aborda como un problema de clasificación, cuya tarea objetivo es determinar si dos conjuntos de características son iguales (clase 0 o [0 1]) o son diferentes (clase 1 o [1 0]), esto es explicado con más detalle en la sección 5.6.

Muchos trabajos han hecho uso de redes siameses, por ejemplo en [Hadsell et al., 2006] se usa una red siamés para la reducción de dimensionalidad aprendiendo un mapeo invariante, el caso de estudio fueron los dígitos del 0 al 9. Mientras que en [Baraldi et al., 2015] se propone una red siamés para la detección de escenas en videos de difusión. Por otra parte [Appalaraju

and Chaoji, 2017] usa una red siamés para determinar la similitud entre imágenes. En [Zagoruyko and Komodakis, 2015] se usa una red siamés para aprender a comparar parches de imágenes. En [Koch et al., 2015] se usa una red siamés para el reconocimiento de imágenes, en el que se busca realizar la predicción correcta utilizando sólo un ejemplo de una clase nueva. La estrategia es que el modelo aprenda a discriminar entre una colección de pares iguales y diferentes, de manera que puede generalizar al evaluar nuevas categorías basado en los mapas de características aprendidos. Para audio, podemos encontrar que en [Nagrani et al., 2017] se usa una red siamés preentrenada para la verificación de locutores.

Como se observa han habido diversos acercamientos a las redes siameses utilizando imágenes. El trabajo que más influye para la realización de este proyecto es el de [Koch et al., 2015], donde se indica que el modelo de la red propuesta aprende a generalizar de tal forma que es capaz de evaluar una clase nueva con la cual no fue entrenada.

Capítulo 4

Verificación vs identificación

El reconocimiento automático de locutor busca que una máquina pueda determinar quién le está hablando o con quién está interactuando. Para ello existen dos formas de llevar a cabo el reconocimiento, mediante identificación o mediante verificación [Campbell, 1997]. Ambos enfoques pueden funcionar para reconocer personas, sin embargo, el modo en que cada uno lo realiza es diferente. Para el caso del identificador, el sistema se entrena únicamente para aprender los locutores que será capaz de identificar. Al momento de realizar una identificación, el sistema clasificará la señal en alguno de los locutores conocidos.

Usualmente en un sistema de verificación de audio se entrenan varios verificadores para identificar diferentes locutores, es decir, se crea un verificador para cada uno, el cual indica si la señal de entrada pertenece o no al locutor para el cual fue entrenado. Cuando se utiliza este enfoque, al requerir una identificación, la señal ingresa a diferentes verificadores, y de acuerdo con los valores obtenidos por estos, se clasifica la señal como de algún hablante conocido o como desconocido.

En contraste a los sistemas clásicos de identificación mediante verificación, en este trabajo se expone un identificador de locutor que utiliza un verificador genérico, capaz de recibir dos señales, una proveniente de una base de datos donde se encuentran los locutores conocidos y otra que es la señal del locutor a identificar. La verdadera tarea del sistema es verificar si dos señales son iguales o no ya que no se entrenan verificadores específicos por cada locutor. Este enfoque permite indicar si la señal entrante existe en la base de datos de locutores conocidos. Además permite agregar locutores sin necesidad de crear y entrenar nuevos verificadores.

La identificación de locutor suele se más exacta que la verificación, sin embargo ésta última tiene la ventaja de poder indicar que un locutor es desconocido [Campbell, 1997].

El capítulo se compone de la siguiente manera: en la sección 4.1 se da una explicación

de la identificación de locutor. En la sección 4.2 se explican las diferencias entre las bases de datos dependientes e independientes del texto. Posteriormente se exponen algunos trabajos que se han llevado a cabo para la identificación de locutores en la sección 4.3 y por último se habla brevemente en la sección 4.4 de cómo el aprendizaje profundo ha impactado en el área de identificación de locutor y en especial para este trabajo.

4.1. Identificación de locutores

El habla es una señal compuesta de muchos factores y en la cual ocurren transformaciones semánticas, lingüísticas, articulatorias y acústicas, lo cual permite que hayan diferencias en la propiedades acústicas de la señal de voz y que puedan ser diferenciadas. Las diferencias son generadas de combinar tanto la anatomía del tracto vocal del locutor como los hábitos del habla, lo que incluye el lugar de nacimiento, el idioma o el acento con el que se habla [Campbell, 1997]. Todas estas diferencias pueden ser explotadas por los sistemas de identificación de locutores.

4.2. Base de datos

Existen dos tipos de sistemas para verificación o identificación, dependientes e independientes del texto. Según la tarea que se desee realizar será la elección de la base de datos, ya que esta impactará en el desempeño del sistema.

4.2.1. Dependiente del texto

Las bases de datos dependientes del texto corresponden a aquellas cuya frase o palabra clave es conocida por el sistema [Campbell, 1997] y no cambia a través del tiempo. Un ejemplo de un sistema que utiliza una base de datos dependiente del texto, es el del reconocimiento telefónico de voz de algunos bancos, en los cuales el usuario para ser identificado dice siempre la misma frase, la cual fue grabada previamente en el sistema.

Para entrenar un sistema dependiente del texto, se requiere una base de datos dependiente del texto, es decir que cada audio de la base de datos tenga la misma frase hablada que se desea reconocer.

4.2.2. Independiente del texto

Las bases de datos independientes del texto no están restringidas por el contenido de las palabras, el locutor puede interactuar con el sistema utilizando cualquier frase. Suele ser más conveniente para el usuario quien no necesita recordar una palabra o frase clave, sino que puede hablarle libremente al sistema. Sin embargo, los sistemas que utilizan base de datos independientes del texto requieren entrenamientos más largos y mayor cantidad de pruebas para llegar a un buen desempeño [Liu et al., 2006].

Para entrenar un sistema independiente del texto, se requiere una base de datos independiente del texto, es decir que cada audio puede contener cualquier contenido hablado ya que no hay restricción en el texto que se desea reconocer.

4.2.3. Bases de datos dependientes e independientes del texto

En la tabla 4.1 se listan algunas de las bases de datos disponibles para verificación e identificación de locutores. En ésta se listan 16 bases datos que han sido utilizadas para abordar el problema del reconocimiento de locutor.

Una de las bases de datos que más ha sido usada es la de TIMIT [John S. Garofolo, 1993] y su extensión a NTIMIT [William M. Fisher, 1993] (en la cual se transmiten los audios de TIMIT a través de la red telefónica). Su alto uso se debe a que es una de las bases de datos para reconocimiento de locutor que tiene más años y que ha reportado buenos resultados en diferentes trabajos, su licencia es privada y el acceso no es libre.

Una dificultad hallada con muchas de las bases de datos es que su acceso es privado, lo que limita la cantidad de datos con los cuales se puede interactuar. Sin embargo en los últimos años se han publicado bases de datos tanto dependientes como independientes del texto con una alta cantidad de locutores y de acceso libre.

La base de datos más reciente encontrada es la de Voxceleb, que no sólo tiene 1,251 locutores, sino que su audio está grabada en diferentes condiciones ambientales, lo que permite desarrollar sistemas más complejos que logren estudiar a los locutores y su contenido de audio en entornos cada vez más difíciles y que se asemejan más al mundo real.

Nombre	Año	Loc.	Texto	Tipo de grabación	H	Lic.	Referencia
Switchboard	Desde 1990	113	Independiente	Audio telefónico		Privada	[John J. Godfrey, 1997]
TIMIT	1993	630	Dependiente e independiente	Grabación con micrófono en entorno controlado		Privada	[John S. Garofolo, 1993]
NTIMIT	1993	630	Dependiente e independiente	Grabación telefónica de los audios de TIMIT		Privada	[William M. Fisher, 1993]
YOHO	1994	138	Dependiente	Grabación con micrófono en entorno controlado		Privada	[Joseph Campbell, 1998]
AHUMADA	2000	104	Dependiente	Audio telefónico, diferentes micrófonos		Pública	[Ortega-Garcia et al., 2000]
POLYCOST	2000	134	Dependiente	Audio telefónico		Privada	[Hennebert et al., 2000]
VeriVox	2000	50	Dependiente	Cabina con micrófono de alta calidad	25		[Karlsson et al., 2000]
ELSDSR	2005	22	Independiente	Cámara	0.108	Pública	[Feng and Kai Hansen, 2005]
MIT-MDSVC	2006	88	Dependiente	Oficina silenciosa, pasillo medianamente ruidoso y concurrida intersección		Pública	[Woo et al., 2006]
Voxforge	2006 - 2017	1232	Independiente			Pública	[Voxforge.org,]
BioSecure	2010	˜600	Dependiente	Internet, en oficina con una computadora de escritorio y con dispositivos móviles en espacios abiertos y cerrados		Privada	[Ortega-Garcia et al., 2010]
UNMC-VIER	2011	123	Dependiente	Controlada		Privada	[Wong et al., 2011]
RSR2015	2012	300	Dependiente	Dispositivos móviles	151	Pública	[Larcher et al., 2014]
LibriSpeech	2015	2484	Independiente	Audiolibros	1000	Pública	[Panayotov et al., 2015]
MUSAN	2015	172	Independiente	Audiolibros y Audiencias, comités y debates del gobierno de los Estados Unidos	60	Pública	[Snyder et al., 2015]
SITW	2016	299	Independiente	Entrevistas de estudio, alfombra roja, al aire libre y escenarios con múltiples oradores		Pública	[McLaren et al., 2016]
Voxceleb	2017	1,251	Independiente	Entrevistas de estudio, alfombra roja, al aire libre, discursos a grandes audiencias, extractos multimedia y videos grabados con dispositivos móviles,		Pública	[Nagrani et al., 2017]

Tabla 4.1: Listado de bases de datos disponibles, donde Loc. se refiere al número de locutores con los que cuenta la base datos, texto indica si es dependiente o independiente, H indica la cantidad de horas de audio grabadas y Lic. indica el tipo de licencia que tiene la base de datos (pública o privada).

4.3. Métodos para la identificación automática de locutor

Durante varias décadas los identificadores de voz han sido estudiados desde diferentes enfoques, buscando una mayor eficiencia en sus resultados, además de utilizar diversos tipos

de implementaciones y tipos de datos. A continuación se citan algunos de los trabajos que han abordado el problema de la identificación automática de locutor y en especial aquellos que usan verificadores, se describe brevemente qué tipo de modelo utilizaron, qué tipo de datos y cuál fue su desempeño.

Durante la década de los 90 e inicios de los años 2000, los modelos de mezcla de gaussianas (GMM) y los modelos ocultos de Markov (HMM) fueron ampliamente estudiados utilizando como características de audio MFCC, MFB y LPC 2.4, por ejemplo en los trabajos de [Reynolds, 1995], [Thévenaz and Hügli, 1995], [Matsui and Furui, 1995], [Forsyth, 1995], [Petrovska-Delacrétaz et al., 2000], [Xiang et al., 2002], [Ramos-Castro et al., 2007], entre otros, se usan este tipo de acercamientos. Sin embargo el enfoque propuesto en este trabajo hace uso de redes neuronales para la identificación de locutor a través de verificación, por lo sólo se nombran trabajos relacionados a ésta área.

El uso de las redes neuronales en conjunto con el audio, no se ha limitado únicamente a la identificación y verificación de locutor, sino que también ha abordado el tema de la extracción o generación de características de audio para que éstas sean más robustas. En los trabajos de [Snyder et al., 2017], [Snyder et al., 2016], [Variani et al., 2014], [Bhattacharya et al., 2017] y [Heigold et al., 2016] se utilizan *embeddings* o redes neuronales completas que generan nuevas características, y posteriormente se emplean métodos estadísticos que permiten hacer la verificación de un locutor. Durante este trabajo se busca que la red convolucional extraiga las características que necesita para la identificación de locutor, por lo que no se toca el tema de la generación de características utilizando redes neuronales.

En [Daqrouq, 2011] se busca realizar la identificación de locutores mediante la comparación de una medida de similitud entre el audio de un locutor a identificar y los patrones previamente generados para los locutores conocidos. Este trabajo utiliza una base de datos de números del 0 al 14 en idioma árabe creada por el mismo autor utilizando una tarjeta *PC-sound* con una frecuencia de muestreo de $16kHz$. Si bien el trabajo dice utilizar una base de datos de texto independiente (ya que las frases compuestas por números no son iguales en cada caso), está limitada únicamente a números. Para la identificación utilizan una red neuronal de 4 capas (1 entrada con 35 neuronas, 1 de salida con 5 neuronas y 2 capas ocultas cada una con 20 neuronas ocultas), ésta red es optimizada utilizando el algoritmo de retropropagación de Levenberg-Marquardt. Éste algoritmo combina el algoritmo de descenso

del gradiente que se utiliza para encontrar el mínimo de una función con el algoritmo de Gauss-Newton, utilizado para resolver problemas no lineales de mínimos cuadrados [Kisi and Uncuoglu, 2005]. Como entrada se utiliza una matriz de 35 características (30 coeficientes de la entropía de Shannon para la transformada ondicular por paquetes y 5 frecuencias formantes que corresponden a las frecuencias de resonancia en el tracto vocal) por 18 muestras. Utilizando 29 locutores el autor obtiene una exactitud del 91.09 % en identificación.

En [Daqrouq and Tutunji, 2015] proponen algunos cambios al sistema del [Daqrouq, 2011]. La base de datos con la que se trabaja tiene 80 locutores grabados con una frecuencia de muestreo de $8kHz$. La base de datos únicamente contiene números. Para la identificación de locutor proponen dos métodos, una red neuronal de 3 capas (12 neuronas de entrada, 30 neuronas en la capa oculta y para la capa de salida una cantidad de neuronas que depende del número de locutores a identificar), ésta red es optimizada utilizando el algoritmo de retropropagación de Levenberg-Marquardt. Como entrada se utiliza una matriz de 12 características (7 coeficientes de la entropía de Shannon para la transformada ondicular por paquetes y 5 frecuencias formantes que corresponden a las frecuencias de resonancia en el tracto vocal) por el número de locutores que se desea identificar. Utilizando los 80 locutores se obtiene una exactitud de identificación del 90.09 % para la red neuronal de 3 capas y del 76.56 % para la red neuronal probabilística. El autor propone una mejora a su sistema incrementado el número de niveles que tiene el árbol de la transformada ondicular, sin embargo esto representa un aumento en el tiempo que se toma para extraer los coeficientes de la entropía de Shannon.

En [Muckenhirn et al., 2017] se hace la verificación de locutor utilizando redes neuronales convolucionales y audio crudo (sin ninguna transformación de tiempo, frecuencia o extracción de características artesanales). Para este trabajo los autores proponen utilizar una red neuronal convolucional que extrae la información relevante de la señal de audio crudo. Primero se entrena un identificador de locutores y posteriormente con estos pesos como inicialización se genera un verificador para cada locutor, quitando la última capa de la red preentrenada y reemplazándola por un verificador que tiene dos salidas o clases (genuino cuando el locutor pertenece a la clase e impostor cuando no). La red neuronal utilizada se compone de dos módulos convolucionales (capa de convolución, submuestreo máximo, función de activación) cada uno con 20 filtros, la primera con un kernel de 300 muestras y la segunda con un kernel de 200, una capa oculta de 100 neuronas previa a la de salida y la capa de salida. La entrada

de la red neuronal es una señal de audio crudo de 510 ms. Para el entrenamiento tanto del identificador como de los verificadores se utiliza el descenso estocástico del gradiente con criterio temprano de paro, utilizando como función de costo una función basada de entropía cruzada. La base de datos que se usa es Voxforge ([Voxforge.org,]) y utilizando 100 locutores en la etapa de evaluación se llega a un desempeño de HTER (*Half Total Error Rate*) del 1.2 %.

En [Nagrani et al., 2017] no sólo se realiza la descripción de la base de datos VoxCeleb, sino que adicionalmente se utilizan redes neuronales profundas tanto para la identificación como para la verificación de locutor. La arquitectura que ellos proponen es una red convolucional que se describe en la tabla 4.2.

Layer	**Support**	**Filt dim.**	**N. Filt**	**Stride**	**Data size**
conv1	7×7	1	96	2×2	254×148
mpool1	3×3	-	-	2×2	126×73
conv2	5×5	96	256	2×2	62×36
mpool2	3×3	-	-	2×2	30×17
conv3	3×3	256	256	1×1	30×17
conv4	3×3	256	256	1×1	30×17
conv5	3×3	256	256	1×1	30×17
mpool5	5×3	-	-	3×2	9×8
fc6	9×1	256	4096	1×1	1×8
apool6	$1 \times n$	-	-	1×1	1×1
fc7	1×1	4096	1024	1×1	1×1
fc8	1×1	1024	1251	1×1	1×1

Tabla 4.2: Arquitectura convolucional descrita en [Nagrani et al., 2017]

La identificación se trata como una tarea de clasificación. Se usa la arquitectura de la tabla 4.2 y la última capa tiene una salida de *SoftMax* de 1251 para los 1251 locutores que tiene la base de datos VoxCeleb. La verificación se realiza utilizando la misma arquitectura que la usada para la identificación pero de forma siamés, ésta tiene dos entradas y dos copias de la misma red hasta la capa fc7, en la última capa (fc8) en vez de tener 1251 neuronas se tienen 256 neuronas, con estas dos salidas se calcula la distancia del coseno con la cual se determina si las señales de entrada son del mismo locutor o de diferente. Para la verificación se utilizan como pesos fijos los pesos resultantes del entrenamiento del identificador excepto en la última capa, la cual es entrenada utilizando pérdida contrastiva.

Los datos con los cuales se entrenan las redes son espectrogramas de frecuencia (descritos

en la sección 2.4.3) en un sólo canal de grabaciones con una tasa de muestreo de 16 kHz, con una ventana de Hamming de 25 ms, con pasos de 10 ms y una FFT de 1024 puntos. Lo que da como resultado imágenes de 512×300 para 3 segundos de audio.

Para identificación obtuvieron una exactitud del 80.5 % y del 92.1 % para el top-1 y top-5 de clasificación respectivamente. Para verificación tuvieron un EER del 7.8 %.

En [Salehghaffari, 2018] se realiza la verificación de locutor utilizando redes neuronales convolucionales. El autor utiliza la base de datos de Voxceleb [Nagrani et al., 2017] y propone el uso de una red convolucional siamés basada en la red CNN-M descrita por el Visual Geometry Group, la red propuesta en este trabajo consta de dos copias de la arquitectura descrita en 4.3, a las dos salidas resultantes se les calcula la distancia del coseno para determinar si son del mismo locutor o de diferente locutor, esta red es entrenada utilizando la función de costo contrastiva.

Las características de audio que alimentan esta red neuronal corresponden a un arreglo de $3 \times 40 \times 100$ compuesto de la energía logarítmica de 40 bancos de filtros y su primera y segunda derivada para cada ventana de Hamming de 25 ms con paso de 10 ms con 512 puntos para la FFT que genera un espectrograma de 256×100 para un audio de 1 segundo.

Layer	**Support**	**N. Filt**	**Stride**	**Data size**
conv1	7×7	32	2×2	$32 \times 17 \times 47$
conv2	5×5	64	1×1	$64 \times 13 \times 43$
conv3	3×3	128	1×1	$128 \times 11 \times 41$
conv4	3×3	256	1×1	$256 \times 9 \times 39$
conv5	3×3	256	1×1	$256 \times 7 \times 37$
fc1	-	1024	-	-
fc2	-	256	-	-
fc3	-	1251	-	-

Tabla 4.3: Arquitectura convolucional descrita en [Salehghaffari, 2018]

El autor reporta en sus resultados un EER de 11.3 % para el modelo propuesto y 10.5 % para un modelo con selección activa de pares.

En [Mobiny, 2018] se realiza la verificación de locutor utilizando LSTM. Para llevar a cabo esta tarea, el autor propone la extracción de la energía logarítmica de 40 bancos de filtros en audios de 1 segundo con ventanas de 25 ms y 60 % de traslape, lo que da como resultado una matriz de información de 40 x 100 datos. Para cada locutor se crea un modelo siamés que se compone de dos LSTM (con 2 capas de 300 nodos cada una) y una capa

completamente conectada para las cuales se comparten los pesos, adicionalmente tiene una pérdida contrastiva, para tener una arquitectura adecuada para la verificación. Esta pérdida contrastiva será la función a optimizar. El autor indica que para una base de datos de texto independiente llega a un EER de 22.1 % con uno de sus modelos y de 22.9 % con otro. Sin embargo no se detalla el algoritmo de optimización utilizado, la base de datos, ni el número de locutores que fueron utilizados para el sistema.

