Desarrollo de un algoritmo en Python para la detección de ataques de tipo reproducción de audios mediante parámetros de magnitud y de fase de la señal de audio

Andrés Mauricio Santamaria Salas, Dionisio Andrés de Jesús Blanco Bello

Trabajo de Grado para optar al título de Ingeniero Electrónico

Director

Alexander Sepulveda Sepulveda

Doctor en Ingeniería

Universidad Industrial de Santander

Facultad de Ingenierías Fisicomecánicas

Escuela de Ingenierías Eléctrica, Electrónica y de Telecomunicaciones

Bucaramanga

2020

Dedicatoria

Este trabajo viene dedicado para todas aquellas personas que apoyaron el desarrollo y ejecución de este trabajo de grado.

En especial reconozco la permanente presencia de Dios en mi camino de vida.

Agradecimientos

Agradezco a mi familia por el apoyo ecónomico y moral que tuvieron para conmigo durante el desarrollo de mi carrera. También agradezco a mis amigos y compañeros por las vivencias de estos inolvidables años de universidad.

Un reconocimiento y agradecimiento importante lo realizo a mi director de trabajo de grado, por dedicar su tiempo, experiencia y conocimiento en la guía de mi proyecto.

Tabla de Contenido

Lista de Figuras

Lista de Tablas

Resumen

Título: Desarrollo de un algoritmo en Python para la detección de ataques de tipo reproducción de audios mediante parámetros de magnitud y de fase de la señal de audio [*]

Autor: Andrés Mauricio Santamaria Salas, Dionisio Andrés de Jesús Blanco Bello [**]

Palabras Clave: ASVspoof, ataque, portabilidad, Python, reproducción.

Descripción: Se busca implementar algoritmos para la detección de ataques del tipo reproducción en señales de audio, el desempeño de estos es comparado mediante el puntaje obtenido en términos de EER(%). Además, se implementa un método de fusión, como alternativa para los algoritmos individuales los cuales van a ser comparados bajo el mismo criterio. La muestra de datos que se utilizó en este trabajo fue la base ASVspoof 2017 (Versión 2). Todo el código está bajo el lenguaje de Python y se inició con un pre procesamiento, donde la señal de entrada son los archivos de audio que ofrece la base de datos; durante este proceso es aplicada la ecuación de cada modelo, para así obtener como salida el vector de coeficientes de características para cada señal. A partir de esto, bajo un esquema de entrenamiento se generan los modelos de mezclas gaussianas para los conjuntos de señales genuinas y reproducidas, así al final se obtiene un puntaje de calificación. Los modelos de fusión propuestos se presentan como una alternativa experimental de contraste con los ya valorados en estudios anteriores. Al final se presenta como una buena alternativa la fusión de los modelos de MFCC e IMFCC como una opción para la detección de ataques de tipo de reproducción con un puntaje de 29.27% de EER, y se descartan bloques como el levantamiento sinusoidal y normalización para este tipo de sistemas en donde la detección del ruido es importante.

---

[*] Trabajo de grado

[**] Facultad de Ingenierías Físico-Mecánicas. Escuela de Ingenierías Eléctrica, Electrónica y Telecomunicaciones. Director: Alexander Sepulveda Sepulveda Doctor en Ingenieria.

Abstract

Title: Development of an algorithm in python for the detection of attacks of the type of audio reproduction by means of magnitude and phase parameters of the audio signal [*]

Author: Andres Mauricio Santamaria Salas, Dionisio Andres de Jesus Blanco Bello [**]

Keywords: ASVspoof, attacks, portability, python, replay.

Description: The aim is to implement algorithms for the detection of attacks of the reproduction type in audio signals, their performance is compared by means of the score obtained in terms of EER (%). Furthermore, a fusion method is implemented as an alternative for the individual algorithms which will be compared under the same criteria. The data sample used in this work was the ASVspoof 2017 (Version 2) database. All the code is under the python language and began with a pre-processing, where the input signal is the audio files offered by the database; During this process the equation of each model is applied, in order to obtain the vector of characteristic coefficients for each signal as an output. From this, under a training scheme, the Gaussian mixture models are generated for the sets of genuine and reproduced signals, thus at the end a qualification score is obtained. The proposed fusion models are presented as an experimental alternative to contrast with those already evaluated in previous studies. In the end, the fusion of the MFCC and IMFCC models is presented as a good alternative as an option for the detection of replay-type attacks with a score of 29.27 % EER, and blocks such as sinusoidal lifting and normalization are discarded for this type of systems where noise detection is important.

---

[*] Trabajo de grado

[**] Facultad de Ingenierías Físico-Mecánicas. Escuela de Ingenierías Eléctrica, Electrónica y Telecomunicaciones. Director: Alexander Sepulveda Sepulveda Doctor en Ingenieria.