4.4. Aprendizaje profundo

Para este trabajo se utiliza una red neuronal convolucional, este tipo de redes se componen de conjuntos de filtros individuales que una vez que son aplicados sobre sucesivos fragmentos de una imagen proveen mapas de características, estos mapas permiten determinar cuáles de estas son importantes a la hora de reconocer un elemento en una imagen o en este caso qué conjuntos de características son importantes para identificar a un locutor. Estas redes también son conocidas por su capacidad de generalizar, en el caso de las imágenes, pueden identificar un elemento deseado a pesar de estar en otro tipo de ambiente (ej. un ave en un árbol y un ave en una cornisa), en este caso se busca que su capacidad de generalización permita identificar a un locutor aún cuando este cambie de ambiente o el ruido de fondo sea diferente. Se espera que una red neuronal convolucional siamés sea capaz de generalizar y aprender una clase con la que no fue entrenada basados en el modelo propuesto por [Koch et al., 2015] y que sirve como inspiración para la red siamés aquí propuesta.

Las redes de tipo siamés han obtenido altos desempeños en la clasificación de imágenes, por lo que se propone hacer uso de espectrogramas de frecuencia de señales de audio, los cuales se puede ver como una representación gráfica de éstas y más precisamente como una imagen de un sólo canal, para entrenar la red convolucional. Este marco de representación es utilizado en [Nagrani et al., 2017], donde se emplean redes neuronales con señales de audio, y que es una de las investigaciones más cercanas a este trabajo. Empíricamente también se comprobó que el uso de espectrogramas de frecuencia mejora el desempeño de los modelos, después de haber realizado experimentos con el audio crudo (resultados no reportados aquí, por que no se obtuvo ningún resultado superior al de elegir aleatoriamente si dos audios eran del mismo locutor o no), transformada de Fourier, componentes de baja frecuencia de la transformada de Fourier y componentes de alta frecuencia de la transformada de Fourier. Los espectrogramas de frecuencia a su vez permiten tener información frecuencial y temporal del locutor que se desea identificar, esto es deseable ya que la forma como interactúa el locutor a

través del tiempo también puede ser un factor que permita su diferenciación. Por último se desea que el sistema puede identificar personas con quienes nunca ha interactuado durante su entrenamiento, independientemente del texto hablado, por lo que se busca una forma de representar la información del locutor que pueda dar datos acerca de su firma espectral o de las características que lo diferencian de otros locutores y no sólo de las palabras que está emitiendo y esto se puede realizar con los espectrogramas de frecuencia.

Capítulo 5

Metodología

Para este trabajo se propone el desarrollo de un identificador de locutor mediante el uso de un verificador utilizando aprendizaje profundo. Este modelo podrá considerarse como un modelo general para la identificación de locutor, ya que en realidad busca verificar si dos señales son iguales o diferentes. El sistema puede identificar aquellos locutores de los cuales tenga audios conocidos a priori en su base de datos, si no los conoce los puede identificar como locutores desconocidos.

Para llevar a cabo este desarrollo, se siguieron los siguientes pasos:

1. Elección de la base de datos con la que se entrena el modelo.
2. Elección del *VAD*.
3. Selección de los datos de entrenamiento.
4. Selección de los datos de verificación.
5. Selección del algoritmo de optimización.
6. Selección del modelo a entrenar.
7. Entrenamiento de modelos.
8. Selección de modelos.
9. Evaluación de los modelos como verificadores.
10. Evaluación de los modelos como identificadores.

5.1. Elección de la base de datos

La selección de la base de datos para este proyecto, se realizó tomando en consideración la aplicación final que se desea tenga el modelo desarrollado, y es la de identificar locutores utilizando un robot de servicio. En este caso se espera que los locutores hablen uno a la vez, bajo condiciones ambientales y acústicas controladas, como es el caso de una casa o un restaurante, que es donde estaría ubicado el sujeto de prueba. En la aplicación final se espera que el robot de servicio interactúe con una cantidad limitada de personas, en el caso de una casa sólo los habitantes de la casa y los amigos frecuentes y en el caso de un restaurante se espera que recuerde a los clientes frecuentes y a los demás que sólo los recuerde durante el tiempo de atención que dura una comida.

De acuerdo con lo anterior se establecen las siguientes características deseadas para la base de datos que entrena el modelo:

- Independiente del texto: se busca que la base de datos sea independiente del texto ya que la interacción humano-robot debe darse de manera natural y de una forma similar a la interacción humano-humano, en la cual no se requiere una conversación preestablecida para identificar al locutor con el que se interactúa.
- Libre: se busca que la base de datos sea libre para tener acceso a los datos requeridos de forma rápida y fácil.
- Contener al menos 100 locutores: se busca que la base de datos contenga al menos 100 locutores porque idealmente el modelo debe aprender qué es una señal de voz y qué la caracteriza. Para abstraer este tipo de características se requiere exponer al modelo a una gran cantidad de señales de voz diferentes, donde la importancia de las características extraídas no radica en saber a qué locutor pertenece una voz, sino en qué se diferencian dos señales de voz.
- Idioma inglés: se busca que la base de datos sea en inglés ya que es el idioma en el que se espera se desarrolle la interacción entre el humano y el robot.

5.1.1. Bases de datos disponibles

En la tabla 5.1 se listan las bases de datos disponibles que cumplen con las características nombradas en la sección 5.1 y se indican sus pros y contras.

Nombre	Locutores	Pros	Contras	Referencia
MUSAN	172	▪ 60 horas de grabación ▪ 12 idiomas	▪ Pocos locutores	[Snyder et al., 2015]
SITW	299	▪ Diversas condiciones de grabación	▪ Más de un locutor por audio ▪ No tiene etiquetado para cada sección del audio	[McLaren et al., 2016]
Voxforge	1232	▪ Cuenta con un repositorio de MFCC para sus audios	▪ Muchos datos anónimos ▪ La base no está balanceada por usuario	[Voxforge.org,]
Voxceleb	1251	▪ Diversas condiciones de grabación	▪ La base de datos proviene de videos de youtube y se desconoce si todos los videos tienen licencia *creative commons* para evitar infringir derechos de autor	[Nagrani et al., 2017]
LibriSpeech	2484	▪ Completamente etiquetado	▪ No tiene diversas condiciones de grabación	[Panayotov et al., 2015]

Tabla 5.1: Listado de bases de datos disponibles

Se elige la base de datos LibriSpeech, que cumple con todas las características deseadas y que ha sido probada en trabajos como el de [Collobert et al., 2016] para el reconocimiento de voz de extremo a extremo, el cual usa una red convolucional y un grafo de decodificación para producir el discurso transcrito de un audio. Esta base de datos se utiliza para el entrenamiento de la red convolucional propuesta.

También se elige la base de datos SITW, la cual es utilizada para la evaluación del sistema entrenado. Esta base de datos cumple con las características deseadas, sin embargo su gran defecto es que tiene más de un locutor por audio y el número total de locutores es inferior al total de locutores de LibriSpeech.

5.2. *VAD*

El uso del *VAD* (descrito en la sección 2.1.7) sirve para elegir trozos de audio cuyo contenido auditivo que supera un umbral sea considerado como iteracción del locutor en ese audio.

Para los efectos de este proyecto, el *VAD* a utilizar es fijo y está dado por un umbral de 0.05 de la magnitud de la energía. Este valor fue elegido a partir del análisis de la base de datos con la que se entrenaron los diferentes modelos propuestos, cuyos valores están contenidos en el rango de energía de -1 a 1, siendo comúnmente los valores superiores a 0.1 e inferiores a -0.1 aquellos que contienen un hablante.

El umbral de 0.05 fue utilizado para el entrenamiento de las diversas máquinas de aprendizaje propuestas en este trabajo.

5.3. Selección de los datos de entrenamiento

La base de datos con la que se entrenan los diversos modelos es LibriSpeech. Esta base de datos cuenta con 2,484 locutores, de los cuales 1,987 locutores con un total de 234,977 grabaciones se seleccionaron para entrenamiento, 248 locutores con un total de 28,641 grabaciones para validación y 249 locutores con un total de 28,749 grabaciones para prueba. Estos datos representan el 80 % de los locutores para entrenamiento, el 10 % para validación y el 10 % para prueba. También representan el 80.4 % del total de audios para entrenamiento, el 9.8 % para validación y 9.8 % para prueba. La elección de los locutores para entrenamiento, validación y prueba se realizó de manera aleatoria y no existe ningún traslape entre los locutores de los 3 conjuntos anteriormente mencionados. Los datos de la base de datos LibriSpeech están muestreados con una tasa de 16 kHz.

Los audios se pueden representar de diversas formas lo que genera diferentes tipos de datos tanto para el entrenamiento, la validación y la prueba y son:

- Transformada de Fourier: en este trabajo esta representación del audio se conoce como FFT. Para esta transformada se toma una ventana de audio de 0.5 s, 1 s o 2 s dependiendo del experimento que se esté realizando. A esta ventana se le aplica la transformada rápida de Fourier, lo que genera el mismo número de puntos que los que se tenían en el tiempo, pero siendo útiles sólo la mitad de estos. Esta mitad de los puntos representan las componentes frecuenciales desde 0 Hz hasta 8 kHz espaciados

a 1 Hz en el caso de los audios de 1 s. Una representación gráfica de lo anteriormente descrito se puede ver en la figura 5.1.

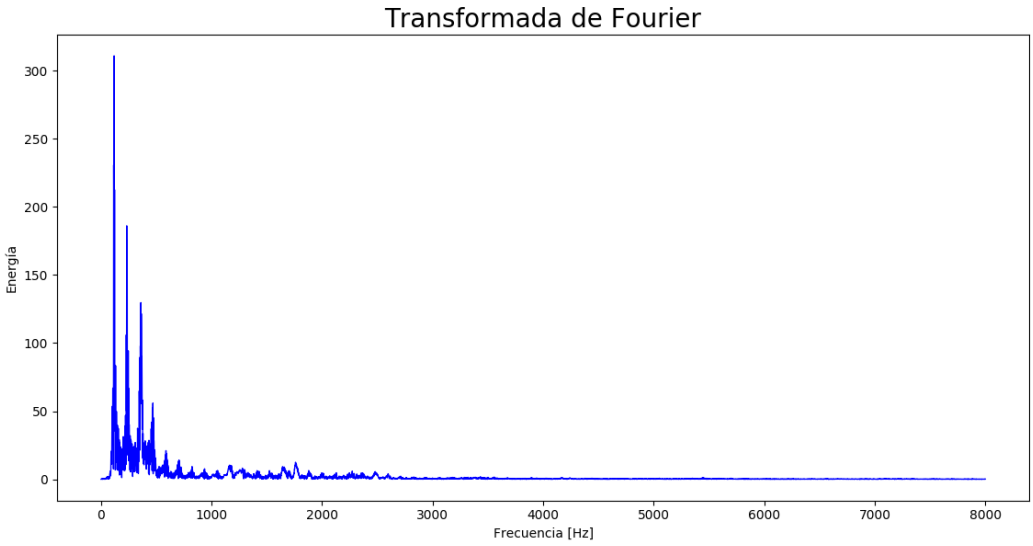

Figura 5.1: Representación gráfica de la transformada rápida de Fourier para un audio de 1 s.

- Componentes de baja frecuencia de la transformada de Fourier: en este trabajo a este tipo de dato se le conoce como FFT LF. Esta selección de datos corresponde a la mitad baja de la transformada de Fourier descrita en el apartado anterior. Para esta característica se aplica la transformada de Fourier pero en vez de tomar todos los datos útiles, sólo se toma la mitad baja de estos. Esta mitad de los puntos representan las componentes frecuenciales desde 0 Hz hasta 4 kHz espaciados a 1 Hz en el caso de los audios de 1 s. Una representación gráfica de lo anteriormente descrito se puede ver en la figura 5.2, donde la sección coloreada de rojo corresponde a los datos de baja frecuencia de la transformada de Fourier.

- Componentes de alta frecuencia de la transformada de Fourier: en este trabajo a este tipo de dato se le conoce como FFT HF. Esta selección de datos corresponde a la mitad alta de la transformada de Fourier descrita en el apartado sobre la transformada de Fourier. Para esta característica se aplica la transformada de Fourier pero en vez de tomar todos los datos útiles, sólo se toma la mitad alta de estos. Esta mitad de los puntos representan las componentes frecuenciales desde 4 kHz hasta 8 kHz espaciados a 1 Hz en el caso de los audios de 1 s. Una representación gráfica de lo anteriormente

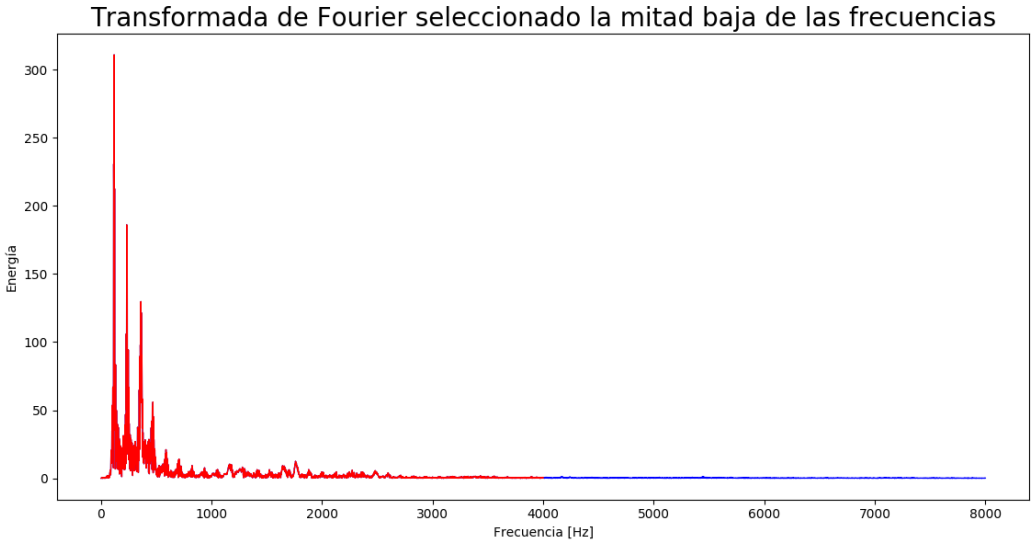

Figura 5.2: Representación gráfica de las componentes de baja frecuencia de la transformada rápida de Fourier para un audio de 1 s. La zona de color rojo indica la mitad baja de las frecuencias.

descrito se muestra en la figura 5.3, donde la sección coloreada de rojo corresponde a los datos de alta frecuencia de la transformada de Fourier.

- Espectrograma de frecuencia: en este trabajo a este tipo de dato se le conoce como Spect. Para esta representación de audio tanto en frecuencia como en tiempo, se utiliza el espectrograma de frecuencia descrito en la sección 2.3 y del cual se describen sus características en la subsección 2.4.3.

  Con este tipo de dato se generan espectrogramas de 0.5 s y 1 s utilizando ventanas de 25 ms con paso de 10 ms, lo que implica un traslape de 15 ms y 1024 puntos de FFT para los modelos entrenados para el módulo VGG. De estos 1024 puntos generados, son útiles sólo 512 y de estos 512 sólo se toman los 256 pertenecientes a las frecuencias bajas. Estos son elegidos porque como se muestra en el siguiente capítulo, el uso de frecuencias bajas y de todas las frecuencias no muestra mayores diferencias. Los 256 puntos utilizados representan las frecuencias de 0 a 4 kHz espaciados cada 15.625 Hz. A este tipo de dato se le conoce en este trabajo como Spect 256.

  Para los experimentos con la ResNet 50 siamés y para el módulo VGG se utiliza 1 s de información de audio, con ventanas de 25 ms con paso de 10 ms, lo que implica un traslape de 15 ms y 400 puntos de FFT de los cuales son útiles 200. De estos 200

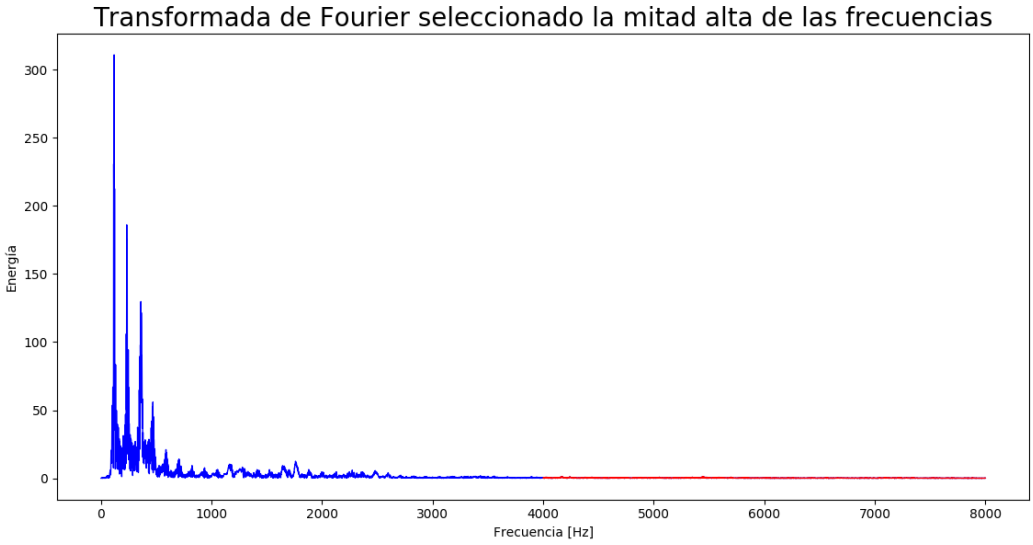

Figura 5.3: Representación gráfica de las componentes de alta frecuencia de la transformada rápida de Fourier para un audio de 1 s. La zona de color rojo indica la mitad alta de las frecuencias.

sólo se toman los 32 de las frecuencias más bajas, los cuales representan las frecuencias de 0 a 1280 Hz espaciados cada 40 Hz. La razón de la elección de los parámetros de 400 puntos de la FFT y 32 frecuencias de información es que la GPU con la que se cuenta para entrenar los modelos, con una red siamés de la ResNet 50 no soporta imágenes con tamaño mayor a 98 píxeles de ancho por 32 píxeles de alto, por lo que fue necesario encontrar un compromiso entre la cantidad de puntos de la FFT y la cantidad de frecuencias que se deseaba representar. El límite superior de frecuencia que se va a utilizar en este caso es de 1280 Hz. A este tipo de dato se le conoce en este trabajo como Spect 32. Los dos tipos de espectrogramas que se explicaron anteriormente, tras ser extraídos, se les aplicó un proceso de normalización.

En la figura 5.4 se observa un espectrograma de frecuencia con 1024 puntos de FFT, 1 s de contenido de audio, ventanas de 25 ms y paso de 10 ms (15 ms de traslape). En la figura 5.5 se observa el mismo espectrograma pero resaltando en rojo la zona de corte de 256 frecuencias. En la figura 5.6 se observa un espectrograma de frecuencia con 400 puntos de FFT, 1 s de contenido de audio, ventanas de 25 ms y paso de 10 ms (15 ms de traslape). En la figura 5.7 se observa el mismo espectrograma pero resaltando en rojo la zona de corte de 32 frecuencias.

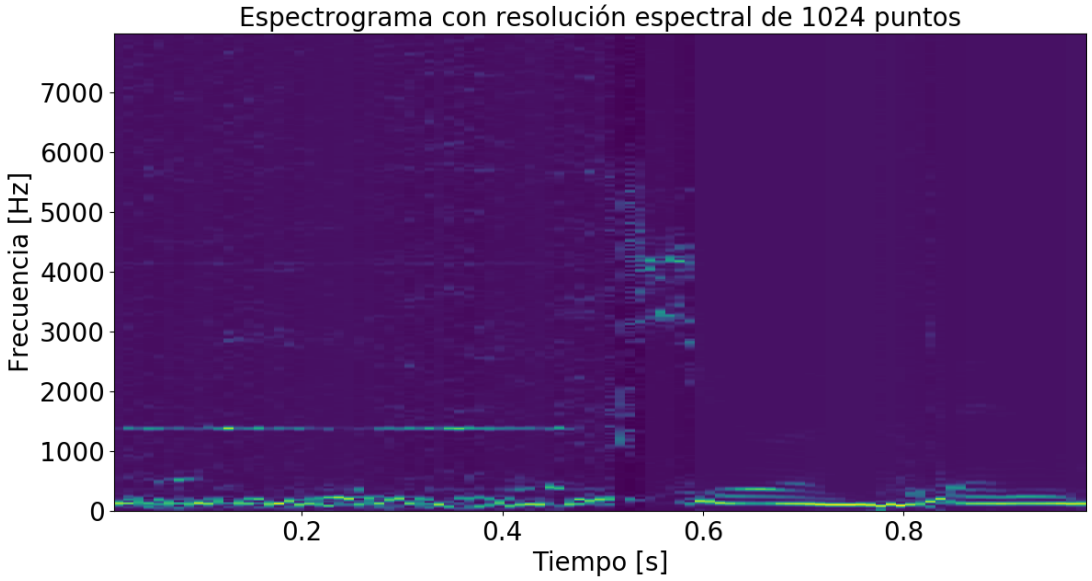

Figura 5.4: Espectrograma de frecuencias con 1024 puntos de FFT.

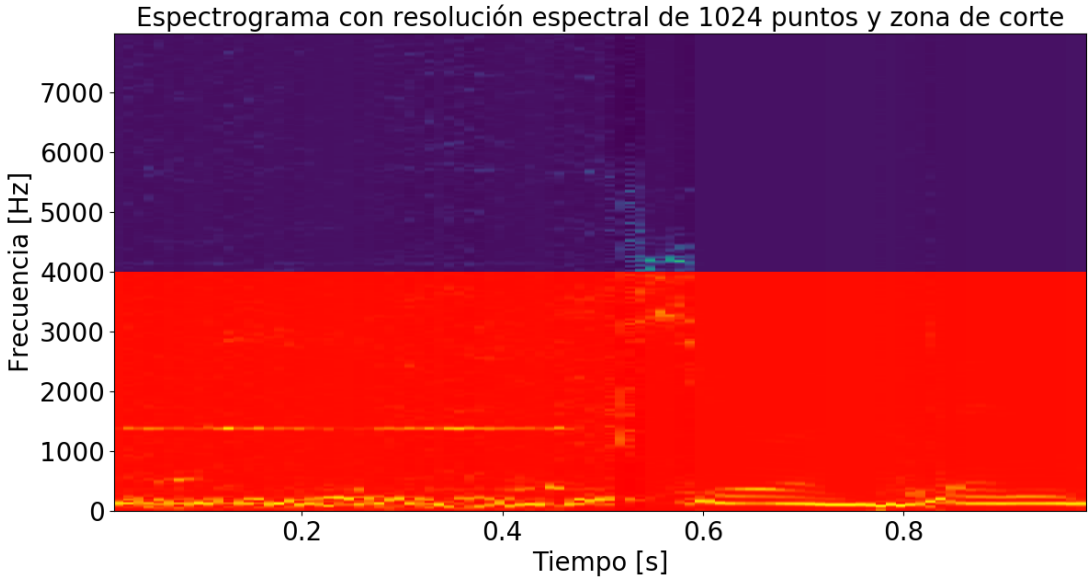

Figura 5.5: Espectrograma de frecuencias con 1024 puntos de FFT y zona de corte para 256 frecuencias. La zona de color rojo indica la zona de corte para 256 frecuencias.

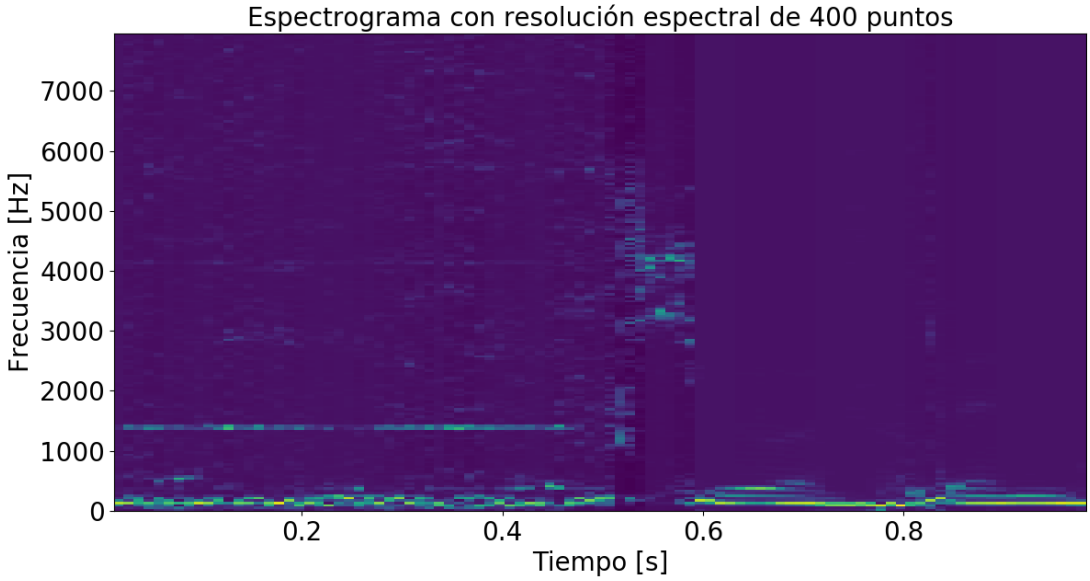

Figura 5.6: Espectrograma de frecuencias con 400 puntos de FFT.

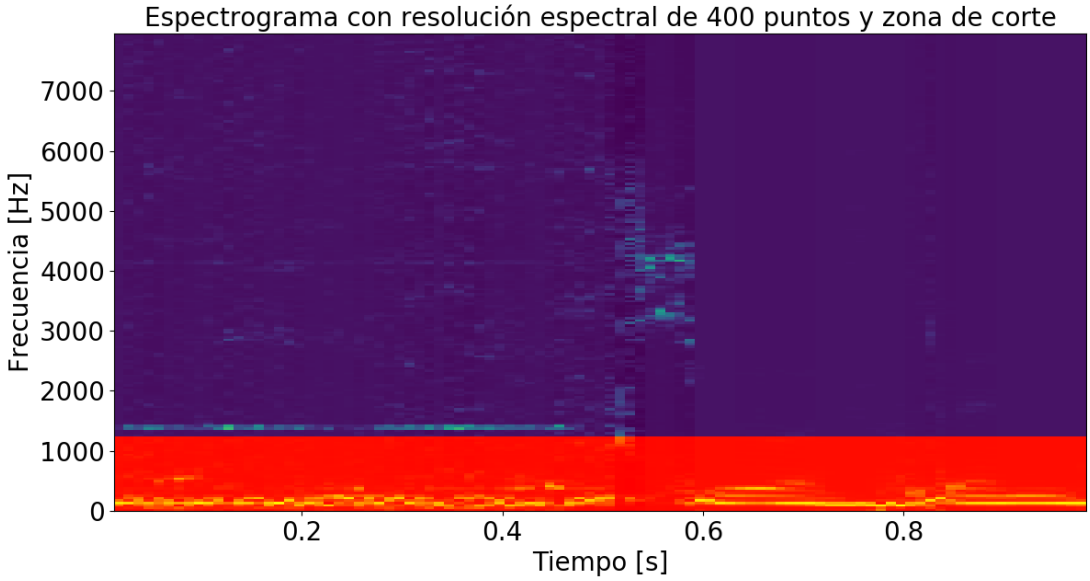

Figura 5.7: Espectrograma de frecuencias con 400 puntos de FFT y zona de corte para 32 frecuencias. La zona de color rojo indica la zona de corte para 32 frecuencias.

Ahora bien para generar los datos en sus diferentes tipos, se requiere hacer uso del VAD como se observa en la figura 5.8, de manera que ingresa la señal de audio a analizar y

que se debe transformar, se analiza su VAD y si supera el umbral deseado se realiza la transformación deseada, FFT, FFT LF, FFT HF o Spec, de este punto en adelante a este proceso de transformación se le conoce como Generación de dato.

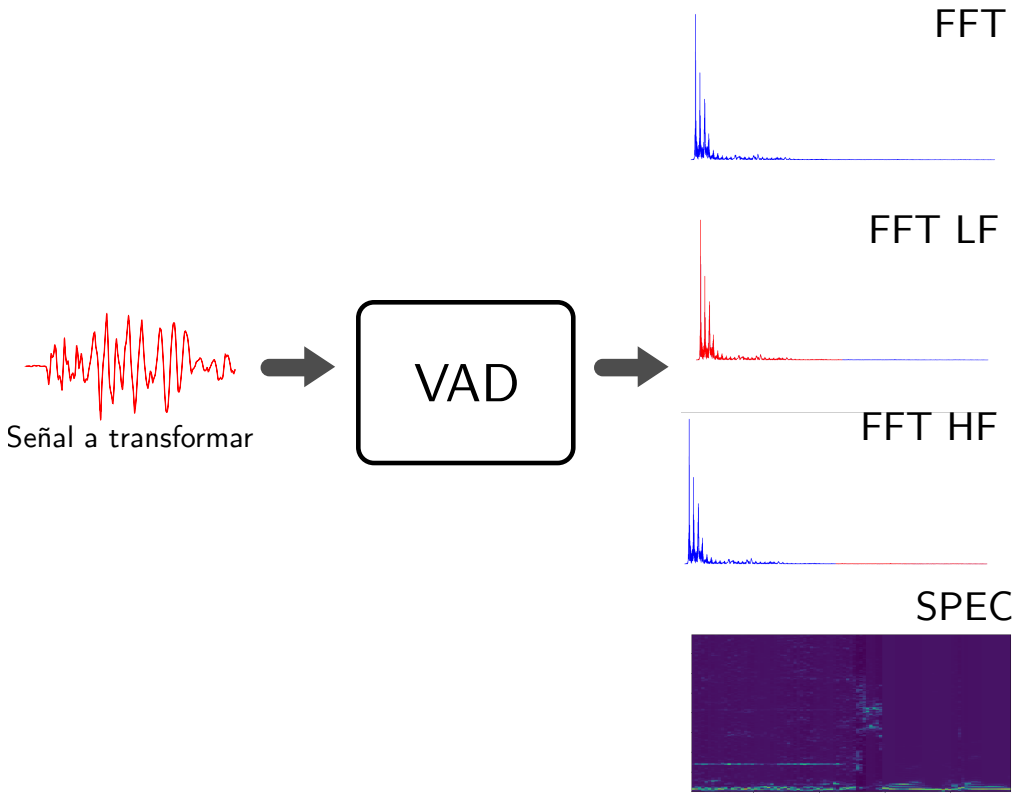

Figura 5.8: Diagrama para la generación de datos.

5.4. Selección de los datos de prueba

En este trabajo se conocen como datos de prueba aquellos datos con los cuales se prueba el sistema de verificación o de identificación mediante verificación que utiliza los pesos resultantes del entrenamiento de las máquinas de aprendizaje y que utilizan una base de datos local como información de los locutores conocidos.

Como datos de estudio se selecciona una cantidad variable de audios dependiendo de la prueba a realizar. Primero se seleccionan para un tipo de evaluación audios de la base de datos LibriSpeech. Estos datos provienen del 10 % de locutores de prueba; no se elige el total de locutores porque la aplicación para la que está pensada este sistema interactuará con alrededor de 10 locutores. Para un segundo tipo de evaluación se seleccionan audios de la

base de datos SITW, los cuales están grabados en diferentes ambientes y con diversos ruidos ambientales. De los 10 locutores elegidos aleatoriamente se extraen manualmente 20 trozos de audio de $10s$, en esta base de datos se tiene en algunas grabaciones más de un locutor por audio.

Una vez que se tienen los audios deseados se procede a realizar el proceso de verificación o identificación mediante verificación, donde los audios son convertidos al mismo tipo de dato con el que se entrenó el modelo, es decir transformadas en frecuencia o espectrogramas de frecuencia como se describió en la sección 5.3.

5.5. Selección del algoritmo de optimización

Las redes neuronales necesitan un algoritmo de optimización que les permita modificar los pesos de su red para aprender a realizar la tarea usando los datos de entrenamiento. Algunos de los algoritmos más conocidos para esta optimización son:

- Descenso estocástico del gradiente (*SGD*): método iterativo para minimizar una función objetivo.
- Propagación del valor cuadrático medio (*RMSProp*): método de tasa de aprendizaje adaptativa, dividiendo la tasa de aprendizaje por un promedio decreciente de gradientes cuadrados.
- Estimación de momento adaptativo (*ADAM*): método que calcula las tasa de aprendizaje de manera adaptativa, es similar a RMSProp pero además del promedio decreciente de gradientes cuadrados también guarda el promedio decreciente de gradientes anteriores.

En este trabajo se hace uso del *SGD* y del *RMSProp*. No se hizo uso del algoritmo de *ADAM* debido a que TensorFlow genera un error de memoria (*OOM*).

5.6. Selección de la arquitectura

La elección de las arquitecturas para los modelos a entrenar se realizó tomando en consideración dos criterios. Primero que el modelo pudiera ser entrenado utilizando una tarjeta gráfica NVIDIA GTX 1080, que es con la que se cuenta para la realización de este trabajo.

Segundo, que la red hubiese sido entrenada para imágenes con un desempeño superior al 80 % de exactitud.

Estos dos criterios se deben a que no todos los modelos pueden ser entrenados en esta tarjeta gráfica. De acuerdo con la experiencia adquirida en este trabajo, el límite que se halló para la GTX 1080 fue el entrenamiento de una ResNet 50 siamés con imágenes de entrada de 98 píxeles de ancho por 32 píxeles de alto. Imágenes o modelos más grandes generan un error de memoria (OOM). El segundo criterio es porque los espectrogramas son representaciones en forma de imágenes de un sólo canal de señales auditivas.

Se seleccionan dos tipos de redes convolucionales para la realización de este trabajo, que se conocen en este documento como Módulo VGG y ResNet 50 siamés.

Como se mencionó en la sección 3.5, una red siamés para verificación se compone principalmente de dos elementos, el primero realiza la extracción de características mientras que el segundo elemento se encarga de determinar la similitud de los vectores resultantes. Para este trabajo en particular, la arquitectura general propuesta puede ser observada en la figura 5.9, donde podemos observar cómo la tarea de verificación es abordada como un problema de identificación con la existencia de dos clases.

La clase 0 determina si dos señales son iguales, mientras que la clase 1 indica que son diferentes. Todas las redes aquí propuestas se componen de un conjunto de entradas x_i, x'_i, las cuales producen una salida y que indica si son o no iguales.

Como es posible observar en la figura 5.9, la red se encarga de realizar dos tareas fundamentales para la verificación: 1) extraer características de los datos de entrada y 2) internamente encontrar una función que determine la similitud entre ambas entradas.

En general, las arquitecturas en sus primeras capas se componen principalmente de capas convolucionales y, como última capa, se les coloca una red completamente conectada (*fully-conected*). Los resultados de esta se pasan por una función *SoftMax* para finalmente extraer las probabilidades de ser iguales o no.

Dado que se requiere extraer la probabilidad de ser o no ser el locutor contra el que se está verificando, la última capa de las dos redes propuestas es pasada por una función *SoftMax*. La función *SoftMax* en este caso permite evaluar tanto la probabilidad de ser el mismo locutor contra el que se verifica (salida ideal 0 1) o no serlo (salida ideal 1 0).

La salida anteriormente descrita sólo es colapsada a valores 0 1 o 1 0 cuando se desea hallar la exactitud del modelo. La salida de la función *SoftMax* es utilizada cuando se ejecuta el sistema completo y se desea calcular con que probabilidad el locutor contra el que se compara es o no es el indicado. Para calcular la pérdida se emplea la salida de la capa sin

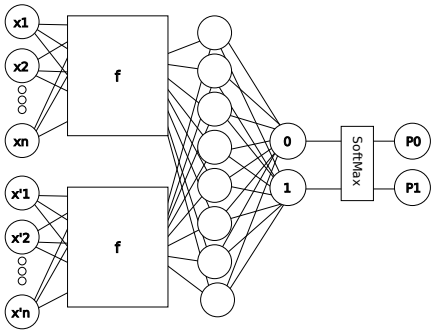

Figura 5.9: Arquitectura general propuesta. Las entradas se definen como $\mathbf{x}$ y $\mathbf{x}'$, f representa la arquitectura implementada, 0 y 1 representan las clases, (0 iguales y 1 diferentes), $P0$ representa la probabilidad de pertenecer a la clase 0 mientras que $P1$ representa la probabilidad de pertenecer a la clase 1.

ser pasada por la función *SoftMax* ya que la función de pérdida a utilizar del marco de trabajo TensorFlow [Abadi et al., 2015] es `softmax_cross_entropy_with_logits`, la cual mide el error de probabilidad en tareas de clasificación discretas en las que las clases son mutuamente excluyentes y que realiza un *SoftMax* internamente.

5.6.1. Módulo VGG

La red convolucional VGG 16 ha sido probada para imágenes, sin embargo, el entrenamiento de esta red completa de forma siamés no pudo ser llevado a cabo en la GTX 1080, por lo que se propuso un modelo de una red siamés inspirado en la VGG 16. Este modelo se puede observar en la figura 5.10. Cuenta con 4 capas convolucionales para cada parte de la red siamés, 2 capas de generalización y 3 capas completamente conectadas para clasificación, que corresponden a las primeras 4 capas y a las últimas 3 de la VGG 16. Las capas convolucionales utilizan filtros de 3×3 y 64 de ellos para las 2 primeras capas y 128 para las capas 3 y 4. Las capas intermedias utilizan como función de activación ReLU.

El entrenamiento de esta red se realiza por lotes de 50 muestras.

5.6.2. ResNet 50 siamés

Utilizando la red convolucional ResNet 50 se crea un modelo siamés, esta se puede observar en la figura 5.11 y cuenta con 1 capa convolucional con 64 filtros de 7×7, 16 bloques de cuello de botella y 2 capas completamente conectadas. Las capas intermedias utilizan normalización por lotes y ReLU como función de activación. El entrenamiento de esta red se realiza por lotes de 10 muestras.

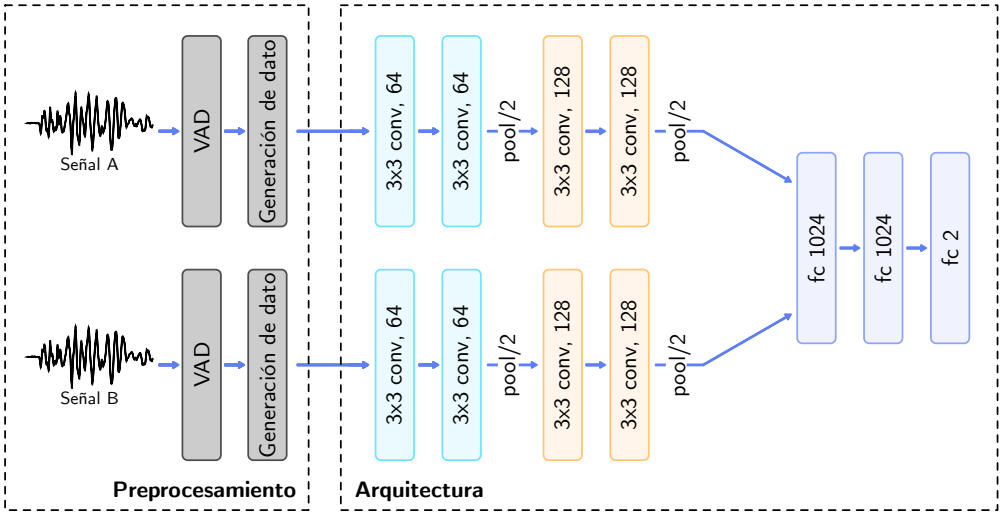

Figura 5.10: Red convolucional conocida en este documento como módulo VGG.

La red dada la cantidad de pesos que maneja puede tardar mucho tiempo en converger, por lo que se utiliza una red de inicialización y que tiene la misma estructura que la ResNet 50 siamés, pero su última capa se modela como una capa multitarea donde se realiza la verificación e identificación de locutor, esto permite que posteriormente se pueda continuar sólo con el entrenamiento del verificador. Esta red se entrena sólo con 100 locutores y no con 1987 locutores que tiene la sección de entrenamiento y se puede observar en la figura 5.12.

5.7. Entrenamiento de modelos

Realizando combinaciones de los cuatro tipos de datos de entrenamiento, los dos tipos de arquitecturas y los dos tipos de optimizadores se propusieron los modelos a entrenar que se pueden observar en la tabla 5.2. Cabe precisar que los entrenamientos se realizaron en el orden que se muestra en la tabla y los modelos que se entrenan dependen de los resultados de sus predecesores, es decir, a medida que se avanzó en el entrenamiento de los modelos se continuó con los tipos de datos que presentaban los mejores resultados o cuyos resultados eran los más consistentes.