Introducción

En la actualidad, los sistemas de reconocimiento de hablantes han obtenido gran popularidad debido a la masificación y uso de dispositivos en el mercado que incluyen la señal de voz. Un ejemplo particular de esto corresponde al auge y masificación de los dispositivos celulares, los cuales pueden incorporar algoritmos de verificación automática del hablante (Ergünay et al. (2015)). Además, en el campo forense la ASV (verificación automática del hablante) por sus siglas en inglés, es también muy importante debido a la numerosa cantidad de pruebas que se presentan mediante grabaciones de voz en muchos casos o juicios.

Es indudable que la fiabilidad de la tecnología ASV ha avanzado considerablemente en los últimos tiempos y se está desplegando actualmente en una variedad cada vez mayor de aplicaciones prácticas, como en los centros de llamadas, en los sistemas de diálogo hablado y en muchos productos de consumo masivo (Wu et al. (2017)). Es de gran importancia, que estos sistemas de verificación basados en biometría sean robustos ante cualquier tipo de ataque, ya que existen casos de naturaleza ilegal en el que las personas en particular buscan vulnerar este tipo de sistemas biométricos. En el mercado se puede encontrar una amplia variedad de plataformas o bases de datos, creadas para promover el desarrollo de contramedidas destinadas a proteger la verificación automática del hablante de los ataques de falsificación.

Una de ellas, es ASVspoof (Wu et al. (2015)) la cual consiste en una serie de desafíos con una plataforma de trabajo realmente interesante y muy completa, además de ofrecer ciertos recursos como su base de datos para el desarrollo y la proactividad de ser usada en muchos estudios

de investigación alrededor de esta área de trabajo. El primer desafío, ASVspoof 2015 se concentró en los ataques de síntesis y conversión de voz, el segundo desafío, ASVspoof 2017 se centró por lo tanto en el desarrollo de contramedidas para el ataque de reproducción de voz (Kinnunen et al. (2017)).

Existe un método fácil de implementar, con el cual pueden vulnerar nuestros sistemas ASV y acceder a nuestra información, este es el de la grabación de voz de la víctima mediante algún dispositivo electrónico con micrófono. Debido a este tipo de situaciones, se han llevado a cabo esfuerzos por desarrollar alternativas con las que se pueda proteger estos datos contra ataques de suplantación (Chen et al. (2017); Font et al. (2017); Hanilçi (2017)); el enfoque de este documento se centra en un solo tipo, el de reproducción de voz. Este, consiste básicamente en obtener la grabación de la voz de una persona mediante el micrófono de algún dispositivo electrónico, en este caso un teléfono celular o un computador, los cuales son dispositivos cotidianos y de fácil acceso, para hacerlo pasar por la voz genuina de la víctima en sistemas biométricos de esta índole y así poder vulnerar sus datos personales. Uno de los escenarios donde toma gran importancia el determinar si una voz es genuina o reproducida, ocurre con pruebas de audio presentadas por los profesionales del derecho involucrados en diligencias judiciales. Por tanto, en este estudio se busca implementar los modelos más utilizados para el desarrollo de algoritmos de contramedida y de manera experimental mediante una serie de pruebas mostrar los resultados obtenidos por estos y las variaciones propuestas (fusión de modelos) (Muckenhirn et al. (2017); Chakroborty and Saha (2009); Patil et al. (2017); Witkowski et al. (2017)).

1. Objetivos

1.1. Objetivo general

Desarrollar un algoritmo en Python, para la detección de ataques de tipo reproducción de audios mediante parámetros de magnitud (MFCC, IMFCC Y LFCC) y fase (Mel-RP) de la señal de audio.

1.2. Objetivos específicos

Evaluar los coeficientes inversos en la escala de Mel (IMFCC) y bancos de filtros con espaciado lineal (LFCC) como parámetros de magnitud, para la detección de ataques de tipo reproducción basado en modelos de mezclas Gaussianas.

Evaluar el método de fase relativa (Mel-RP) como parámetro de fase para la detección de ataques de tipo reproducción en modelos de mezclas Gaussianas.

Medir el desempeño en términos de EER al fusionar características IMFCC, LFCC y Mel-RP, respecto a los coeficientes MFCC mediante la fusión lineal de modelos.