El nombre de los modelos que se observa en la tabla 5.2 se compone primero del tipo de arquitectura que se utiliza (M para el Módulo VGG y R para la ResNet 50 siamés), luego del tipo de dato que se utiliza para entrenarlo (FFT para la F, L para la FFT LF, H para la FFT HF y S para el espectrograma de frecuencia), posteriormente se utiliza una letra para

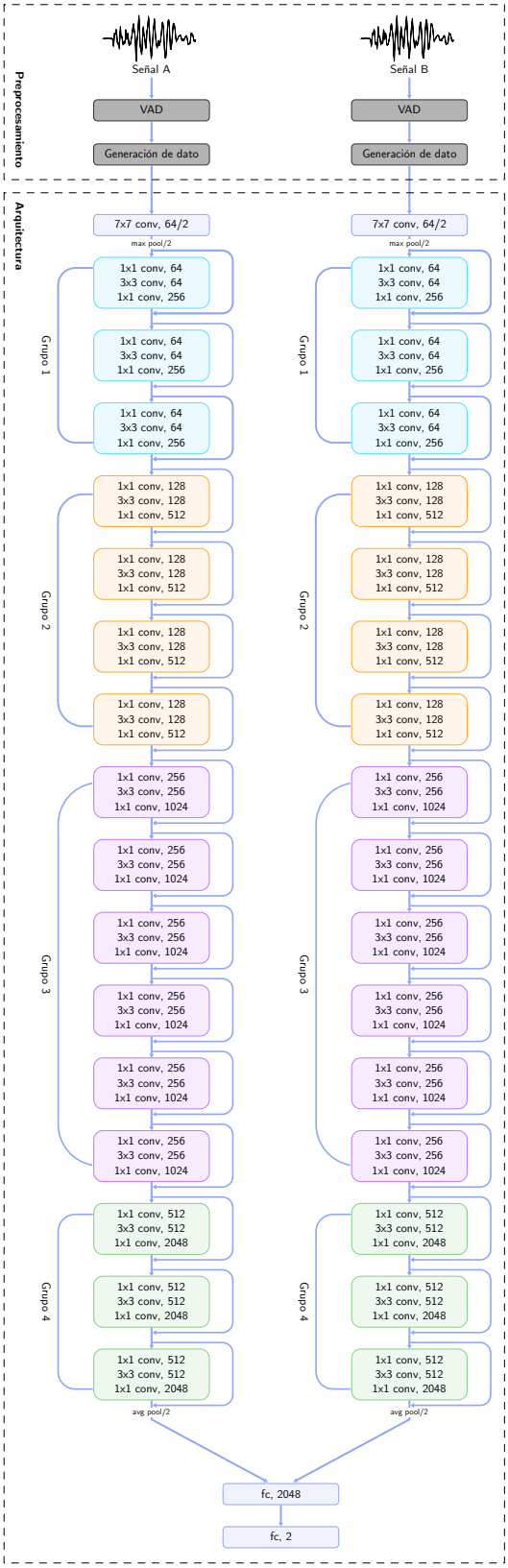

Figura 5.11: Red convolucional conocida en este documento como ResNet 50 siamés.

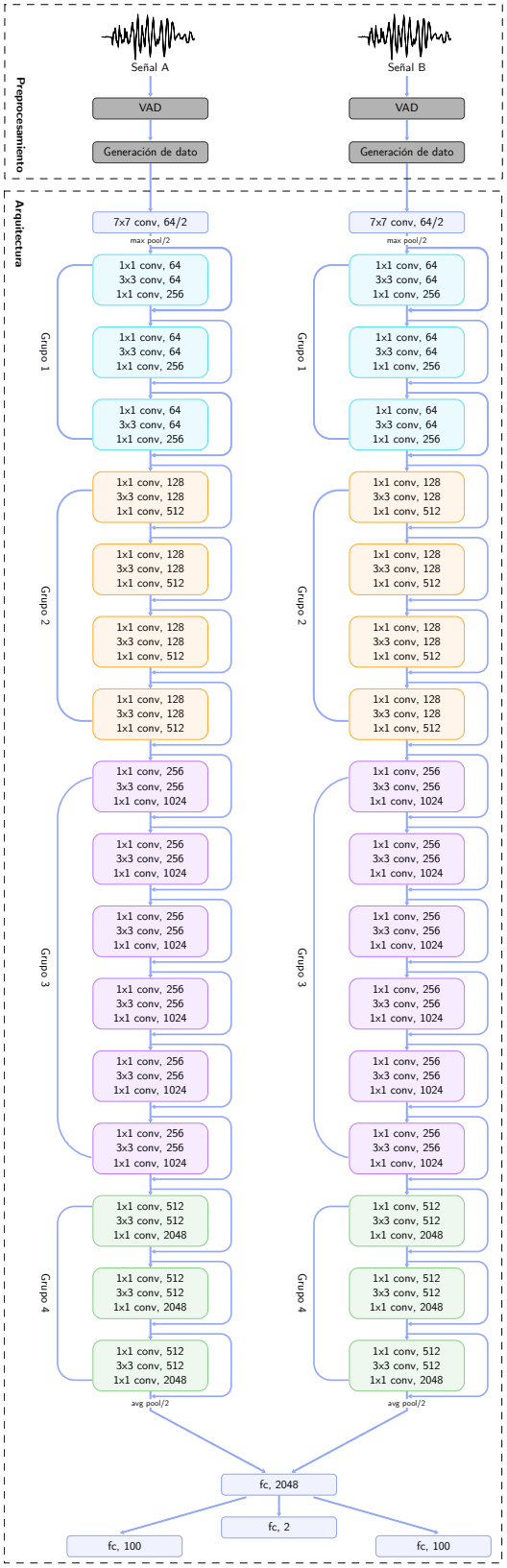

Figura 5.12: Red convolucional de inicialización para la ResNet 50 siamés.

indicar el tipo de optimizador (G para *SGD* y R para *RMSProp*), luego se utiliza un número para indicar la cantidad de segundos de audio con los que se entrenó, después se utiliza un número para indicar la cantidad de frecuencias utilizadas, luego se indica la tasa de aprendizaje utilizada, la cual se escribe si tiene más de dos cifras decimales como una potencia de 10 y por último se indica el factor de afectación del error, un ejemplo se muestra en la figura 5.13.

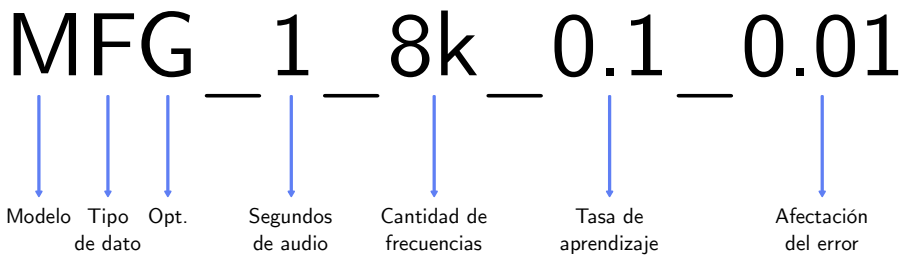

Figura 5.13: Ejemplo de la creación de un nombre de modelo.

Los modelos se entrenan por épocas. Una época consta del barrido de una lista de audios, sin embargo las muestras de audio de los locutores en cada iteración o época son elegidas aleatoriamente.

Para el entrenamiento de cada uno de los modelos se generan $800,000$ datos de entrenamiento por época con los locutores que previamente se seleccionaron para entrenamiento. Estos datos constan de $400,000$ datos cuyas entradas son del mismo locutor y $400,000$ datos cuyas entradas son de locutores diferentes, esto con el objetivo de que los datos de entrenamiento estén balanceados. Los trozos de audio que se toman son elegidos aleatoriamente y a pesar de que es la misma lista de audios a través de las épocas, los datos de entrenamiento no son iguales por la elección aleatoria del trozo de audio con el que se va a entrenar en cada iteración.

Los modelos que utilizan como arquitectura la ResNet 50 siamés tienen un preentrenamiento de identificación y verificación de 10 épocas con una tasa de aprendizaje de 0.01, posteriormente se realiza el entrenamiento del modelo de verificación con 10 épocas con una tasa de aprendizaje de 0.01 y 10 épocas con una tasa de aprendizaje del 0.001.

Los modelos que utilizan como arquitectura el Módulo VGG se entrenan sólo durante 15 épocas con la tasa de aprendizaje indicada en el tabla.

En diversos modelos se utiliza un factor de afectación del error. Esto se debe a que en algunos modelos el error cuando se iniciaba el entrenamiento aumentaba hasta el límite superior del valor de la variable antes de iniciar su descenso, lo que generaba una saturación

del error máximo que se volvía infinito, por esto fue necesario afectar el error por algún valor.

La ResNet 50 siamés se entrenó con tres variaciones:

ResNet 1: El modelo fue entrenado con muchos ejemplos iguales que provienen del mismo audio pero cuya información es tomada de una ventana aleatoria de este.

ResNet 2: El modelo fue entrenado con todos los ejemplos iguales que provenían del mismo locutor pero de diferentes audios.

ResNet 3: El modelo utiliza pesos fijos de la fase de preentrenamiento en toda la red excepto en la última capa.

Arquitectura	**Dato**	**Seg.**	**Frec.**	**Opt.**	**TA**	**Fact.**	**Nombre**
Módulo VGG	FFT	1	8000	*SGD*	0.1	0.01	MFG_1_8k_0.1_0.01
Módulo VGG	FFT LF	1	4000	*SGD*	0.1	0.01	MLG_1_4k_0.1_0.01
Módulo VGG	FFT LF	1	4000	*SGD*	0.1	0.1	MLG_1_4k_0.1_0.1
Módulo VGG	FFT HF	1	4000	*SGD*	0.1	0.01	MHG_1_4k_0.1_0.01
Módulo VGG	SPEC	1	256	*SGD*	0.01	0.01	MSG_1_256_0.01_0.01
Módulo VGG	SPEC	1	256	*SGD*	0.01	1	MSG_1_256_0.01_1
Módulo VGG	FFT LF	1	4000	*RMSProp*	0.0001	0.01	MLR_1_4k_10^{-4}_0.01
Módulo VGG	FFT LF	1	4000	*RMSProp*	0.0001	1	MLR_1_4k_10^{-4}_1
Módulo VGG	SPEC	1	256	*RMSProp*	0.0001	0.01	MSR_1_256_10^{-4}_0.01
Módulo VGG	SPEC	1	256	*RMSProp*	0.0001	1	MSR_1_256_10^{-4}_1
Módulo VGG	FFT LF	0.5	4000	*SGD*	0.1	0.01	MLG_0.5_4k_0.1_0.01
Módulo VGG	FFT LF	0.5	4000	*RMSProp*	0.0001	0.01	MLR_0.5_4k_10^{-4}_0.01
Módulo VGG	SPEC	0.5	256	*SGD*	0.01	0.01	MSG_0.5_256_0.01_0.01
Módulo VGG	SPEC	0.5	256	*RMSProp*	0.0001	0.01	MSR_0.5_256_10^{-4}_0.01
Módulo VGG	FFT	2	8000	*SGD*	0.1	0.01	MFG_2_8k_0.1_0.01
Módulo VGG	FFT LF	2	4000	*SGD*	0.1	0.01	MLG_2_4k_0.1_0.01
Módulo VGG	FFT HF	2	4000	*SGD*	0.1	0.01	MHG_2_4k_0.1_0.01
ResNet 1	SPEC	1	32	*SGD*	0.01-0.001	1	RSG_1_32_0.01_1
ResNet 2	SPEC	1	32	*SGD*	0.01-0.001	1	RSG_1_32_0.01_1
ResNet 3	SPEC	1	32	*SGD*	0.01-0.001	1	RSG_1_32_0.01_1
Módulo VGG	SPEC	1	32	*SGD*	0.01	1	MSG_1_32_0.01_1

Tabla 5.2: Listado de los modelos entrenados, donde Arquitectura se refiere a la arquitectura a utilizar (Módulo VGG o ResNet), Dato se refiere al tipo de dato con el que se entrena la red (FFT, FFT LF FFT HF, Spec), Seg. se refiere a la cantidad de segundos de audio que se utiliza para el modelo, Frec. indica la cantidad de frecuencias que se van a utilizar, Opt. indica el tipo de optimizador a utilizar, TA indica la tasa de aprendizaje utilizada, Fact. indica el factor de afectación para el error del modelo y Nombre será en nombre del modelo a utilizar de este punto en adelante.)

5.8. Selección de modelos

De los resultados de los diferentes modelos de verificación propuestos en la tabla 5.2, se eligen los 3 primeros en exactitud de prueba.

Para todas las configuraciones y evaluaciones se busca un desempeño superior al 80 % ya que el sistema más cercano al que se desea implementar fue el de [Nagrani et al., 2017], que tuvo una exactitud del 80.5 % para el top 1 y en este trabajo sólo se va a comparar con el top 1 (la primer clase en exactitud es contra la que se determina si el sistema identificó a la persona de forma correcta). La métrica del top 5 (que entre las 5 primeras clases aparezca la clase del locutor correcto), no es adecuada para este trabajo ya que la cantidad de locutores con los que interactúa un robot de servicio es limitada.

Con los modelos resultantes de esta selección, se realizan las evaluaciones descritas de aquí en adelante.

5.9. Evaluación de los modelos como verificadores

De los modelos elegidos con el criterio de la sección 5.8, se evaluan los verificadores con 1000 muestras, donde 500 muestras tienen como entradas audios de la misma clase de locutor y 500 muestras tienen como entradas audios de diferentes locutores. Estos audios se seleccionaron aleatoriamente de la sección de prueba de la base de datos LibriSpeech. Este proceso se realiza 10 veces por cada verificador y se obtiene el promedio de verdaderos positivos, verdaderos negativos, falsos positivos y falsos negativos de estos resultados.

Para evaluar cada verificador se calcula la precisión, la exhaustividad (*Recall*), la métrica F1 (*F1 Score*) y la exactitud (*accuracy*) de los resultados promedio.

A continuación se describen brevemente estas métricas, se da un concepto breve de ellas y se indica la forma de calcularlas en términos de verdaderos positivos (VP, aquellos valores que debían ser positivos y efectivamente fueron positivos), verdaderos negativos (VN, aquellos valores que debían ser negativos y efectivamente fueron negativos), falsos positivos (FP, aquellos valores que debían dar un resultado negativo pero fueron positivos) y falsos negativos (FN, aquellos valores que debían ser positivos y fueron negativos):

- Precisión: Es la relación entre la cantidad de datos positivos reales con respecto al total de datos que fueron marcados positivos. Se calcula mediante el cociente de los verdaderos positivos sobre la suma de los verdaderos positivos más los falsos positivos así: $VP/(VP + FP)$, si un modelo no tuvieran ningún falso positivo, sería un modelo muy

preciso incluso aunque tuviese muchos falsos negativos. Para este trabajo ser preciso es indicar correctamente quién es locutor.

- Exhaustividad: Es la relación entre la cantidad de datos positivos reales con respecto al total de datos que debían ser positivos. Se calcula mediante el cociente de los verdaderos positivos sobre la suma de los verdaderos positivos más los falsos negativos así: $VP/(VP + FN)$, si el modelo no tuviera ningún falso negativo, sería un modelo muy exhaustivo incluso aunque tuviese muchos falsos positivos. Para este trabajo ser exhaustivo es indicar correctamente a la mayor cantidad de locutores posibles.

- F1: es una combinación de la medida de precisión y exhaustividad y se calcula por el cociente de dos veces la precisión por la exhaustividad sobre la precisión más la exhaustividad así: $2(precisión * exhaustividad)/(precisión + exhaustividad)$. Esta medida es aproximadamente la media de las dos medidas cuando están cerca. Para este trabajo la F1 dirá que tan preciso y exhaustivo es el modelo.

- Exactitud: Es la cantidad de datos positivos reales y negativos reales con respecto a todos los datos. Se calcula mediante el cociente de la suma de verdaderos positivos y verdaderos negativos y la suma de verdaderos positivos, verdaderos negativos, falsos positivos y falsos negativos así: $(VP+VN)/(VP+VN+FP+FN)$. Para este trabajo ser exacto implica que el modelo es bueno diciendo tanto quiénes son los locutores como quiénes no lo son.

5.10. Evaluación de los modelos como identificadores

Posteriormente se procede a evaluar los 3 modelos como identificadores, si bien los modelos se crearon como verificadores, el objetivo final es que funcionen correctamente como identificadores y que puedan discriminar entre una cantidad de locutores.

Para la evaluación de modelos fuera de línea con respecto al entrenamiento, de este punto en adelante se utiliza el modelo descrito a continuación. El verificador tiene la misma construcción de la sección 5.7, pero ahora se utiliza una base de datos como conocimiento a priori. Se consideran como locutores conocidos, aquellos de los cuales se tiene alguna información en la base de datos y como desconocidos aquellos de los cuales no se tiene ninguna información. Un diagrama del sistema se puede observar en la figura 5.14.

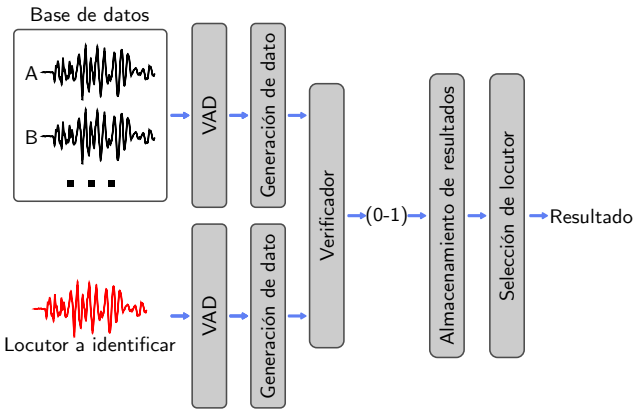

Figura 5.14: Diagrama del sistema de identificación del locutor.

Para las evaluaciones como identificadores se tienen dos opciones, se puede tomar como clase aquella que tenga el resultado de verificación más alto por audio o se puede tomar como clase aquella que tenga el resultado promedio más alto por clase, un pequeño ejemplo se describe utilizando la tabla 5.3. En esta tabla se muestran los resultados de verificar un audio contra 6 audios conocidos, 3 de la clase A y 3 de la clase B, el audio con el resultado más alto es el A1 con un 0.98. Si se extrae el resultado promedio por clase, la clase A tendría un 0.56 mientras que la B tendría un 0.95. Si se usa como clase aquella con resultado del audio más alto se elegirá la clase A, mientras que si se usa como clase aquella con el resultado promedio por clase más alto se elegirá la clase B.

Audio	**A1**	**A2**	**A3**	**B1**	**B2**	**B3**
Clase	**A**	**A**	**A**	**B**	**B**	**B**
Resultado de verificación	0.98	0.4	0.3	0.94	0.95	0.96

Tabla 5.3: Ejemplo de resultado de verificación.

De acuerdo con resultados hallados de manera preliminar, se decide utilizar la identificación cuya clase es decidida por su promedio, este tipo de identificación es más efectiva que la identificación utilizando la clase del audio con el resultado más alto, ya que existe una probabilidad de que el trozo aleatorio del audio elegido no contenga información que permita discriminar claramente al locutor, sin embargo al promediar la clase, es posible que el locutor responda bien a todos los audios de su misma clase.

La medida que se observará en estos resultados es la exactitud del sistema. No se analiza precisión, exhaustividad y F1 porque el resultado para todas estas métricas es igual, dado que los experimentos se clasifican en una única clase, por cada falso positivo que se genere,

también se genera un falso negativo.

Para identificación de locutores se proponen dos experimentos, en uno se toma una cantidad de clases conocidas y una clase adicional que será conocida como "locutor desconocido" la cual está compuesta de 20 audios de 20 locutores diferentes que no van a estar agregados dentro de la base de datos, de cada clase tanto conocida como desconocida se identifican 20 audios. Se evalúa el desempeño de los verificadores como identificadores cuando se tienen diferente cantidad de locutores conocidos (clases) que poseen entre 1 y 10 audios a priori por locutor conocido en la base de datos, para trazar un mapa de calor que muestre cómo varía el desempeño del sistema a medida que se tienen más clases y más audios por clase. En la tabla 5.4 se describe brevemente cómo se realizan los experimentos y cómo se aumenta el número de clases a identificar, esto se realiza con entre 1 y 10 audios por locutor conocido.

	Clases Conocidas											
Exp.	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**D**	**Total**
1	X	-	-	-	-	-	-	-	-	-	X	40
2	X	X	-	-	-	-	-	-	-	-	X	60
3	X	X	X	-	-	-	-	-	-	-	X	80
4	X	X	X	X	-	-	-	-	-	-	X	100
5	X	X	X	X	X	-	-	-	-	-	X	120
6	X	X	X	X	X	X	-	-	-	-	X	140
7	X	X	X	X	X	X	X	-	-	-	X	160
8	X	X	X	X	X	X	X	X	-	-	X	180
9	X	X	X	X	X	X	X	X	X	-	X	200
10	X	X	X	X	X	X	X	X	X	X	X	220

Tabla 5.4: Descripción de los experimentos para evaluar el sistema como identificador, donde Exp. indica el número del experimento a realizar, D indica la clase desconocido, Total indica el número total de audios utilizados a identificar en el experimento. Las X en las casillas indican que la clase, a excepción de la clase desconocido, va a ser conocida para el experimento y tendrá audios en la base de datos y - indica que no se conoce la clase y no se va a solicitar identificación con audios de ésta.

Para un segundo experimento, se procede a evaluar con los 3 modelos elegidos en un escenario en el cual se inicia con una cantidad de locutores desconocidos que son agregados por el sistema a medida que se interactúa con ellos. La diferencia entre este experimento y el anterior es que aquí la cantidad de personas desconocidas va disminuyendo a medida que se agregan locutores, hasta que finalmente el sistema conoce a todas las personas con quienes va a interactuar, lo que se busca evaluar es el desempeño del sistema cuando se van conociendo los locutores con los que interactúa; en el experimento anteriormente nombrado cambia el

número de clases a identificar, siempre se va a contar con la clase de locutor desconocido y lo que se va a evaluar es que tan bien se comporta el modelo con más o menos clases.