2. Marco teórico

2.1. Base de datos

ASVspoof 2017 se origina en el corpus RedDots que fue recogido por voluntarios de todo el mundo, en su mayoría investigadores de verificación automática del hablante (ASV) utilizando teléfonos inteligentes. La base de datos utilizada para el desarrollo de este trabajo es la ASVspoof 2017 versión 2, la cual se enfoca principalmente en los ataques de tipo reproducción. Es importante tener en cuenta que las declaraciones genuinas, son un subconjunto de las grabaciones originales de RedDots, mientras que las grabaciones de tipo reproducción son versiones reproducidas y re-capturadas. Las grabaciones de tipo reproducción corresponden a un escenario de "voz robada", en el que el atacante tiene acceso a una copia digital de una grabación original de un orador, que se reproduce a través de transductores de calidad variable. Teniendo en cuenta los múltiples elementos señalados, se hace necesario hacer claridad sobre los componentes del corpus del ASVspoof 2017 Versión 2, el cual está dividido en tres subconjuntos: entrenamiento, desarrollo y evaluación (Kinnunen et al. (2017)). A continuación, se explica cada uno de ellos:

Subconjunto de entrenamiento: El subconjunto de entrenamiento consiste en una cantidad de archivos de audio utilizados en machine learning para establecer el punto de operación. Este consiste en 1.507 discursos auténticos y 1.507 señales reproducidas de 10 locutores masculinos. Las muestras reproducidas se originaron a partir de tres condiciones reproducidas en seis sesiones diferentes. El subconjunto de entrenamiento fue recogido en un solo lugar (Phapatanaburi et al. (2019)).

Subconjunto de desarrollo: Incluye grabaciones de voz genuina y de señales reproducidas de un total de 8 locutores. El número total de discurso auténtico es de 760, y el número total de señales reproducidas es de 950. Las muestras reproducidas se originaron en 10 sesiones diferentes con 10 condiciones de reproducción. El subconjunto de desarrollo se grabó en tres lugares diferentes, uno de ellos, es idéntico al empleado para las grabaciones realizadas en el subconjunto de entrenamiento (Phapatanaburi et al. (2019))

Subconjunto de evaluación: El subconjunto de evaluación consiste en archivos de audios utilizados para evaluar el rendimiento del modelo entrenado. Este consta de 1.298 señales de voces auténticas y 12.008 repetidas de un total de 24 locutores. Las muestras de habla reproducidas se originaron a partir de 110 condiciones de habla reproducidas en 161 sesiones diferentes. El subconjunto de evaluación se desarrolló con diferentes condiciones a la de los subconjuntos de entrenamiento y desarrollo (Phapatanaburi et al. (2019))

Tabla 1
Número de archivos total de la base de datos ASVspoof 2017 (Versión 2)

Subconjunto	Genuino	Reproducido
Entrenamiento	1507	1507
Desarrollo	760	950
Evaluacion	1298	12008
Total	3565	14465

Nota: Número de audios contenidos en cada una de las subcarpetas (Entrenamiento, desa-

rrollo y evaluación) dentro de la base de datos ASVspoof 2017 Versión 2 (Delgado et al. (2018))

2.2. Preprocesamiento de la señal

Este acápite, expone los bloques que permiten adecuar la señal, con el fin de procesarla. Seguidamente, se explica cada uno.

2.2.1. Filtro pre énfasis. Al aplicar este filtro se ganan varias características: equilibrar el espectro de frecuencia, evitar problemas numéricos al realizar la operación de la transformada de Fourier y por último puede mejorar la relación señal-ruido (SNR). Este filtro es de primer orden y se rige mediante la siguiente ecuación:

$$y(t) = x(t) - \alpha x(t-1) \tag{1}$$

El coeficiente del filtro (alpha) puede variar entre 0.95 y 0.97, el usado en este trabajo es de 0.97.

2.2.2. Ventaneo. Aplicamos una función de ventana; para este trabajo se usó la ventana de Hamming a cada cuadro. Existen diferentes razones por las cuales se debe aplicar este bloque, estas son, contrarrestar la suposición hecha por la FFT de que los datos son infinitos y otro factor es para reducir la fuga espectral. La ecuación que la rige tiene la siguiente forma:

$$w[n] = 0.54 - 0.46cos(\frac{2\pi n}{N-1}) \qquad 0 \leq n \leq N-1 \tag{2}$$

Dónde N es la longitud de la ventana

2.2.3. Transformada de Fourier. Se calcula el espectro de frecuencia para cada una de las ventanas, a esto se le halla el espectro en potencia con la siguiente fórmula:

$$P = \frac{|FFT(X_i)|^2}{N} \tag{3}$$

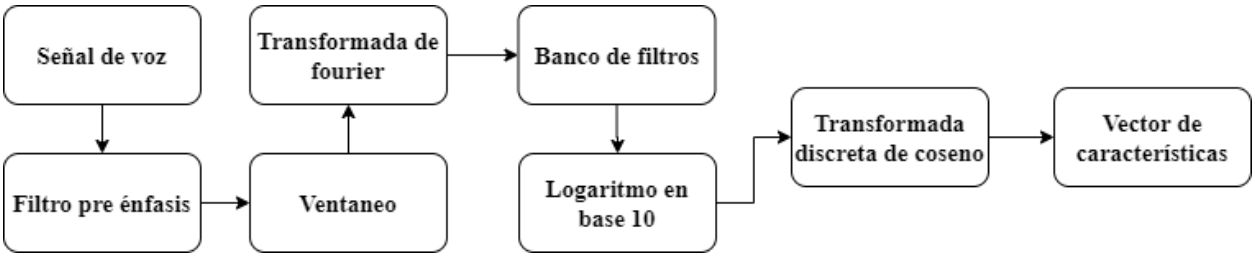
Donde, *Xi* es el iésimo cuadro de señal x y N es el numero de muestras de la transformada.

2.2.4. Banco de filtros. Se usan filtros triangulares para extraer bandas de frecuencia, el espaciado entre cada uno de estos filtros está determinado por la escala en cada una de las funciones. Cada filtro tiene una respuesta de 1 en la frecuencia central y disminuye linealmente llegando a 0, hasta que alcance las frecuencias centrales de los dos filtros adyacentes, donde la respuesta es 0. Este es uno de los bloques fundamentales en el modelado de las funciones para la extracción de características, en él radica la principal diferencia entre las funciones de espectro en magnitud, ya que cada función tiene una fórmula diferente para el espaciado en frecuencia de estos filtros. La función general para filtros triangulares es la siguiente:

$$f(x) = \begin{cases} Hm(k) = 0 & si \quad k < f(m-1) \\ Hm(k) = \frac{[k-f(m-1)]}{[f(m)-f(m-1)]} & si \quad f(m-1) \leq K < f(m) \\ Hm(k) = 1 & si \quad k = f(m) \\ Hm(k) = \frac{[f(m+1)-k]}{[f(m+1)-f(m)]} & si \quad x \geq 5 \\ Hm(k) = 0 & si \quad k > f(m+1) \end{cases} \tag{4}$$

2.2.5. Transformada discreta de coseno. Este bloque se utiliza para descorrelacionar los coeficientes del banco de filtros y producir una representación comprimida. Nota: En

Figura 1. Diagrama de bloques del preprocesamiento de la señal



este diagrama, se muestra cada uno de los pasos para obtener el vector de características de una señal de voz; aplica para los modelos de MFCC, IMFCC y LFCC, en donde la ecuación característica de cada uno fue puesta en el bloque de banco de filtros.

2.2.6. Levantamiento sinusoidal. Este bloque es utilizado en sistemas de verificación del hablante para eliminar el énfasis a los coeficientes más altos; ademas de filtrar en el dominio cepstral.

2.2.7. Normalización. Existen distintos métodos para normalizar datos; para este trabajo se escogió la normalización media, es decir restamos la media a cada coeficiente de todos los cuadros. Este bloque es de utilidad para equilibrar el espectro y mejorar la SNR.

2.3. Modelos

En esta sección se exponen algunos de los modelos más utilizados en el campo de la detección de ataques de tipo reproducción, en sistemas automáticos de verificación del hablante y el proceso de selección de estos se basó en los siguientes criterios:

El rendimiento obtenido por estos modelos en estudios revisados en la planeación y formación de los objetivos de este trabajo (Patil and Kamble (2018)).

El amplio campo de variación o pequeñas modificaciones que se pueden realizar para alterar su rendimiento.

Su fácil implementación en el lenguaje escogido (Python), además de su simetría al escribir su código, pues la estructura al ser la misma solo cambia la ecuación característica que determina la distribución de los filtros.

El modelo CQCC se escogió como referencia al ser utilizado en un amplio número de estudios (Todisco et al. (2017)), además de ser el modelo base otorgado por ASVspoof en su página oficial.

El modelo Mel-RP referencia se escogió bajo el criterio de ser un complemento para los modelos en magnitud escogidos al inicio del estudio, puesto que las características en fase, son discriminadas totalmente por los anteriores Li et al. (2018).

Por lo anterior, los modelos de fusión propuestos, se escogieron bajo distintos criterios: el primero de ellos (MFCC e IMFCC) surge como una alternativa para comprobar si una escala y su inversa pueden ofrecer de manera conjunta mejores resultados, el segundo (MFCC y LFCC) y tercero (IMFCC y Mel-RP) surgen para responder a la pregunta; ¿Es mejor obtener todas las características de una señal tanto en fase como en magnitud? Chakroborty and Saha (2009).