Un diagrama ejemplo de ingreso de locutores desconocidos se puede observar en la figura 5.15. En este ejemplo llega una señal del locutor C para ser reconocida, sin embargo en la base de datos sólo se tienen audios de los locutores A y B, por lo que el resultado del proceso de identificación indica que es un locutor desconocido, al obtener este resultado se agrega el audio que se recibió como entrada a la base de datos como un nuevo locutor. Posteriormente cuando vuelve a llegar una señal del locutor C, el sistema lo identifica correctamente.

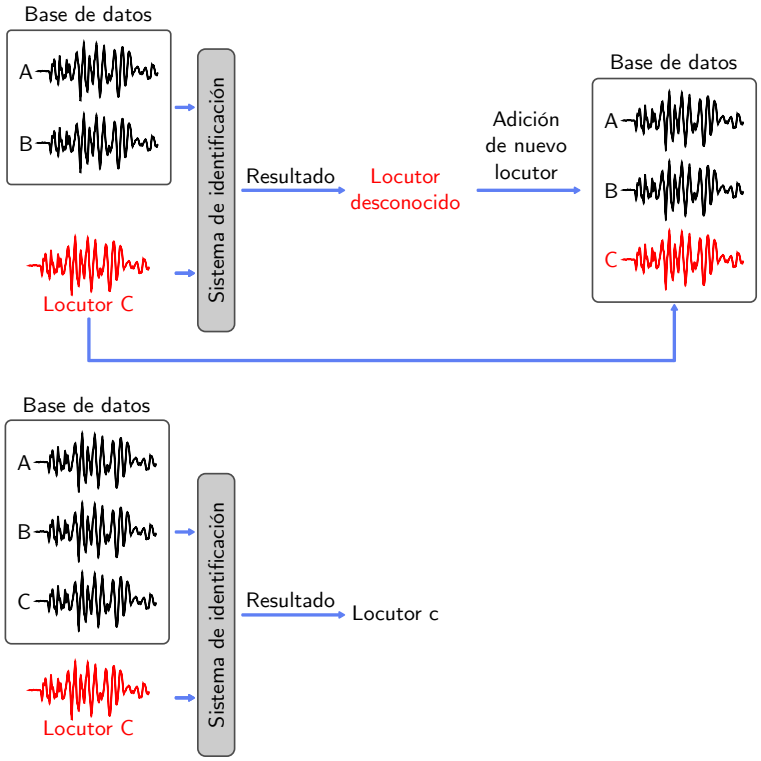

Figura 5.15: Diagrama del sistema de identificación del locutor a medida que ingresan locutores.

Para esta evaluación se propone un escenario en el cual un robot de servicio, debe reconocer a 10 locutores que habitan una casa, o a 10 comensales que entran a un restaurante. El mundo, en este ejemplo, se limitará a un máximo de 10 personas, es decir, si el robot conoce a 1 persona, cualquiera de las 9 restantes que le hable debe ser considerado desconocido; o si el robot conoce a 5 personas, las 5 personas restantes le serán desconocidas. Una vez que agregue a los 10 locutores que interactuarán con él, no habrán más locuciones de personas desconocidas y sólo estará reconociendo la voz de esos 10 locutores.

Para evaluar el sistema se hacen uso de 200 audios, 20 audios por cada uno de los 10

locutores que se van a reconocer. Inicialmente el robot conoce a 1 persona, por lo que habrán 20 audios de personas conocidas y 180 audios de personas conocidas, posteriormente con los mismos audios y conociendo a 2 personas, habrán 40 audios de personas conocidas y 160 audios de personas desconocidas y así sucesivamente como se observa en la tabla 5.5.

	Clases Conocidas											
Exp.	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Con.**	**Descon.**
1	X	D	D	D	D	D	D	D	D	D	20	180
2	X	X	D	D	D	D	D	D	D	D	40	160
3	X	X	X	D	D	D	D	D	D	D	60	140
4	X	X	X	X	D	D	D	D	D	D	80	120
5	X	X	X	X	X	D	D	D	D	D	100	100
6	X	X	X	X	X	X	D	D	D	D	120	80
7	X	X	X	X	X	X	X	D	D	D	140	60
8	X	X	X	X	X	X	X	X	D	D	160	40
9	X	X	X	X	X	X	X	X	X	D	180	20
10	X	X	X	X	X	X	X	X	X	X	200	0

Tabla 5.5: Descripción de los experimentos para evaluar el sistema a medida que conoce a sus locutores, donde Exp. indica el número del experimento a realizar, Con. indica la cantidad de audios a evaluar que son de locutores conocidos, Descon. indica la cantidad de audios a evaluar que son de locutores desconocidos. Las X en las casillas indican que la clase va a ser conocida para el experimento y tendrá audios en la base de datos y D indica que no se conoce la clase y sus audios serán considerados desconocidos.

La evaluación se realiza de esta forma ya que una vez se conozca 1 locutor, cualquiera de los otros 9 locutores, con cualquier tipo de frase hablada puede intervenir, al igual que si conoce 4 personas, cualquiera de las otras 6 personas puede interactuar con él en cualquier momento y debe poder medirse que tanto se modifica la exactitud del sistema a medida que se ingresan los locutores con los que va a interactuar.

Para los experimentos se evalúa el desempeño a medida que aumenta el número de locutores conocidos teniendo entre 1 y 10 audios a priori por locutor conocido en la base de datos, de esta forma se puede trazar un mapa de calor que muestre cómo varía el desempeño del sistema a medida que se conocen los locutores con los que se interactúa y la cantidad de audios que se tienen por cada uno de estos.

Capítulo 6

Evaluación y resultados

Para este trabajo se realizaron una serie de evaluaciones de los modelos propuestos en el capítulo 5. Estos resultados serán visualizados y analizados en este capítulo.

Se proponen las siguientes secciones:

1. Resultados de las máquinas de aprendizaje para su entrenamiento, validación y prueba con la base de datos LibriSpeech.
2. Selección de modelos.
3. Evaluación de los modelos como verificadores.
4. Evaluación de los modelos como identificadores.
5. Duración de los modelos de simulación.
6. Resumen de los resultados.

6.1. Resultados de los modelos de verificación

En la tabla 6.1 se muestran los resultados de los modelos de verificación propuestos en la sección 5.7.

Nombre	Entrenamiento	Validación	Prueba	Pos.
MFG_1_8k_0.1_0.01	93.16 %	86.01 %	87.45 %	9
MLG_1_4k_0.1_0.01	93.08 %	86.22 %	87.08 %	10
MLG_1_4k_0.1_0.1	97.30 %	89.22 %	89.52 %	6
MHG_1_4k_0.1_0.01	72.07 %	70.52 %	72.55 %	18
MSG_1_256_0.01_0.01	80.97 %	77.42 %	81.89 %	13
MSG_1_256_0.01_1	93.57 %	89.57 %	91.17 %	3
MLR_1_4k_10^{-4}_0.01	98.27 %	89.76 %	89.78 %	5
MLR_1_4k_10^{-4}_1	98.21 %	89.40 %	88.51 %	8
MSR_1_256_10^{-4}_0.01	92.26 %	85.58 %	88.52 %	7
MSR_1_256_10^{-4}_1	89.24 %	77.16 %	75.62 %	17
MLG_0.5_4k_0.1_0.01	89.03 %	82.38 %	83.09 %	21
MLR_0.5_4k_10^{-4}_0.01	97.67 %	86.40 %	86.16 %	11
MSG_0.5_256_0.01_0.01	74.53 %	71.51 %	76.13 %	16
MSR_0.5_256_10^{-4}_0.01	89.37 %	79.85 %	80.97 %	14
MFG_2_8k_0.1_0.01	96.16 %	90.30 %	90.80 %	4
MLG_2_4k_0.1_0.01	96.06 %	89.27 %	90.76 %	19
MHG_2_4k_0.1_0.01	78.47 %	75.94 %	76.30 %	15
RSG_1_32_0.01_1^1	95.85 %	93.19 %	92.39 %	1
RSG_1_32_0.01_1^2	90.01 %	84.00 %	84.13 %	20
RSG_1_32_0.01_1^3	86.02 %	81.84 %	82.23 %	12
MSG_1_32_0.01_1	92.26 %	89.27 %	92.22 %	2

Tabla 6.1: Resultados de los modelos entrenados, donde Nombre es el nombre clave del modelo descrito en la sección 5.7, Entrenamiento, Validación y Prueba la exactitud de cada modelo para ese conjunto de datos de la base de datos LibriSpeech y Pos. indica la posición que ocuparon en la exactitud de prueba, siendo el modelo con la exactitud más alta el número 1 y el modelo con la exactitud más baja el número 21.

En los resultados de la tabla 6.1 se puede ver que el uso de una cantidad reducida de frecuencias bajas tiene una exactitud con menos del 1 % de diferencia con respecto a los modelos que utilizan todas las frecuencias, como se observa en los primeros experimentos. Por otra parte, el uso de la mitad de las frecuencias altas tuvo una exactitud 17 % menor que el modelo que utiliza todas las frecuencias. De estos experimentos se puede demostrar empíricamente que las frecuencias bajas, que son aquellas con la mayor cantidad de energía para la voz humana, son las que presentan más información relevante al modelo de verificación.

El uso del optimizador de descenso estocástico del gradiente presenta mejores resultados que aquellos obtenidos con la propagación del valor cuadrático medio (ambos descritos en la sección 5.5). A pesar de que los modelos que utilizan propagación del valor cuadrático medio tienen un resultado más alto en entrenamiento, no lo tienen en prueba, es posible que con

este optimizador haya un sobreajuste a los datos de entrenamiento.

En los resultados de entrenamiento del módulo VGG con espectrogramas de un segundo, y 32 o 256 frecuencias, se observó que la diferencia de exactitud en prueba es tan solo del 1 % de exactitud, dando mejores resultados el uso de 32 frecuencias que de 256, lo que nuevamente refuerza empíricamente la evidencia de que las componentes bajas de frecuencia, que son aquellas con mayor energía, son las más relevantes para la verificación.

En la tabla 6.1 también se mostró que el uso de ventanas de 0.5 segundos empeora el desempeño del sistema alrededor de un 5 % con respecto al uso de ventanas de 1 segundo y que el uso de ventanas de 2 segundos con datos solamente de frecuencia, si bien puede tener un desempeño con una mejora del 2.5 % en exactitud de verificación, implica mayor tiempo de entrenamiento; mayor contenido auditivo del locutor que se está identificando y no llegó a los 3 primeros lugares con respecto a todos los modelos.

También se puede evidenciar que el uso de una arquitectura más compleja como es el caso de la ResNet 50 siamés no garantiza que su desempeño sea mucho mayor al de una red más simple como es el caso del módulo VGG, lo que da mayor peso a los datos que utilizan para entrenar y probar la red.

6.2. Selección de modelos

Los resultados de entrenamiento, validación y prueba para los diferentes modelos de verificación propuestos se observan en la tabla 6.1, de estos se selección los 3 modelos con mayor exactitud de prueba y que corresponden a:

1. La arquitectura ResNet 50 siamés utilizando espectrogramas de un segundo, 32 frecuencias, descenso estocástico del gradiente como método de optimización y en cuyo entrenamiento muchos de los ejemplos del mismo locutor provienen del mismo audio pero de diferente posición, conocida de este punto en adelante como ResNet 1.

2. El módulo VGG utilizando espectrogramas de un segundo, 32 frecuencias y descenso estocástico del gradiente como método de optimización, conocido de este punto en adelante como VGG 32.

3. El módulo VGG utilizando espectrogramas de un segundo, 256 frecuencias y descenso estocástico del gradiente como método de optimización, conocido de este punto en adelante como VGG 256.

6.3. Evaluación de los modelos como verificadores

A continuación se muestran los resultados de los 3 modelos seleccionados en la sección anterior cuando se evalúan como verificadores. Esta evaluación permite determinar si los modelos independientemente del locutor pueden decir si dos audios pertenecen al mismo o a diferente locutor, de la exactitud de los verificadores depende que los sistemas de identificación, que precisamente los usan, tengan un buen desempeño. La gráfica de la figura 6.1 muestra el promedio de la precisión, exhaustividad, F1 y exactitud de verificar 10 veces 1000 pares de audios de la sección de prueba de la base de datos Librispeech.

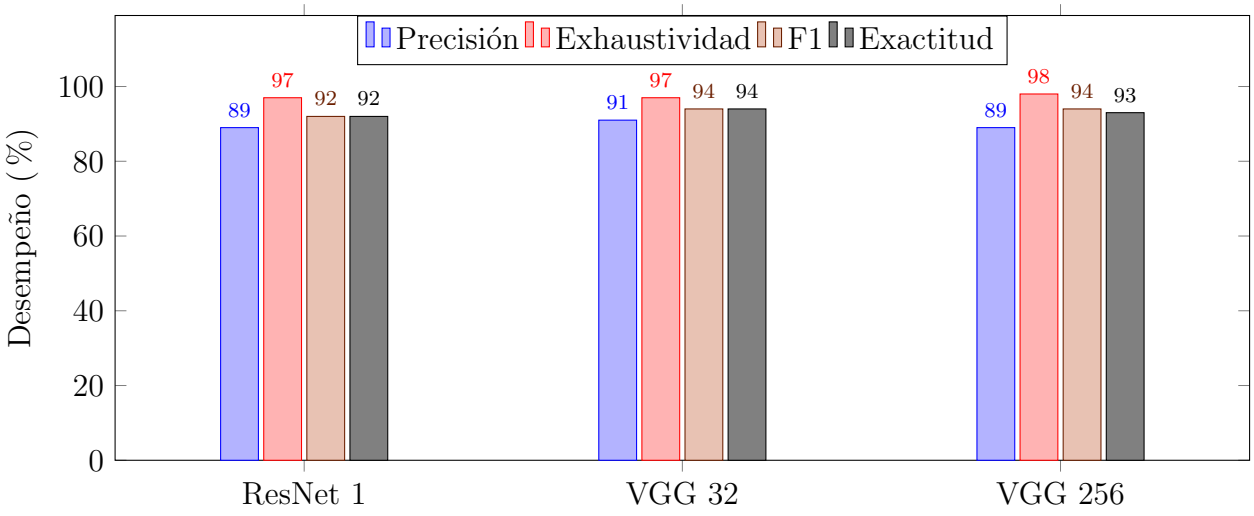

Figura 6.1: Gráfica para la evaluación de los verificadores con 1000 audios.

En la gráfica de la figura 6.1 se observa que el modelo con mayor precisión es el VGG 32, llegando a un 91 %, seguido por el VGG 256 y la ResNet 1 con un 89 % lo que indica que los modelos son muy buenos diciendo cuando dos señales son del mismo locutor, la exhaustividad de todos los modelos es superior al 97 % de manera que los modelos pueden indicar correctamente la mayor cantidad casos en que los audios son del mismo locutor, la exactitud de todos lo modelos es superior al 92 % por lo que los modelos no son sólo buenos diciendo cuando dos audios son de un mismo locutor, sino también cuando son de diferente locutor. Todos los resultados están por encima del 80 % deseado.

De los resultados anteriores se puede inferir que los modelos propuestos de verificación funcionan para comparar si dos señales de audios son del mismo locutor o no lo son con una exactitud superior al 90 %. Estos resultados sólo evalúan el desempeño del verificador y aún no utilizan una base de datos externa para llevar a cabo el proceso de identificación.

6.4. Evaluación de los modelos como identificadores

Para la evaluación de los verificadores ahora como sistemas completos de identificación se sigue el procedimiento detallado en la sección 5.10 tanto para la base de datos LibriSpeech como para SITW, trazando mapas de calor.

6.4.1. LibriSpeech

En la gráfica 6.2 se observan los mapas de calor para el primer experimento propuesto con la base de datos LibriSpeech que es la evaluación del sistema con más o menos clases, dado que los valores en algunos puntos son muy cercanos es difícil diferenciar que valor numérico tiene cada color por lo que se agrega una tabla con los valores numéricos de los datos aquí graficados en el apéndice A.1.1.

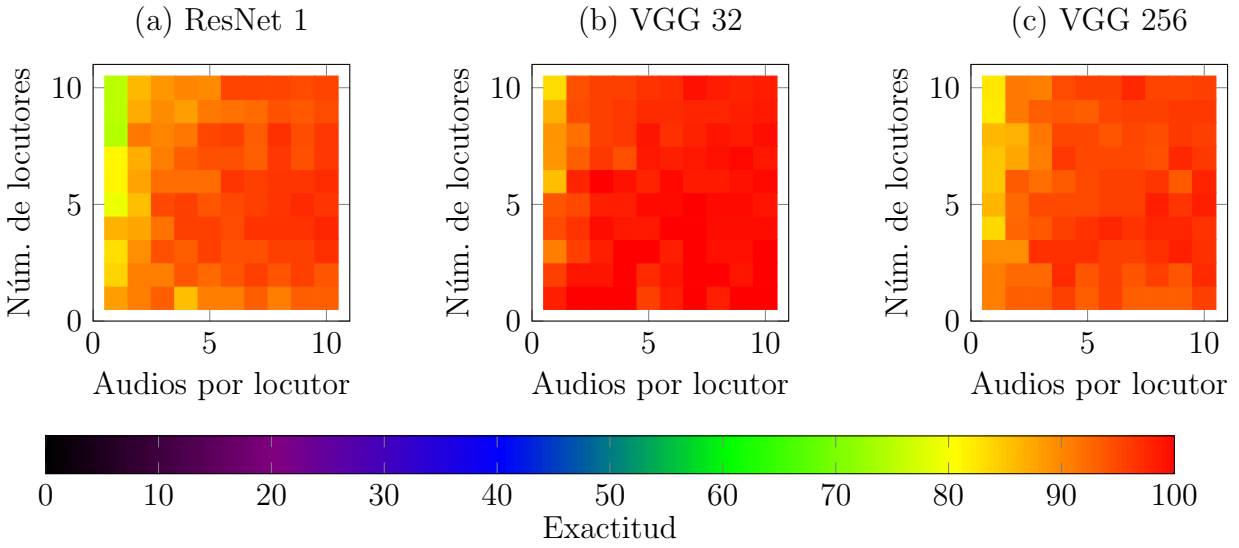

Figura 6.2: Mapas de calor para la evaluación de los sistemas como identificadores utilizando la base de datos LibriSpeech, donde Num. de locutores indica el número de locutores que se van a identificar sin tener en cuenta la clase desconocido y Audios por locutor indica el número de audios agregados en la base de datos por locutor conocido. Las zonas negras representan los valores más pequeños, las zonas azules y verdes valores intermedios, mientras que las zonas rojas los valores más altos. (a) Identificación utilizando la ResNet 1. (b) Identificación utilizando la VGG 32. (c) Identificación utilizando la VGG 256.

De los mapas de calor de la figura 6.2 se observa a primera vista que los tres sistemas están funcionando con exactitudes superiores al 70 % para identificar entre 2 y 11 clases y que llegan a valores muy cercanos a 100 % en algunas combinaciones. Una observación más detallada nos permite apreciar que el identificador con la VGG 32 es el modelo con los mejores resultados.

También podemos ver que los 3 modelos tienen un patrón similar, en donde a medida que se aumenta el número de clases, se requieren más audios conocidos a priori para tener una mejor identificación, esto quiere decir que hay una relación entre la cantidad de locutores que se desean identificar y la cantidad de audios que tengo, lo que también demuestra que este sistema es útil cuando se tienen pocos locutores, porque para identificar muchos locutores el sistema va a ser muy lento.

Para una identificación con 11 clases (10 clases que corresponden a locutores conocidos y una clase que corresponde a locutores desconocidos) y 10 audios a priori en la base de datos por locutor conocido, se tiene una exactitud del 97.7 % con la VGG 32, 95 % con la VGG 256 y 94.5 % con al ResNet 1. Para este experimento con esta base de datos no se muestra ninguna matriz de confusión ya que esta no nos da información relevante con respecto a la identificación, sin embargo si se desea conocer en que locutores se confundió cada sistema, remitirse al apéndice A.1.2 donde se muestran las matrices de confusión para cada uno.

Para el segundo experimento propuesto se observa en la gráfica 6.3 los mapas de calor para la evaluación del sistema a medida que se ingresan locutores, que busca analizar cómo funciona el sistema a medida que se conocen los locutores con los cuales va a interactuar, de manera que al final de experimento se conocen (se tiene información en la base de datos) todos los locutores que se van a identificar. Al igual que en el primer experimento diferenciar el valor numérico de cada color es difícil por lo que se agrega en el apéndice A.2.1 los valores numéricos de los datos aquí graficados.