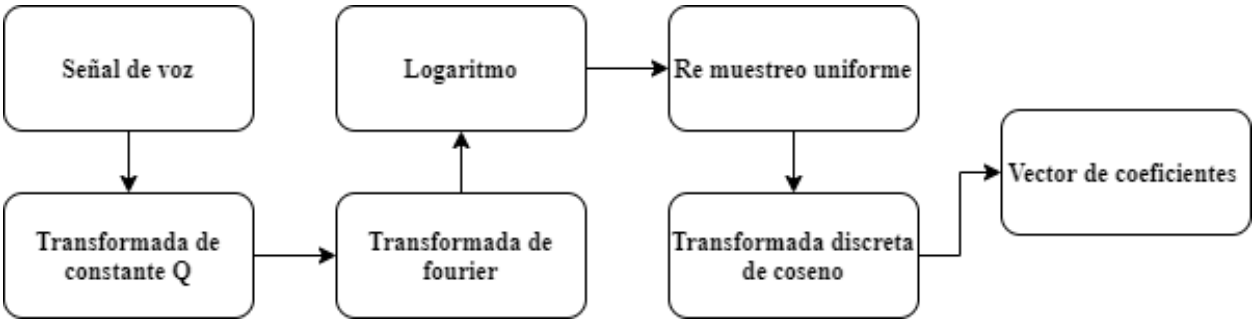
2.3.1. CQCC (Constant Q Cepstral Coefficients) . Para la extracción de características CQCC es necesaria la transformada de la constante Q o Constant Q Transform por sus siglas en inglés (CQT). La CQT ofrece una mayor resolución en frecuencias más bajas y una mayor

resolución de tiempo en frecuencias más altas. El espectro de una señal suele obtenerse mediante la transformada discreta de Fourier o Discrete Fourier Transform (DFT), mientras que el cepstrum se suele implementar con la transformada discreta del coseno (DCT) (Todisco et al. (2017)). Las características CQCC se extraen de acuerdo con:

$$CQCC_{(p)} = \sum_{l=0}^{L-1} \log |\bar{X}^{CQ}(l)|^2 \cos(\frac{p(l-\frac{1}{2})\pi}{L}) \qquad p = 0...L-1 \qquad (5)$$

Donde L es el índice de escala lineal.

Figura 2. Diagrama de bloques para el modelo CQCC



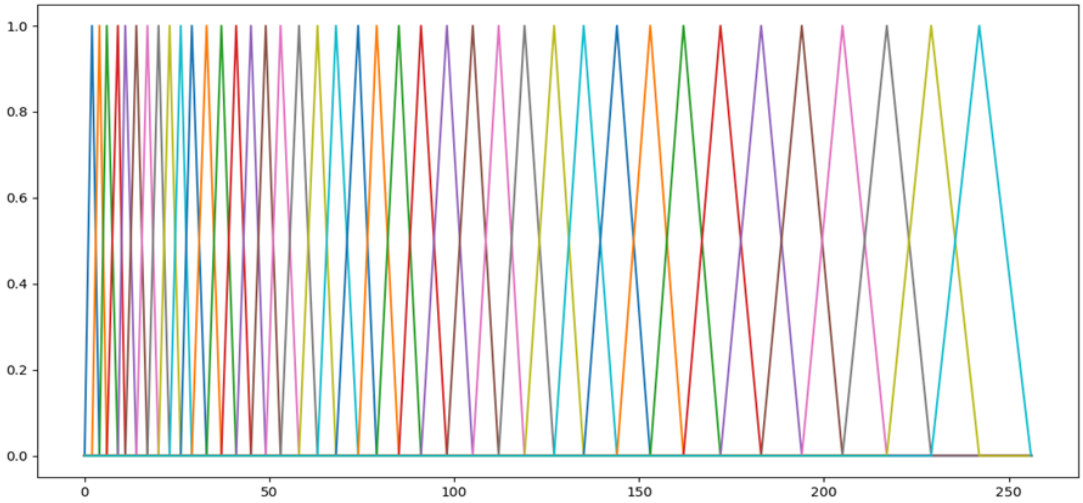
Nota: En este, se muestra cada uno de los pasos para obtener el vector de características para el modelo de coeficientes cepstrales de constante Q.

2.3.2. MFCC (Mel-Frequency Cepstral Coefficients). Son los coeficientes para la representación de la voz basados en la percepción auditiva del ser humano. Estos surgen de la necesidad de extraer características de las componentes de una señal de audio que sean adecuadas para la identificación de su contenido, así como remover todas aquellas que posean información innecesaria, como lo son; el ruido de fondo, emociones, volumen, tono, entre otras, debido a que

estas no aportan nada al proceso de reconocimiento, al contrario, lo hacen menos robusto (Chakroborty and Saha (2009)). Las características MFCC se extraen de acuerdo con:

$$f_{mel} = 2595 \log_{10}(1 + \frac{f}{700}) \tag{6}$$

Figura 3. Banco de filtros triangulares en escala frecuencia de Mel



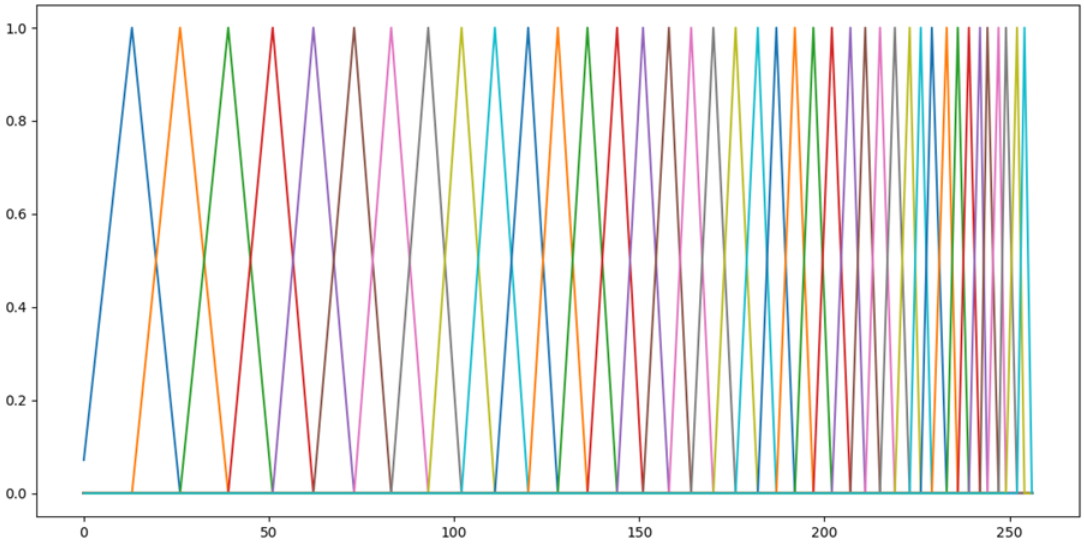
Nota: La gráfica muestra la forma y distribución de los filtros para el modelo MFCC según su ecuación característica, el eje vertical corresponde al valor de amplitud y el eje horizontal al número de componentes de mezclas gaussianas (1 y 256 respectivamente) para este caso.

2.3.3. IMFCC (Inverse Mel-Frequency Cepstral Coefficients). Los coeficientes cepstrales en la escala de Mel invertida, son un modelo propuesto en algunos estudios (Chakroborty and Saha (2009)). Al igual que el modelo MFCC consta de un banco de filtros triangulares distribuidos por su ecuación en frecuencia característica, en el cual se observa un sistema auditivo

que crece de manera opuesta al sistema auditivo humano. La tarea de esta función es recopilar la información que se pierde con el MFCC. Las características IMFCC se extraen de acuerdo con:

$$\hat{f}_{mel}(f) = 2195.2860 - 2595\log_{10}(1 + \frac{4031.25 - f}{700}) \tag{7}$$

Figura 4. Banco de filtros triangulares en escala frecuencia inversa de Mel



Nota: La gráfica indica la forma y distribución de los filtros para el modelo IMFCC según su ecuación característica, el eje vertical corresponde al valor de amplitud y el eje horizontal al número de componentes de mezclas gaussianas (1 y 256 respectivamente) para este caso.

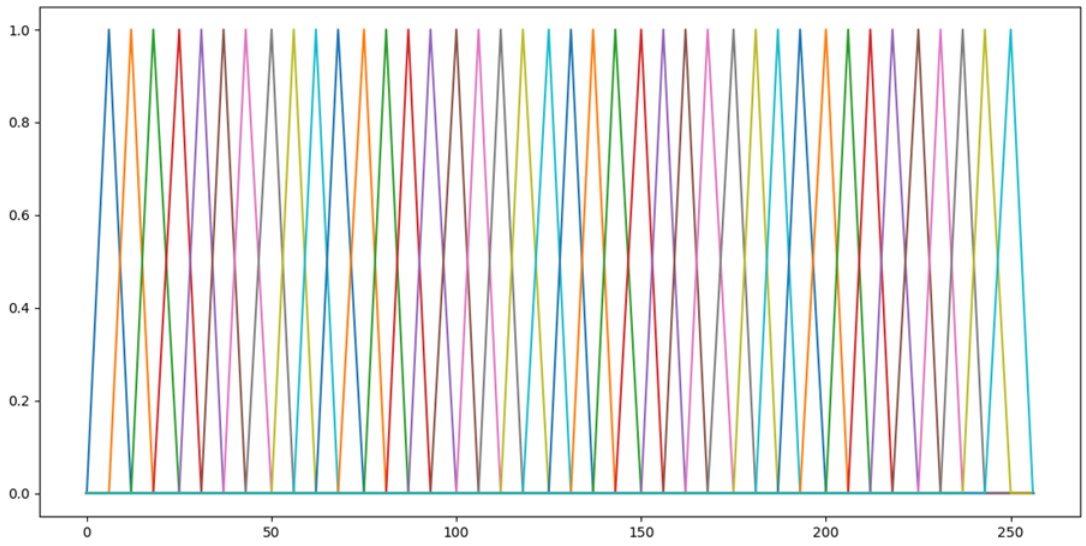
2.3.4. LFCC (Linear-Frequency Cepstral Coefficients). Los coeficientes cepstrales de frecuencia lineal son un modelo que utiliza un banco de filtros triangulares con una separación uniforme entre ellos, con los que se obtiene una mejor resolución de la señal a altas frecuencias, porque la separación entre filtros no aumenta o disminuye como si lo hacen los de

MFCCs e IMFCCs (Zhou et al. (2011)). Las características LFCC se extraen de acuerdo con:

$$f_{lineal} = \frac{f}{N} \tag{8}$$

Donde N es el numero de filtros.

Figura 5. Banco de filtros triangulares en escala lineal de frecuencia



Nota: La gráfica presenta la forma y distribución de los filtros para el modelo LFCC según su ecuación característica, el eje vertical asume el valor de amplitud y el eje horizontal el número de componentes de mezclas gaussianas (1 y 256 respectivamente) para este caso.

2.3.5. Mel-RP (Mel-Frequency Relative Phase). Para obtener la información en magnitud y fase se realiza la operación de la transformada discreta de fourier para una señal de voz $X_{(w,t)}$:

$$Magnitud = |X_{(w,t)}| \qquad Fase = e^{j\theta_{(w,t)}} \tag{9}$$

Sin embargo, la fase extraída presenta un problema y es que la posición del cuadro del discurso de entrada en la misma frecuencia provoca cambios en la información de fase (Nakagawa et al. (2011)). Una alternativa para resolver este inconveniente es mantener constante la fase en una frecuencia base, y por consiguiente la fase de las otras frecuencias se calcula a partir de esta: (Phapatanaburi et al. (2019)).

Un ejemplo presentado en algunos estudios (Nakagawa et al. (2011)) para entender mejor esta situación, consiste en establecer la fase de la frecuencia base en 0, con lo que se obtendría lo siguiente:

$$|X_{(w,t)}|e^{j\theta_{(w,t)}} * e^{-j\theta_{(w,t)}} \tag{10}$$

Es así como se puede normalizar aún más la información de fase y obtener la información que necesitamos:

$$\theta_{(w',t)} + \frac{w'}{w}(-\theta_{(w,t)}) \tag{11}$$

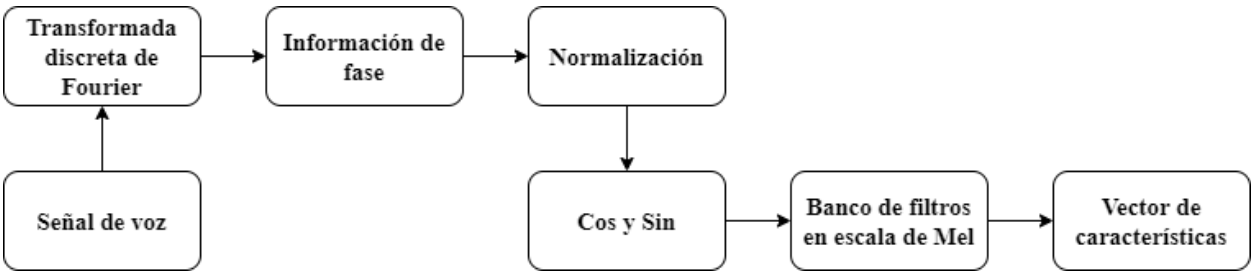
No obstante, como se observa en (Nakagawa et al. (2011)) esta ecuación en la que se comparan dos valores de fase presenta un problema. La solución a esto consiste en pasar esta información a coordenadas de círculo unitario:

$$[\cos(\theta), \sin(\theta)] \tag{12}$$

Finalmente, para completar el modelo, la información es procesada a través de un banco de filtros en escala de Mel como se observó en la figura 3.

Nota: En la figura se visualiza cada uno de los pasos para obtener el vector de características

Figura 6. Diagrama de bloques para el modelo Mel-RP



para el modelo de Mel-RP.