Para este escenario el modelo va reduciendo poco a poco la cantidad de locutores desconocidos con los que debe tratar, hasta que en un momento los conoce todos. Se observa en la figura 6.3 un patrón donde a mayor número de locutores conocidos y por ende menor número de locutores desconocidos mejor desempeño tiene el sistema, más allá de la cantidad de audios en la base de datos por locutor conocido. A medida que aumenta el número de locutores conocidos el sistema funciona mejor, lo que también indica que el sistema no tiende a confundir los locutores que ya conoce, sino que su confusión radica principalmente en que identifica a locutores desconocidos como conocidos.

En el modelo con la VGG 32 con 10 locutores conocidos se llega a un desempeño del 100 % teniendo desde 7 audios por locutor y del 91 % con sólo 1 audio por locutor.

Esto quiere decir que el modelo conociendo a todos sus locutores tiene una muy buena identificación, lo que indica que el sistema de verificación adaptado a ser un identificador aprende a identificar claramente a sus locutores con una base de datos como la de LibriSpeech.

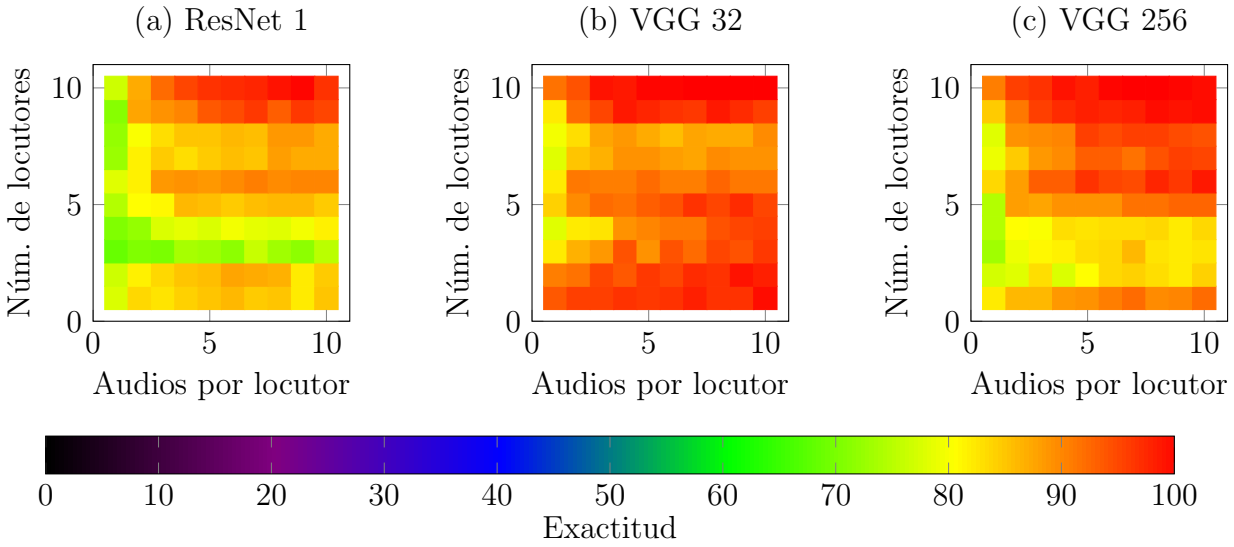

Figura 6.3: Mapas de calor para la evaluación de los sistemas como identificadores con aprendizaje de locutores utilizando la base de datos LibriSpeech, donde Num. de locutores indica el número de locutores que se van a identificar y Audios por locutor indica el número de audios agregados en la base de datos por locutor conocido. Las zonas negras representan los valores más pequeños, las zonas azules y verdes valores intermedios, mientras que las zonas rojas los valores más altos. (a) Identificación utilizando la ResNet 1. (b) Identificación utilizando la VGG 32. (c) Identificación utilizando la VGG 256.

Para este modelo no se agrega la matriz de confusión ya que no nos da ninguna información adicional. Sin embargo las matrices de confusión para la VGG 256 y para la ResNet 1 se pueden encontrar en el apéndice A.2.2.

6.4.2. SITW

En la gráfica 6.4 se observan los mapas de calor para el primer experimento propuesto para la evaluación del sistema como identificador utilizando la base de datos SITW. Dado que la transición de colores es muy suave, se pueden conocer los resultados numéricos de los datos que se grafican en la figura 6.4 en el apéndice A.1.3.

De los mapas de calor de la figura 6.4 se observa un desempeño de los modelos mucho menor al hallado en la sección 6.4.1. Este resultado se esperaba ya que SITW tiene condiciones de grabación diferentes a las de la base de datos de entrenamiento LibriSpeech con la cual fue entrenada el modelo. Sin embargo en los 3 modelos se observa que con menor cantidad de locutores los sistemas funcionaron de mejor manera y que a medida que se aumenta el número de clases es necesario tener una mayor cantidad de audios conocidos a priori, es decir que se repite el mismo patrón de la sección anterior aunque ahora con una exactitud mucho

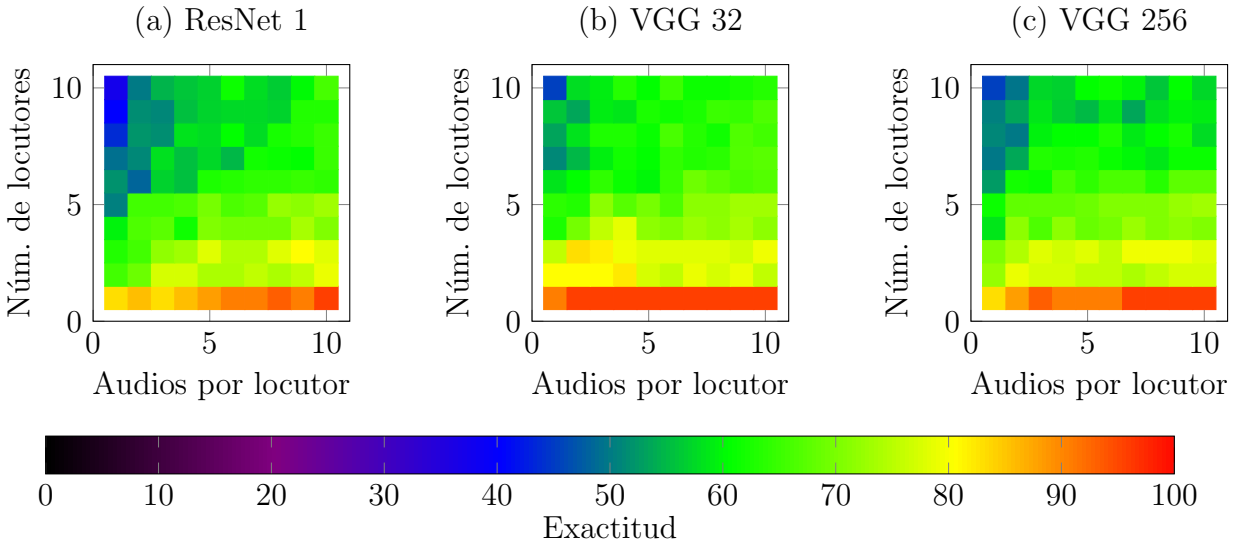

Figura 6.4: Mapas de calor para la evaluación de los sistemas como identificadores utilizando la base de datos SITW, donde Num. de locutores indica el número de locutores que se van a identificar sin tener en cuenta la clase desconocido y Audios por locutor indica el número de audios agregados en la base de datos por locutor conocido. Las zonas negras representan los valores más pequeños, las zonas azules y verdes valores intermedios, mientras que las zonas rojas los valores más altos. (a) Identificación utilizando la ResNet 1. (b) Identificación utilizando la VGG 32. (c) Identificación utilizando la VGG 256.

menor.

Para la identificación de 11 clases se llegan a un desempeño del 62.7 % con la VGG 32, del 65.5 % con la ResNet 1 y 56.8 % con la VGG 256. Este resultado no se considera bueno para lo que se esperaba en este modelo, sin embargo se parte de la premisa de que esta base de datos tiene audios mucho más difíciles de identificar, debido a sus variadas condiciones de grabación.

Para la identificación con la base de datos SITW se tiene una dificultad adicional y es que el VAD que se usa para los audios de LibriSpeech garantiza en buena medida que el contenido de audio que se está evaluando si corresponde al del locutor que se desea identificar, mientras que para la base de datos SITW un audio puede superar el VAD aún sin tener el locutor de interés, ya que el ruido de fondo podría confundirse con un locutor.

A continuación se muestra la matriz de confusión para la VGG 32 con 11 clases y 10 audios a priori por locutor conocido; las matrices de confusión para la VGG 256 y para la ResNet 1 se pueden encontrar en el apéndice A.1.4.

	Número de audios										
Loc.	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**D**
1	18	0	0	0	0	0	0	0	0	0	2
2	0	4	0	0	0	0	0	3	0	10	3
3	0	0	17	1	0	0	0	1	0	0	1
4	0	0	0	7	0	0	0	2	0	0	11
5	0	0	0	0	19	0	0	0	0	0	1
6	1	0	5	1	1	5	0	0	0	0	7
7	0	1	0	0	0	0	13	1	0	1	4
8	0	0	7	0	0	1	0	12	0	0	0
9	0	0	2	0	0	0	0	1	13	0	4
10	0	0	0	0	0	1	1	0	0	14	4
D	0	1	0	0	3	0	0	0	0	0	16

Tabla 6.2: Matriz de confusión para la identificación de 11 clases con la VGG 32 con 10 audios a priori por locutor conocido, donde Loc. indica el locutor y D indica la clase desconocido.

En la matriz de confusión de la tabla 6.2 para la VGG 32 con 11 clases y 10 audios a priori por locutor conocido, se puede observar que la fuente de errores más fuerte para el modelo es identificar como desconocido un locutor que es conocido, 37 de los 200 audios de locutores conocidos fueron identificados como desconocidos, lo que representa la mitad de los errores que cometió el modelo. Algunos locutores son confundidos entre sí, pero la mayor dificultad en este caso es que presumiblemente los audios que tienen mucho ruido son detectados como locutores desconocidos.

Para el segundo experimento propuesto se observa en la gráfica 6.5 los mapas de calor para la evaluación del sistema como identificador con aprendizaje de locutores utilizando la base de datos SITW. Al igual que en el primer experimento diferenciar el valor numérico de cada color es difícil por lo que se agregan los resultados numéricos de los datos que se grafican en el apéndice A.2.3.

Para este escenario se observa que el desempeño de los modelos sólo es bueno cuando se tiene únicamente 1 locutor conocido. Cuando se tienen 10 locutores conocidos los modelos sólo llegan a desempeños del 62 % con la VGG 32 y con la ResNet 50 y del 58.5 % con la VGG 256. Este es un resultado indeseable para lo que se busca en este proyecto. Sin embargo este tipo de de datos no tienen un ambiente similar al de los audios de entrenamiento y las redes pueden no funcionar tan bien al no verse expuestas a audios con diferentes ruidos y ambientes durante el entrenamiento.

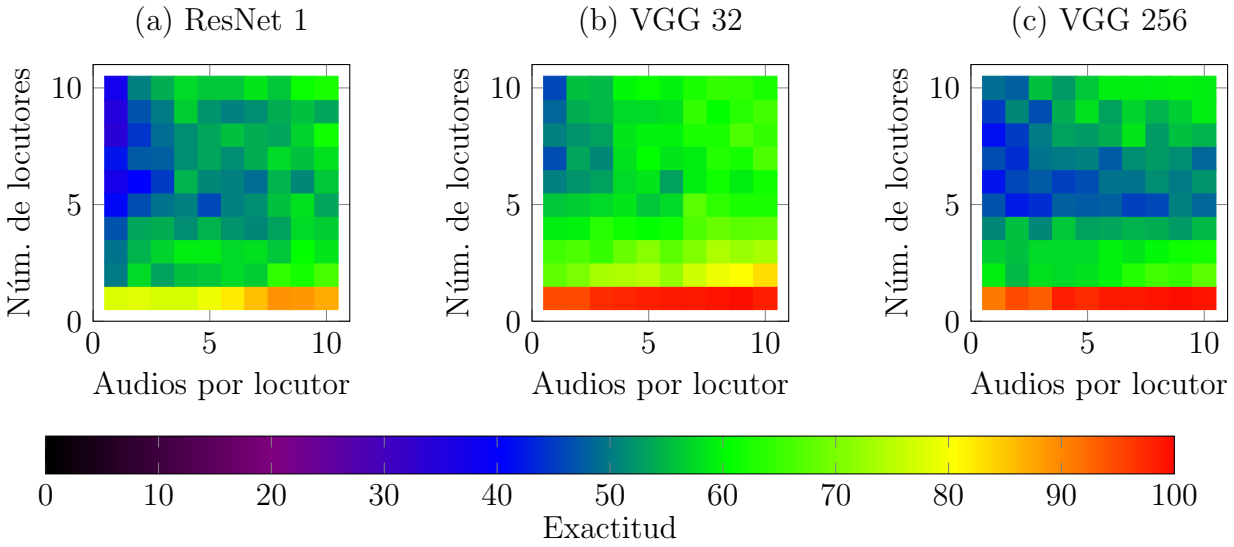

Figura 6.5: Mapas de calor para la evaluación de los sistemas como identificadores con aprendizaje de locutores utilizando la base de datos SITW, donde Num. de locutores indica el número de locutores que se van a identificar y Audios por locutor indica el número de audios agregados en la base de datos por locutor conocido. Las zonas negras representan los valores más pequeños, las zonas azules y verdes valores intermedios, mientras que las zonas rojas los valores más altos. (a) Identificación utilizando la ResNet 1. (b) Identificación utilizando la VGG 32. (c) Identificación utilizando la VGG 256.

A continuación se muestra la matriz de confusión para la VGG 32 con 10 clases y 10 audios a priori por locutor conocido; las matrices de confusión para la VGG 256 y para la ResNet 1 se pueden encontrar en el apéndice A.2.4.

	Número de audios										
Loc.	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**D**
1	18	0	0	0	0	0	0	0	1	0	1
2	0	6	0	0	0	0	1	1	0	9	3
3	0	0	17	0	0	0	0	2	0	0	1
4	0	0	0	9	0	0	0	1	0	0	10
5	0	0	0	0	16	0	0	0	0	0	4
6	2	0	5	1	1	6	0	0	0	0	5
7	0	0	2	0	0	0	13	1	0	1	3
8	0	0	4	0	0	0	2	13	0	0	1
9	0	1	0	0	0	2	0	0	13	0	4
10	0	0	0	0	0	2	0	0	0	13	5
D	0	0	0	0	0	0	0	0	0	0	0

Tabla 6.3: Matriz de confusión para la identificación de 10 clases con la VGG 32 con 10 audios a priori por locutor conocido, donde Loc. indica el locutor y D indica la clase desconocido.

De la matriz de confusión de la tabla 6.3 para la VGG 32 con 10 clases y 10 audios a priori por locutor conocido, se puede observar que la fuente de errores en este caso es identificar como desconocido a un locutor conocido, nuevamente 37 de 200 muestras fueron identificadas como desconocidos lo que supone la mitad de los errores, también nos deja ver que puede que hayan 37 audios cuyos fallos son consistentes dado que en ambos experimentos se tuvo la misma cantidad de fallos de este tipo.

6.5. Duración de los modelos de verificación

Para determinar la eficiencia de los modelos en tiempo de ejecución se realiza un experimento en el cual se mide el tiempo promedio de comparación de 1 contra 100 audios existentes en la base de datos. El promedio es hallado comparando 10 veces cada audio. Los tiempos que se miden para cada modelo son: el tiempo de creación de los espectrogramas de frecuencia, la duración de la verificación y la duración total de estos dos procesos.

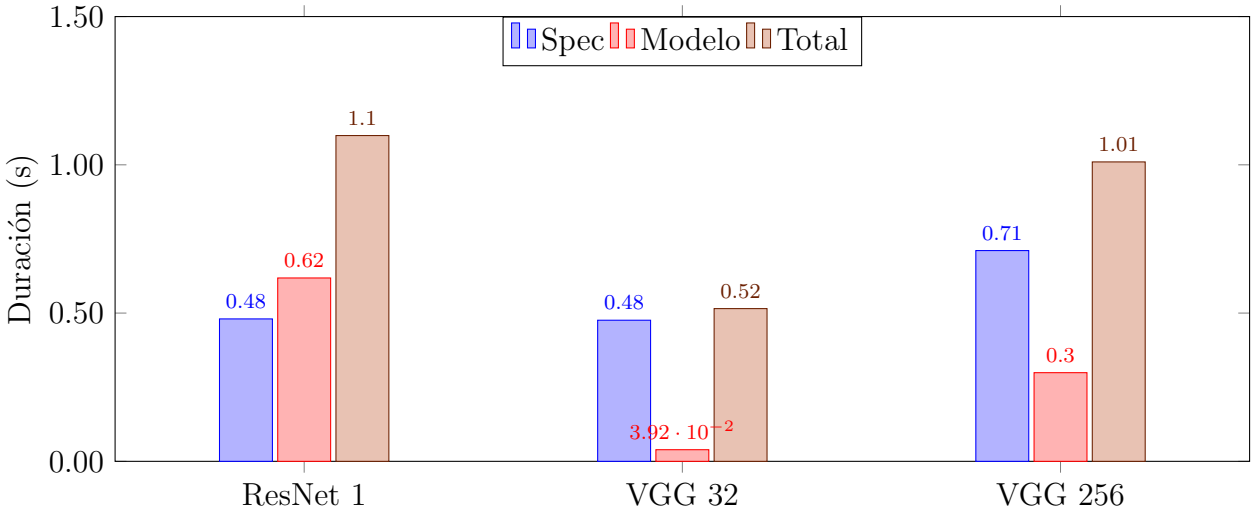

Figura 6.6: Gráfica para la duración promedio de verificación de los modelos utilizando 100 audios conocidos a priori, donde Spect muestra la duración de la creación de los espectrogramas de frecuencias, Modelo indica la duración de la verificación (el tiempo que tarda el modelo en dar una respuesta) y Total presenta el tiempo total de estos dos procesos.

En la gráfica de la figura 6.6 se observa que el modelo más rápido para la verificación es el módulo VGG utilizando 32 frecuencias, seguido por el módulo VGG utilizando 256 frecuencias y por último están el modelo ResNet.

Se observa también que la creación de los espectrogramas de frecuencia con 400 puntos

de FFT y de los cuales se toman 32 frecuencias es más rápida que la creación de los espectrogramas de frecuencia con 1024 puntos de FFT y de los cuales se toman 256 frecuencias. Este tiempo afecta directamente a la duración total del modelo. En el caso del módulo VGG con 32 frecuencias la creación de los espectrogramas toma el 92.3 % del tiempo total de verificación. En el caso del módulo VGG con 256 frecuencias la creación de los espectrogramas toma el 70 % del tiempo total de verificación. Y por último en el caso del modelo ResNet, la creación de los espectrogramas toma el 43.6 % del tiempo total de verificación. Una forma de acelerar la operación, es crear y guardar los espectrogramas de frecuencias, en lugar de la señal de audio; esto evita calcular el espectrograma una y otra vez en cada verificación.

6.6. Resumen de los resultados

De los resultados hallados en las secciones 6.3 y 6.4, se observa que el modelo más consistente es el módulo VGG utilizando 32 frecuencias. Los mejores resultados son hallados con la sección de prueba de la base de datos LibriSpeech. Con la base de datos SITW se intenta realizar la transferencia del conocimiento para la red entrenada con Librispeech, sin embargo no se obtienen buenos resultados.

Para la identificación de locutores se requiere un compromiso entre la cantidad de repeticiones de un mismo locutor, el número de locutores y la exactitud que se desea; a mayor cantidad de locutores mayor es la necesidad de audios conocidos a priori para tener un desempeño alto.

Los resultados de los modelos utilizando como clase aquella con el resultado promedio más alto tiene una mayor exactitud que con la clase del audio cuyo resultado de verificación es el más alto, en especial cuando se tienen muchos locutores desconocidos y pocos conocidos.

El módulo VGG 32 puede identificar 11 clases (10 clases que corresponden a locutores conocidos y una clase que corresponde a locutores desconocidos) con una exactitud del 97.7 % utilizando la base de datos LibriSpeech. Sin embargo, sólo llega a una exactitud del 62.7 % utilizando la base de datos SITW que contiene audios grabados en diferentes ambientes.

El módulo VGG 32 puede diferenciar con una exactitud del 100 % a 10 locutores de la base de datos LibriSpeech de los cuales tenga al menos en su base de datos, 7 audios conocidos a priori por locutor. Sin embargo, sólo lo puede hacer con una exactitud del 62 % cuando utiliza la base de datos SITW; esto debido a la naturaleza de estos audios.

El módulo VGG utilizando 32 frecuencias es 8 veces más rápido que el módulo VGG utilizando 256 frecuencias y 16 veces más rápido que los modelos ResNet.

Capítulo 7

Conclusiones y Trabajo Futuro

7.1. Conclusiones

De este trabajo se concluye que:

- Sí es posible implementar un sistema de identificación de locutor que no requiera reentrenar un nuevo modelo con nuevos locutores. Durante este trabajo esto se comprobó empíricamente utilizando 10 locutores, 3 modelos convolucionales de arquitectura siamés, cuyos resultados son superiores al 94 % en exactitud con audios de una base de datos del mismo tipo con el que fueron entrenados los modelos o hasta del 65 % con una base de datos grabada con diferentes condiciones ambientales.

- Se entrenaron con éxito 21 modelos convolucionales haciendo uso de la base de datos LibriSpeech. De estos se seleccionaron inicialmente 5 y posteriormente 3 de estos 5 para realizar modelos de identificación de locutor mediante verificación con cualquier base de datos deseada.

- Se realizó con éxito la verificación de locutores con los modelos entrenados utilizando una sección de datos de prueba de la base de datos de LibriSpeech.

- Se intenta realizar la transferencia de conocimiento de los modelos entrenados para verificar locutores de la base de datos SITW, sin embargo no se llegó al desempeño deseado.

- La creación de modelos de verificación que puedan evaluar cuando dos señales pertenecen al mismo locutor y cuando son de diferente locutor permite desarrollar sistemas de

identificación de locutores tanto para la tarea de identificación como para el aprendizaje de nuevos locutores sin necesidad de reentrenamiento.

- Los armónicos de alta frecuencia presentes en las señales de voz humana no fueron un factor determinante en la identificación de locutor para este proyecto, ya que con el uso de 32 frecuencias (1280 Hz) o 256 frecuencias (4000 Hz) se obtienen casi los mismos resultados.

- No siempre una red más grande o más profunda genera un resultado más alto para la identificación de locutor. En este trabajo el uso de una red pequeña, como es el Módulo VGG con 256 o 32 frecuencias, tiene un mejor desempeño que una ResNet 50 aún cuando esta es una red más grande y profunda, lo que da mayor peso al tipo de características con las que se entrenan los modelos de verificación.

- Es eficiente hacer uso de modelos multitarea para mejorar la inicialización de los pesos en redes convolucionales muy profunda. En este trabajo se utilizó un modelo multitarea para realizar la inicialización de la ResNet 50, disminuyendo el tiempo de entrenamiento del verificador. Si bien la inicialización de pesos es muy buena con este tipo de redes, se pudo observar que dejar los pesos libres (permitir que los pesos sigan modificando su valor después de la inicialización) en las etapas posteriores al preentrenamiento, da mejores resultados que dejar los pesos fijos (congelar los valores de los pesos de manera que no se modifiquen tras el preentrenamiento).

7.2. Trabajo futuro

Este trabajo presenta varias áreas en las cuales puede ser mejorado y continuado. A continuación se habla de algunas de ellas:

- El primer trabajo futuro para este proyecto es realizar la implementación completa en el robot Golem III del Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas (IIMAS), como un caso de estudio, utilizando una computadora portátil con GPU.

- Posteriormente se propone realizar una implementación del juego de Marco Polo con el robot Golem III, utilizando el identificador de locutor propuesto en este proyecto, el cual permitirá que el robot sepa con quién está jugando y, en caso de que hayan diversas personas, pueda determinar a quien encontró.

- Para mejorar el sistema de verificación y hacerlo robusto frente a muchos tipos de ambientes de grabación se propone entrenar las 3 redes con el mejor desempeño con una base de datos con muchos locutores en muchos tipos de ambientes como VoxCeleb [Nagrani et al., 2017].
- Se propone evaluar una red siamés con una arquitectura aún más sencilla que el módulo VGG propuesto para analizar cuál es la mínima red que se puede utilizar para la identificación automática de locutor mediante verificación.
- Se propone guardar espectrogramas de frecuencia en vez de la señal de audio completa, de manera que los modelos puedan aumentar su desempeño en tiempo.

Bibliografía

[Abadi et al., 2015] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. (2015). Tensorflow: Large-scale machine learning on heterogeneous systems. `https://www.tensorflow.org`, consultado el 07/08/2018.

[Anand, 2010] Anand, A. V. (2010). A brief study of discrete and fast fourier transorm. `http://www.math.uchicago.edu/~may/VIGRE/VIGRE2010/REUPapers/Anand.pdf`, consultado el 12 de Noviembre de 2017.

[Appalaraju and Chaoji, 2017] Appalaraju, S. and Chaoji, V. (2017). Image similarity using deep CNN and curriculum learning. *CoRR*, abs/1709.08761.

[Asano et al., 1968] Asano, S., Bloch, S., Dziewonski, A., Landisman, M., and Sato, Y. (1968). Progress report on recent improvements in the analysis of surface wave observations. 16:1–26.

[Baraldi et al., 2015] Baraldi, L., Grana, C., and Cucchiara, R. (2015). A deep siamese network for scene detection in broadcast videos. *CoRR*, abs/1510.08893.

[Bengio, 2009] Bengio, Y. (2009). Learning deep architectures for ai. *Found. Trends Mach. Learn.*, 2(1):1–127.

[Benyassine et al., 1997] Benyassine, A., Shlomot, E., Su, H. Y., Massaloux, D., Lamblin, C., and Petit, J. P. (1997). Itu-t recommendation g.729 annex b: a silence compression scheme

for use with g.729 optimized for v.70 digital simultaneous voice and data applications. *IEEE Communications Magazine*, 35(9):64–73.

[Bhattacharya et al., 2017] Bhattacharya, G., Alam, M. J., and Kenny, P. (2017). Deep speaker embeddings for short-duration speaker verification. In *INTERSPEECH*.

[Bimbot et al., 2004] Bimbot, F., Bonastre, J.-F., Fredouille, C., Gravier, G., Magrin-Chagnolleau, I., Meignier, S., Merlin, T., Ortega-García, J., Petrovska-Delacrétaz, D., and Reynolds, D. A. (2004). A tutorial on text-independent speaker verification. *EURASIP Journal on Advances in Signal Processing*, 2004(4):101962.

[Bromley et al., 1993] Bromley, J., Guyon, I., LeCun, Y., Säckinger, E., and Shah, R. (1993). Signature verification using a "siamese"time delay neural network. In *Proceedings of the 6th International Conference on Neural Information Processing Systems*, NIPS'93, pages 737–744, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

[Campbell, 1997] Campbell, J. P. (1997). Speaker recognition: A tutorial. *Proceedings of the Ieee*, 85(9):1437–1462.

[Collobert et al., 2016] Collobert, R., Puhrsch, C., and Synnaeve, G. (2016). Wav2letter: an end-to-end convnet-based speech recognition system. *CoRR*, abs/1609.03193.

[Daqrouq, 2011] Daqrouq, K. (2011). Wavelet entropy and neural network for text-independent speaker identification. *Eng. Appl. Artif. Intell.*, 24(5):796–802.

[Daqrouq and Tutunji, 2015] Daqrouq, K. and Tutunji, T. A. (2015). Speaker identification using vowels features through a combined method of formants, wavelets, and neural network classifiers. *Applied Soft Computing*, 27:231 – 239.

[Deng and Yu, 2014] Deng, L. and Yu, D. (2014). Deep learning: Methods and applications. *Foundations and Trends® in Signal Processing*, 7(3–4):197–387.

[Feng and Kai Hansen, 2005] Feng, L. and Kai Hansen, L. (2005). A new database for speaker recognition.

[Forsyth, 1995] Forsyth, M. (1995). Discriminating observation probability (dop) hmm for speaker verification. *Speech Commun.*, 17(1-2):117–129.

[Goodfellow et al., 2016] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. `http://www.deeplearningbook.org`.

[Hadsell et al., 2006] Hadsell, R., Chopra, S., and LeCun, Y. (2006). Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742.

[He et al., 2015] He, K., Zhang, X., Ren, S., and Sun, J. (2015). Deep residual learning for image recognition. *CoRR*, abs/1512.03385.

[Heideman et al., 1985] Heideman, M. T., Johnson, D. H., and Burrus, C. S. (1985). Gauss and the history of the fast Fourier transform. *Archive for History of Exact Sciences*, 34(3):265–277.

[Heigold et al., 2016] Heigold, G., Moreno, I., Bengio, S., and Shazeer, N. (2016). End-to-end text-dependent speaker verification. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5115–5119.

[Hennebert et al., 2000] Hennebert, J., Melin, H., Petrovska, D., and Genoud, D. (2000). Polycost: A telephone-speech database for speaker recognition. *Speech Communication*, 31(2):265 – 270.

[Hinton et al., 2006] Hinton, G. E., Osindero, S., and Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554.

[Hinton et al., 2012] Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. *ArXiv e-prints*.

[Ioffe and Szegedy, 2015] Ioffe, S. and Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *ArXiv e-prints*.

[John J. Godfrey, 1997] John J. Godfrey, E. H. (1997). Darpa switchboard telephone conversation corpus. `https://catalog.ldc.upenn.edu/LDC97S62`, consultado el 07/08/2018.

[John S. Garofolo, 1993] John S. Garofolo, Lori F. Lamel, W. M. F.-J. G. F. D. S. P. N. L. D. V. Z. (1993). Timit database. `https://catalog.ldc.upenn.edu/ldc93s1`, consultado el 07/08/2018.

[Joseph Campbell, 1998] Joseph Campbell, A. H. (1998). Yoho. `https://catalog.ldc.upenn.edu/products/LDC94S16`, consultado el 07/08/2018.

[Karlsson et al., 2000] Karlsson, I., Banziger, T., Dankovicová, J., Johnstone, T., Lindberg, J., Melin, H., Nolan, F., and Scherer, K. (2000). Speaker verification with elicited speaking styles in the verivox project. *Speech Communication*, 31(2):121 – 129.

[Kasabov, 1996] Kasabov, N. K. (1996). *Foundations of neural networks, fuzzy systems, and knowledge engineering*. Marcel Alencar.

[Kisi and Uncuoglu, 2005] Kisi, Ö. and Uncuoglu, E. (2005). Comparison of three backpropagation training algorithms for two case studies. volume 12, pages 434–442. CSIR.

[Koch et al., 2015] Koch, G., Zemel, R., and Salakhutdinov, R. (2015). Siamese neural networks for one-shot image recognition.

[Larcher et al., 2014] Larcher, A., Lee, K. A., Ma, B., and Li, H. (2014). Text-dependent speaker verification: Classifiers, databases and rsr2015. *Speech Communication*, 60:56 – 77.

[LeCun et al., 2015] LeCun, Y., Bengio, Y., and Hinton, G. E. (2015). Deep learning. *Nature*, 521(7553):436–444.

[Lee et al., 1993] Lee, C. H., Gauvain, J. L., Pieraccini, R., and Rabiner, L. R. (1993). Subword-based large-vocabulary speech recognition. *AT T Technical Journal*, 72(5):25–36.

[Lerch, 2012] Lerch, A. (2012). *An Introduction to Audio Content Analysis: Applications in Signal Processing and Music Informatics*. Wiley, Hoboken, N.J.

[Liu et al., 2006] Liu, M., Huang, T. S., and Zhang, Z. (2006). Robust local scoring function for text-independent speaker verification. In *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, volume 2, pages 1146–1149.

[Malekesmaeili and Ward, 2014] Malekesmaeili, M. and Ward, R. K. (2014). A local fingerprinting approach for audio copy detection. *Signal Processing*, 98:308–321.

[Matsui and Furui, 1995] Matsui, T. and Furui, S. (1995). Likelihood normalization for speaker verification using a phoneme- and speaker-independent model. *Speech Communication*, 17(1):109 – 116.

[McLaren et al., 2016] McLaren, M., Ferrer, L., Castan, D., and Lawson, A. (2016). The speakers in the wild (sitw) speaker recognition database. In *Interspeech 2016*, pages 818–822.

[Mobiny, 2018] Mobiny, A. (2018). Text-independent speaker verification using long short-term memory networks. *EESS*.

[Mohammed et al., 2017] Mohammed, M., Khan, M. B., and Bashier, E. B. M. (2017). *Machine learning : algorithms and applications.* Boca Raton : CRC Press, 2017.

[Muckenhirn et al., 2017] Muckenhirn, H., Magimai.-Doss, M., and Marcel, S. (2017). Towards directly modeling raw speech signal for speaker verification using cnns.

[Nagrani et al., 2017] Nagrani, A., Chung, J. S., and Zisserman, A. (2017). Voxceleb: a large-scale speaker identification dataset. *CoRR*, abs/1706.08612.

[Ortega-Garcia et al., 2010] Ortega-Garcia, J., Fierrez, J., Alonso-Fernandez, F., Galbally, J., Freire, M. R., Gonzalez-Rodriguez, J., Garcia-Mateo, C., Alba-Castro, J.-L., Gonzalez-Agulla, E., Otero-Muras, E., Garcia-Salicetti, S., Allano, L., Ly-Van, B., Dorizzi, B., Kittler, J., Bourlai, T., Poh, N., Deravi, F., Ng, M. W. R., Fairhurst, M., Hennebert, J., Humm, A., Tistarelli, M., Brodo, L., Richiardi, J., Drygajlo, A., Ganster, H., Sukno, F. M., Pavani, S.-K., Frangi, A., Akarun, L., and Savran, A. (2010). The multiscenario multienvironment biosecure multimodal database (bmdb). *IEEE transactions on pattern analysis and machine intelligence*, 32(6):1097–111.

[Ortega-Garcia et al., 2000] Ortega-Garcia, J., Gonzalez-Rodriguez, J., and Marrero-Aguiar, V. (2000). Ahumada: A large speech corpus in spanish for speaker characterization and identification. *Speech Communication*, 31(2):255 – 264.

[O'Shaughnessy, 2008] O'Shaughnessy, D. (2008). Invited paper: Automatic speech recognition: History, methods and challenges. *Pattern Recogn.*, 41(10):2965–2979.

[Panayotov et al., 2015] Panayotov, V., Chen, G., Povey, D., and Khudanpur, S. (2015). Librispeech: an ASR corpus based on public domain audio books. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE.

[Petrovska-Delacrétaz et al., 2000] Petrovska-Delacrétaz, D., Černocký, J., Hennebert, J., and Chollet, G. (2000). Segmental approaches for automatic speaker verification. *Digital Signal Processing*, 10(1):198 – 212.

[Prasad, 2009] Prasad, B. (2009). *Speech, Audio, Image and Biomedical Signal Processing Using Neural Networks*. Springer-Verlag, Berlin, Heidelberg.

[Ramos et al., 2018] Ramos, D., Franco-Pedroso, J., Lozano-Diez, A., and Gonzalez-Rodriguez, J. (2018). Deconstructing cross-entropy for probabilistic binary classifiers. *Entropy*, 20(3).

[Ramos-Castro et al., 2007] Ramos-Castro, D., Fierrez-Aguilar, J., Gonzalez-Rodriguez, J., and Ortega-Garcia, J. (2007). Speaker verification using speaker- and test-dependent fast score normalization. *Pattern Recognition Letters*, 28(1):90 – 98.

[Reynolds, 1995] Reynolds, D. A. (1995). Speaker identification and verification using gaussian mixture speaker models. *Speech Commun.*, 17(1-2):91–108.

[Salehghaffari, 2018] Salehghaffari, H. (2018). Speaker verification using convolutional neural networks. *EESS*.

[Simonyan and Zisserman, 2014] Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

[Snyder et al., 2015] Snyder, D., Chen, G., and Povey, D. (2015). Musan: A music, speech, and noise corpus. *CoRR*, abs/1510.08484.

[Snyder et al., 2017] Snyder, D., Garcia-Romero, D., Povey, D., and Khudanpur, S. (2017). Deep neural network embeddings for text-independent speaker verification. pages 999–1003.

[Snyder et al., 2016] Snyder, D., Ghahremani, P., Povey, D., Garcia-Romero, D., Carmiel, Y., and Khudanpur, S. (2016). Deep neural network-based speaker embeddings for end-to-end speaker verification. In *2016 IEEE Spoken Language Technology Workshop (SLT)*, pages 165–170.

[Thévenaz and Hügli, 1995] Thévenaz, P. and Hügli, H. (1995). Usefulness of the lpc-residue in text-independent speaker verification. *Speech Communication*, 17(1):145 – 157.

[Variani et al., 2014] Variani, E., Lei, X., McDermott, E., Moreno, I. L., and Gonzalez-Dominguez, J. (2014). Deep neural networks for small footprint text-dependent speaker verification. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4052–4056.

[Voxforge.org,] Voxforge.org. Free speech... recognition (linux, windows and mac) - voxforge.org. `http://www.voxforge.org/`. accessed 07/08/2018.

[Western Electric CO., 1969] Western Electric CO., I. (1969). *Fundamentals of Telephone Communication Systems*.

[William M. Fisher, 1993] William M. Fisher, George R. Doddington, K. M. G.-M. C. J. A. K. S. B. J. S. (1993). Ntimit database. `https://catalog.ldc.upenn.edu/LDC93S2`, consultado el 07/08/2018.

[Wong et al., 2011] Wong, Y. W., Ch'ng, S. I., Seng, K. P., Ang, L.-M., Chin, S. W., Chew, W. J., and Lim, K. H. (2011). A new multi-purpose audio-visual unmc-vier database with multiple variabilities. *Pattern Recognition Letters*, 32(13):1503 – 1510.

[Woo et al., 2006] Woo, R. H., Park, A., and Hazen, T. J. (2006). The mit mobile device speaker verification corpus: Data collection and preliminary experiments. In *IEEE Odyssey – The Speaker and Language Recognition Workshop*.

[Xiang et al., 2002] Xiang, B., Chaudhari, U. V., Navrátil, J., Ramaswamy, G. N., and Gopinath, R. A. (2002). Short-time gaussianization for robust speaker verification. In *2002 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages I–681–I–684.

[Yu and Deng, 2015] Yu, D. and Deng, L. (2015). *Automatic Speech Recognition: A Deep Learning Approach*. Signals and Communication Technology. Springer, London.

[Zagoruyko and Komodakis, 2015] Zagoruyko, S. and Komodakis, N. (2015). Learning to compare image patches via convolutional neural networks. *CoRR*, abs/1504.03641.

Apéndice A

Resultados adicionales

A.1. Resultados para evaluación de los verificadores como identificadores

A.1.1. Identificador con LibriSpeech

A continuación se muestran las tablas de resultados para los mapas de calor de la sección 6.4.1.

	Número de audios									
Loc.	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
10	74.5	85.5	88.2	89.5	89.1	94.5	94.5	94.5	94.1	94.5
9	74.5	86.5	89.0	87.5	90.5	91.0	91.5	93.5	93.0	94.0
8	73.9	90.6	89.4	90.6	94.4	95.0	92.8	96.1	93.9	95.6
7	80.6	86.3	90.0	92.5	93.8	93.8	92.5	95.6	93.8	95.6
6	80.7	87.1	91.4	91.4	91.4	95.7	95.0	95.7	95.7	96.4
5	78.3	85.0	94.2	95.0	93.3	94.2	95.0	95.8	96.7	95.8
4	86.0	87.0	91.0	95.0	95.0	94.0	96.0	96.0	96.0	97.0
3	82.5	88.7	93.8	92.5	95.0	93.8	93.8	95.0	95.0	96.3
2	83.3	90.0	90.0	93.3	91.7	93.3	95.0	93.3	95.0	93.3
1	87.5	90.0	92.5	85.0	90.0	90.0	92.5	90.0	92.5	92.5

Tabla A.1: Resultados numéricos para identificación utilizando la ResNet 1 y la base de datos LibriSpeech, donde Loc. indica el número de locutores conocidos.

	Número de audios									
Loc.	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
10	81.8	90.5	90.0	94.1	95.0	95.0	96.8	94.5	94.5	95.0
9	81.5	90.5	92.5	93.0	92.5	94.5	95.0	95.0	95.5	95.5
8	85.6	86.1	90.6	94.4	94.4	93.3	94.4	93.9	95.6	95.0
7	84.4	86.9	90.0	95.6	94.4	94.4	94.4	93.8	96.9	95.6
6	84.3	92.9	91.4	92.9	94.3	95.0	95.0	95.7	92.9	97.1
5	85.8	91.7	94.2	94.2	94.2	95.0	95.0	97.5	95.8	97.5
4	83.0	92.0	93.0	95.0	96.0	97.0	96.0	97.0	97.0	96.0
3	88.7	88.7	96.3	96.3	96.3	95.0	95.0	96.3	97.5	96.3
2	90.0	91.7	91.7	96.7	93.3	95.0	93.3	95.0	93.3	96.7
1	90.0	92.5	92.5	95.0	92.5	95.0	92.5	92.5	92.5	95.0

Tabla A.2: Resultados numéricos para identificación utilizando la VGG 256 y la base de datos LibriSpeech, donde Loc. indica el número de locutores conocidos.

	Número de audios									
Loc.	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
10	82.7	93.6	95.0	95.0	95.9	96.4	98.6	97.7	97.3	97.7
9	86.0	94.0	95.0	95.5	96.5	96.5	97.0	97.0	97.5	98.0
8	88.3	91.1	95.0	95.6	98.3	96.1	97.2	98.3	97.8	98.9
7	88.1	92.5	95.6	93.8	98.1	97.5	98.1	98.8	99.4	98.1
6	85.0	97.1	100.0	98.6	97.1	99.3	97.9	97.9	98.6	99.3
5	93.3	94.2	97.5	97.5	99.2	99.2	100.0	99.2	99.2	98.3
4	94.0	96.0	99.0	98.0	98.0	100.0	100.0	99.0	99.0	99.0
3	90.0	95.0	97.5	100.0	100.0	97.5	100.0	98.8	98.8	100.0
2	95.0	98.3	98.3	100.0	96.7	98.3	100.0	100.0	98.3	100.0
1	97.5	100.0	100.0	100.0	95.0	97.5	100.0	97.5	100.0	100.0

Tabla A.3: Resultados numéricos para identificación utilizando la VGG 32 y la base de datos LibriSpeech, donde Loc. indica el número de locutores conocidos.

A.1.2. Matrices de confusión para el identificador con Librispeech

	Número de audios										
Loc.	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**D**
1	20	0	0	0	0	0	0	0	0	0	0
2	0	20	0	0	0	0	0	0	0	0	0
3	0	0	19	0	0	0	0	0	1	0	0
4	0	0	0	20	0	0	0	0	0	0	0
5	0	0	0	0	20	0	0	0	0	0	0
6	0	0	0	0	0	20	0	0	0	0	0
7	0	0	0	0	0	0	20	0	0	0	0
8	0	0	0	0	0	0	0	20	0	0	0
9	0	0	0	0	0	2	0	0	18	0	0
10	0	0	0	0	0	0	0	0	0	20	0
D	3	0	0	0	1	0	1	1	1	2	11

Tabla A.4: Matriz de confusión para la identificación de 11 clases con la ResNet 1 con 10 audios a priori por locutor conocido, donde Loc. indica el locutor y D indica la clase desconocido.

	Número de audios										
Loc.	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**D**
1	20	0	0	0	0	0	0	0	0	0	0
2	0	20	0	0	0	0	0	0	0	0	0
3	0	0	20	0	0	0	0	0	0	0	0
4	0	0	0	20	0	0	0	0	0	0	0
5	0	0	0	0	20	0	0	0	0	0	0
6	0	0	0	0	0	20	0	0	0	0	0
7	0	0	0	0	0	0	20	0	0	0	0
8	0	0	0	0	0	0	0	20	0	0	0
9	0	0	0	0	0	1	0	0	19	0	0
10	0	0	0	0	0	0	0	0	0	20	0
D	0	0	0	0	0	1	1	1	0	1	16

Tabla A.5: Matriz de confusión para la identificación de 11 clases con la VGG 32 con 10 audios a priori por locutor conocido, donde Loc. indica el locutor y D indica la clase desconocido.

	Número de audios										
Loc.	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**D**
1	20	0	0	0	0	0	0	0	0	0	0
2	0	20	0	0	0	0	0	0	0	0	0
3	0	0	19	0	0	1	0	0	0	0	0
4	0	0	0	20	0	0	0	0	0	0	0
5	0	0	0	0	20	0	0	0	0	0	0
6	0	0	0	0	0	20	0	0	0	0	0
7	0	0	0	0	0	0	20	0	0	0	0
8	0	0	0	0	0	0	0	20	0	0	0
9	0	0	0	0	0	0	0	0	20	0	0
10	0	0	0	0	0	0	0	0	0	20	0
D	1	1	0	0	2	0	2	1	2	1	10

Tabla A.6: Matriz de confusión para la identificación de 11 clases con la VGG 256 con 10 audios a priori por locutor conocido, donde Loc. indica el locutor y D indica la clase desconocido.

A.1.3. Identificador con SITW

A continuación se muestran las tablas de resultados para los mapas de calor de la sección 6.4.2.

	Número de audios									
Loc.	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
10	36.8	49.5	54.1	55.5	56.4	60.9	57.3	55.9	59.5	65.5
9	40.0	51.0	50.5	55.0	56.5	57.0	57.0	56.5	62.5	62.5
8	43.3	51.7	51.1	57.8	57.2	60.0	57.2	61.7	62.8	64.4
7	48.7	50.6	57.5	55.0	56.9	54.4	61.3	60.6	60.0	64.4
6	51.4	47.9	56.4	55.0	62.1	62.9	63.6	63.6	64.3	63.6
5	50.0	65.0	65.0	65.8	69.2	65.8	66.7	71.7	70.8	72.5
4	59.0	66.0	67.0	61.0	70.0	69.0	69.0	69.0	73.0	70.0
3	62.5	65.0	71.3	72.5	77.5	73.8	73.8	77.5	80.0	77.5
2	65.0	68.3	76.7	76.7	73.3	73.3	75.0	73.3	75.0	78.3
1	82.5	85.0	82.5	85.0	87.5	90.0	90.0	92.5	90.0	95.0

Tabla A.7: Resultados numéricos para identificación utilizando la ResNet 1 y la base de datos SITW, donde Loc. indica el número de locutores conocidos.

	Número de audios									
Loc.	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
10	45.0	49.1	56.8	56.4	60.0	60.9	59.1	55.5	60.0	56.8
9	50.0	53.0	58.0	56.0	54.5	58.0	53.0	57.5	59.0	58.5
8	50.6	49.4	58.9	60.0	60.0	62.2	58.3	60.0	61.7	57.2
7	49.4	53.1	62.5	61.9	62.5	60.6	59.4	58.1	60.6	60.6
6	52.1	61.4	60.7	65.7	63.6	65.7	65.7	62.9	67.1	67.1
5	60.8	67.5	67.5	67.5	68.3	70.0	70.0	70.0	71.7	72.5
4	58.0	71.0	66.0	71.0	68.0	69.0	72.0	70.0	71.0	70.0
3	70.0	73.8	77.5	76.2	77.5	75.0	78.7	78.7	78.7	77.5
2	71.7	78.3	76.7	76.7	75.0	75.0	73.3	75.0	76.7	76.7
1	82.5	87.5	92.5	90.0	90.0	90.0	95.0	95.0	95.0	95.0

Tabla A.8: Resultados numéricos para identificación utilizando la VGG 256 y la base de datos SITW, donde Loc. indica el número de locutores conocidos.

	Número de audios									
Loc.	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
10	45.0	57.3	58.6	62.7	60.0	64.5	60.9	60.9	61.8	62.7
9	55.5	53.0	58.5	58.0	62.0	61.0	62.0	64.5	64.0	66.0
8	52.8	57.8	62.2	62.2	60.6	65.6	62.8	61.1	66.1	65.0
7	50.6	54.4	58.8	61.3	60.0	63.7	62.5	63.1	66.2	66.9
6	57.9	60.0	63.6	60.7	59.3	64.3	68.6	67.1	65.7	66.4
5	63.3	62.5	70.8	66.7	69.2	65.8	70.8	70.8	72.5	72.5
4	62.0	69.0	75.0	78.0	71.0	70.0	72.0	71.0	73.0	71.0
3	75.0	82.5	81.3	78.7	77.5	77.5	77.5	77.5	76.2	78.7
2	80.0	80.0	80.0	81.7	76.7	75.0	78.3	76.7	78.3	75.0
1	90.0	95.0	95.0	95.0	95.0	95.0	95.0	95.0	95.0	95.0

Tabla A.9: Resultados numéricos para identificación utilizando la VGG 32 y la base de datos SITW, donde Loc. indica el número de locutores conocidos.

A.1.4. Matrices de confusión para el identificador con SITW

	Número de audios										
Loc.	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**D**
1	18	0	0	0	0	0	0	0	0	0	2
2	0	8	0	0	0	0	1	4	0	7	0
3	1	0	16	1	0	1	0	0	0	0	1
4	1	0	2	10	0	2	0	0	3	0	2
5	0	0	0	0	18	0	0	0	2	0	0
6	5	1	5	2	0	3	0	0	0	0	4
7	0	3	0	0	0	0	17	0	0	0	0
8	0	0	1	3	0	1	2	12	0	0	1
9	1	0	1	1	0	0	0	0	13	0	4
10	0	0	0	0	0	1	2	0	0	14	3
D	1	1	1	0	1	0	1	0	0	0	15

Tabla A.10: Matriz de confusión para la identificación de 11 clases con la ResNet 1 con 10 audios a priori por locutor conocido, donde Loc. indica el locutor y D indica la clase desconocido.

	Número de audios										
Loc.	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**D**
1	18	0	0	0	0	0	0	0	2	0	0
2	0	7	0	0	0	0	0	0	0	12	1
3	0	0	18	1	0	0	0	0	1	0	0
4	0	0	0	9	1	0	0	1	2	0	7
5	0	0	0	2	9	0	0	0	9	0	0
6	2	1	4	2	1	4	3	1	0	1	1
7	0	10	0	0	0	0	8	0	0	1	1
8	0	1	7	0	0	0	0	12	0	0	0
9	0	0	4	1	0	1	0	0	13	0	1
10	0	1	0	0	0	0	2	0	0	14	3
D	0	1	1	0	2	1	1	0	0	1	13

Tabla A.11: Matriz de confusión para la identificación de 11 clases con la VGG 256 con 10 audios a priori por locutor conocido, donde Loc. indica el locutor y D indica la clase desconocido.

A.2. Resultados para evaluación de los verificadores como identificadores con ingreso de nuevos locutores

A.2.1. Identificador con ingreso y cantidad limitada de locutores para LibriSpeech

A continuación se muestran las tablas de resultados para los mapas de calor de la sección ??.

	Número de audios									
Loc.	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
10	76.0	86.5	91.5	94.5	96.0	96.5	97.0	98.5	99.5	96.0
9	70.5	87.0	88.5	89.5	93.0	94.0	95.5	92.5	95.5	94.5
8	71.5	79.5	82.5	84.5	84.5	85.5	85.0	88.0	88.0	86.5
7	72.0	81.0	84.0	82.5	84.0	85.0	84.5	87.5	86.5	86.5
6	77.5	81.0	88.0	88.5	88.0	89.0	90.0	89.0	89.5	89.5
5	74.0	80.0	80.5	85.5	85.0	84.0	85.0	85.5	85.5	84.0
4	70.0	71.5	77.0	78.0	77.0	79.0	78.0	77.5	79.0	81.0
3	68.0	70.0	69.5	73.0	72.5	71.0	75.5	72.5	71.0	74.5
2	76.0	81.0	83.0	84.5	85.0	87.0	86.5	86.0	81.5	84.0
1	77.0	83.0	82.0	83.5	84.5	85.5	84.0	84.5	81.5	84.5

Tabla A.12: Resultados numéricos para identificación con ingreso y una cantidad limitada de locutores utilizando la ResNet 1 y la base de datos LibriSpeech, donde Loc. indica el número de locutores conocidos.

	Número de audios									
Loc.	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
10	90.0	95.0	96.0	98.5	97.5	99.5	100.0	100.0	99.5	99.0
9	84.0	90.5	95.0	96.5	97.5	97.0	97.5	99.0	98.5	99.0
8	77.5	88.5	89.0	89.5	95.0	94.0	95.0	95.0	94.0	93.5
7	78.5	84.0	88.0	89.0	92.5	92.5	91.0	93.5	95.0	94.5
6	83.0	87.5	92.5	92.5	96.0	94.5	94.0	97.0	95.5	98.0
5	74.0	87.5	87.0	88.5	88.5	88.5	91.0	91.0	92.0	92.0
4	74.0	80.0	79.5	82.5	82.0	82.5	82.5	83.0	81.5	83.0
3	72.5	78.5	80.5	81.0	82.5	83.0	85.5	82.0	81.5	82.0
2	76.5	77.5	82.5	77.0	80.0	83.0	83.5	82.5	81.5	83.5
1	81.5	85.5	85.5	88.0	88.5	90.0	91.5	89.0	89.5	91.5

Tabla A.13: Resultados numéricos para identificación con ingreso y una cantidad limitada de locutores utilizando la VGG 256 y la base de datos LibriSpeech, donde Loc. indica el número de locutores conocidos.

	Número de audios									
Loc.	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
10	91.0	93.5	98.5	98.0	99.5	99.5	100.0	100.0	100.0	100.0
9	81.5	91.5	94.5	98.0	97.0	96.0	95.5	98.0	96.5	95.0
8	79.0	82.5	87.0	88.0	86.5	85.0	87.0	86.5	86.5	89.0
7	77.5	84.5	86.0	88.5	88.5	87.5	87.0	89.0	88.5	88.5
6	81.5	90.5	90.0	90.0	91.5	90.0	90.0	92.0	90.5	90.5
5	83.5	89.0	91.5	91.0	92.5	93.5	96.0	94.5	96.5	94.5
4	77.5	81.5	82.0	88.5	89.5	90.5	90.5	93.5	94.5	95.0
3	81.5	86.0	88.0	93.5	88.5	93.5	91.5	94.5	93.5	95.5
2	90.0	91.0	94.5	93.0	95.5	94.5	96.5	96.0	98.5	97.5
1	93.0	95.0	95.0	96.0	95.5	96.0	97.5	96.0	96.0	99.0

Tabla A.14: Resultados numéricos para identificación con ingreso y una cantidad limitada de locutores utilizando la VGG 32 y la base de datos LibriSpeech, donde Loc. indica el número de locutores conocidos.

A.2.2. Matrices de confusión con ingreso y cantidad limitada de locutores para LibriSpeech

	Número de audios										
Loc.	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**D**
1	20	0	0	0	0	0	0	0	0	0	0
2	0	20	0	0	0	0	0	0	0	0	0
3	0	0	19	0	0	1	0	0	0	0	0
4	0	0	0	20	0	0	0	0	0	0	0
5	0	1	0	0	19	0	0	0	0	0	0
6	0	0	0	0	0	20	0	0	0	0	0
7	0	0	0	0	0	0	20	0	0	0	0
8	0	0	0	0	0	0	0	20	0	0	0
9	0	0	0	0	0	0	0	0	20	0	0
10	0	0	0	0	0	0	0	0	0	20	0
D	0	0	0	0	0	0	0	0	0	0	0

Tabla A.15: Matriz de confusión para la identificación de 10 clases con la ResNet 1 con 10 audios a priori por locutor conocido, donde Loc. indica el locutor y D indica la clase desconocido.

	Número de audios										
Loc.	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**D**
1	20	0	0	0	0	0	0	0	0	0	0
2	0	19	0	0	1	0	0	0	0	0	0
3	0	0	20	0	0	0	0	0	0	0	0
4	0	0	0	20	0	0	0	0	0	0	0
5	0	0	0	0	20	0	0	0	0	0	0
6	0	0	0	0	0	20	0	0	0	0	0
7	0	0	0	0	0	0	18	0	0	0	2
8	0	0	0	0	0	0	0	20	0	0	0
9	0	0	0	0	0	5	0	0	15	0	0
10	0	0	0	0	0	0	0	0	0	20	0
D	0	0	0	0	0	0	0	0	0	0	0

Tabla A.16: Matriz de confusión para la identificación de 10 clases con la VGG 256 con 10 audios a priori por locutor conocido, donde Loc. indica el locutor y D indica la clase desconocido.

A.2.3. Identificador con ingreso y cantidad limitada de locutores para SITW

A continuación se muestran las tablas de resultados para los mapas de calor de la sección ??.

	Número de audios									
Loc.	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
10	37.0	50.0	53.5	57.0	55.5	55.5	58.0	55.5	60.5	62.0
9	34.5	46.5	49.5	56.0	51.5	50.0	51.0	53.5	55.0	53.0
8	33.5	44.0	48.5	51.0	53.0	55.0	53.5	53.0	56.5	61.0
7	41.5	47.0	47.5	51.0	53.0	51.0	53.5	57.0	55.0	57.5
6	36.5	40.0	44.5	54.0	50.5	50.0	48.5	54.0	50.5	55.5
5	40.5	46.0	49.0	50.0	45.5	50.0	51.0	55.0	57.5	53.0
4	46.5	53.0	53.5	51.0	54.0	51.5	50.5	52.5	57.0	56.0
3	49.0	54.0	56.5	58.5	58.5	57.0	57.5	55.5	60.0	58.0
2	49.5	57.0	52.5	55.0	55.5	57.0	56.0	63.5	61.5	65.5
1	77.0	77.5	76.5	76.5	78.5	81.5	85.0	88.5	88.0	86.5

Tabla A.17: Resultados numéricos para identificación con ingreso y una cantidad limitada de locutores utilizando la ResNet 1 y la base de datos SITW, donde Loc. indica el número de locutores conocidos.

	Número de audios									
Loc.	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
10	48.5	47.5	55.0	52.0	55.5	58.5	58.5	58.5	59.0	58.5
9	44.5	50.5	46.0	53.5	57.5	52.5	56.5	54.0	56.0	58.5
8	41.0	44.5	49.5	52.5	52.0	53.5	58.0	52.0	55.0	54.0
7	46.0	43.5	49.0	49.5	50.0	47.0	51.0	52.0	51.0	48.0
6	41.5	45.5	47.0	45.0	46.0	49.0	49.0	51.0	49.5	51.5
5	46.5	42.0	43.5	47.5	47.0	47.5	45.0	45.5	50.0	48.0
4	50.5	55.0	50.5	55.5	52.5	53.0	54.0	53.5	52.0	54.5
3	56.0	55.0	57.0	57.0	57.0	59.0	58.0	59.5	61.0	61.5
2	58.5	54.5	57.5	57.0	57.5	59.0	62.5	65.0	64.0	67.0
1	90.5	94.0	92.5	97.5	96.5	98.0	98.0	98.5	99.0	98.5

Tabla A.18: Resultados numéricos para identificación con ingreso y una cantidad limitada de locutores utilizando la VGG 256 y la base de datos SITW, donde Loc. indica el número de locutores conocidos.

	Número de audios									
Loc.	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
10	45.5	55.0	54.5	59.5	60.5	59.0	62.0	64.0	63.5	62.0
9	48.0	53.5	54.5	57.0	57.0	57.5	64.0	60.0	63.0	65.0
8	50.0	51.5	52.5	58.0	59.0	59.0	62.0	62.0	66.0	64.0
7	46.0	53.0	50.5	57.5	60.0	58.0	58.5	62.5	61.5	66.0
6	50.0	51.5	52.5	57.0	58.0	52.5	58.5	61.0	58.5	61.0
5	55.5	56.0	57.0	58.0	55.5	57.0	67.0	63.5	62.0	62.0
4	58.5	59.0	63.0	61.0	60.0	61.0	66.0	68.0	67.0	67.5
3	65.0	66.0	64.0	68.0	70.5	67.0	70.0	73.5	72.0	73.0
2	67.5	69.5	73.0	73.5	74.5	71.5	76.0	78.5	80.0	82.5
1	94.0	94.0	96.5	97.0	97.5	98.0	98.0	98.5	99.0	97.5

Tabla A.19: Resultados numéricos para identificación con ingreso y una cantidad limitada de locutores utilizando la VGG 32 y la base de datos SITW, donde Loc. indica el número de locutores conocidos.

A.2.4. Matrices de confusión con ingreso y cantidad limitada de locutores para SITW

	Número de audios										
Loc.	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**D**
1	18	0	0	0	0	0	0	0	0	0	2
2	0	8	0	0	0	0	1	6	0	3	2
3	0	0	17	2	0	0	0	0	0	0	1
4	0	1	4	9	0	1	0	0	2	0	3
5	0	0	0	0	16	0	0	0	3	0	1
6	4	1	6	0	0	4	0	0	0	1	4
7	0	2	0	0	0	3	14	1	0	0	0
8	0	0	1	3	0	0	2	12	0	0	2
9	1	0	2	1	0	1	0	0	13	0	2
10	0	1	0	0	0	1	2	0	0	13	3
D	0	0	0	0	0	0	0	0	0	0	0

Tabla A.20: Matriz de confusión para la identificación de 10 clases con la ResNet 1 con 10 audios a priori por locutor conocido, donde Loc. indica el locutor y D indica la clase desconocido.

	Número de audios										
Loc.	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**D**
1	18	0	0	0	0	0	0	0	1	0	1
2	0	5	0	1	0	0	1	0	0	13	0
3	0	0	16	1	0	0	0	1	1	1	0
4	0	0	0	10	1	0	0	0	1	0	8
5	0	0	0	1	12	0	0	0	7	0	0
6	0	0	6	1	1	8	0	0	0	0	4
7	0	10	0	0	0	0	8	0	0	1	1
8	0	0	8	0	0	0	0	12	0	0	0
9	0	0	2	1	0	3	0	0	13	0	1
10	0	0	0	0	0	0	2	0	0	15	3
D	0	0	0	0	0	0	0	0	0	0	0

Tabla A.21: Matriz de confusión para la identificación de 10 clases con la VGG 256 con 10 audios a priori por locutor conocido, donde Loc. indica el locutor y D indica la clase desconocido.