2.4. Archivo PKL

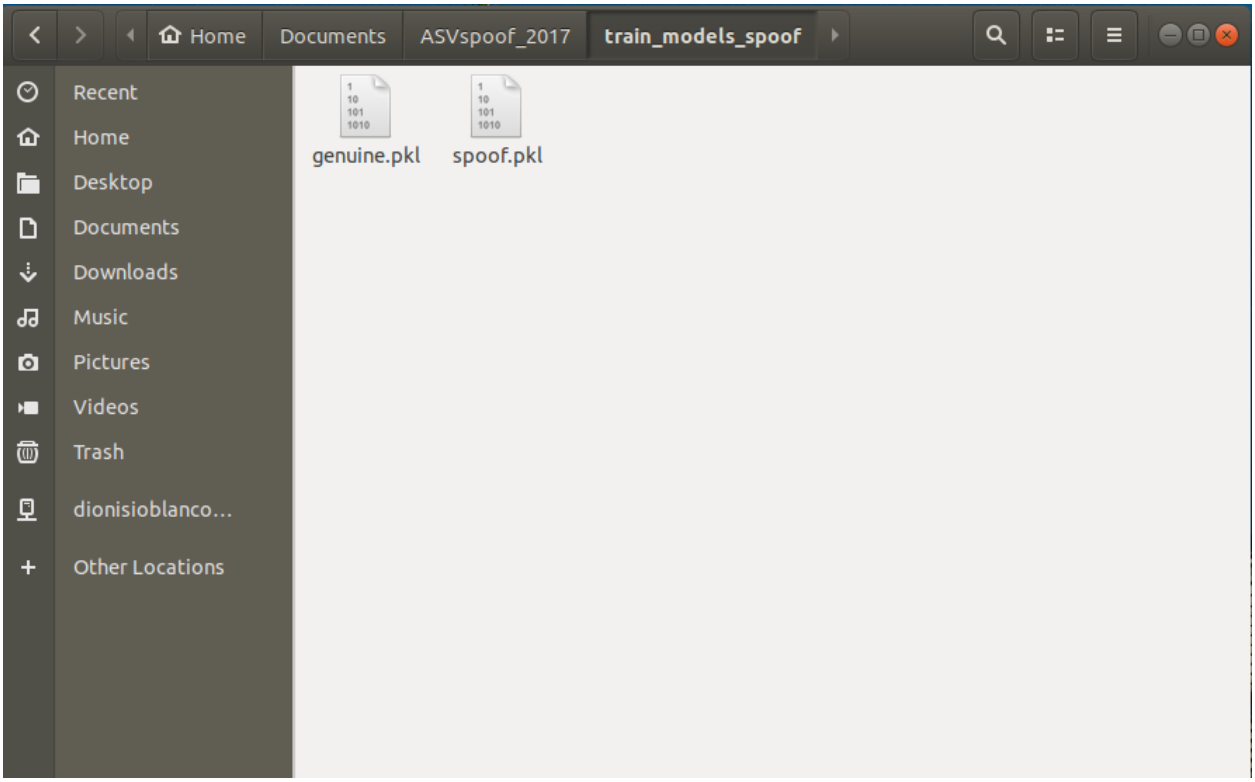
Es un archivo creado por pickle, un módulo de Python que permite que los objetos puedan ser serializados como archivos en el disco y des serializados de nuevo en el código, cuando este se ejecuta. Principalmente este archivo contiene un flujo de bytes que representa el objeto en cuestión. Una de las principales ventajas de este tipo de archivos es el espacio ahorrado al almacenar gran cantidad de información y su fácil acceso a ésta, cuando se carga de nuevo en el programa. Esta característica le da una gran viabilidad en portabilidad, ya que los procesos de machine learning suelen tener un alto consumo computacional y de tiempo.

Nota: La carpeta de los modelos (objetos) guardados como cadenas de bytes con la extensión .pkl están separados para voces genuinas y voces repetidas en este caso.

2.5. Clasificador

Aunque existe una gran variedad de clasificadores como; i-vector, redes neuronales profundas (DNN) (Yang et al. (2018)), redes neuronales convolucionales (CNN) y redes residuales (ResNet) (Lai et al. (2019)), que se utilizan para la detección de ataques de repetición, la im-

Figura 7. Vista dentro de la carpeta de modelos entrenados



plementación del modelo de mezcla gaussiana (GMM) es simple y ofrece un buen rendimiento (Lai et al. (2019)). Así mismo, el algoritmo Baseline (CQCC) usado como referencia y tomado de ASVspoof 2017 (versión 2) utiliza GMM como clasificador. Es por este motivo, que en este trabajo se usa este clasificador para la detección de ataques de repetición en los experimentos. La decisión de si el discurso es genuino o repetido se calcula mediante la razón de probabilidad logarítmica, definida de la siguiente manera (Phapatanaburi et al. (2019)):

$$\lambda(O) = \log p(O|\lambda_{genuino}) - \log p(O|\lambda_{reproducido}) \tag{13}$$

donde O es el vector de características, y λgenuine y λreplay definen los GMM para las etiquetas de señales genuinas y reproducidas, respectivamente.

En algunos estudios se ha discutido que la información de magnitud y fase componen todas las características de una señal, mostrando a priori una mejora en la detección de muchos sistemas de clasificación de voz (Phapatanaburi et al. (2019)), (Murty and Yegnanarayana (2005)). En este trabajo, la combinación lineal sugerida en (Phapatanaburi et al. (2019)) se aplica para obtener un nuevo puntaje de decisión Lcomb:

$$L_{comb} = (1-\alpha)L_1 + \alpha L_2 \tag{14}$$

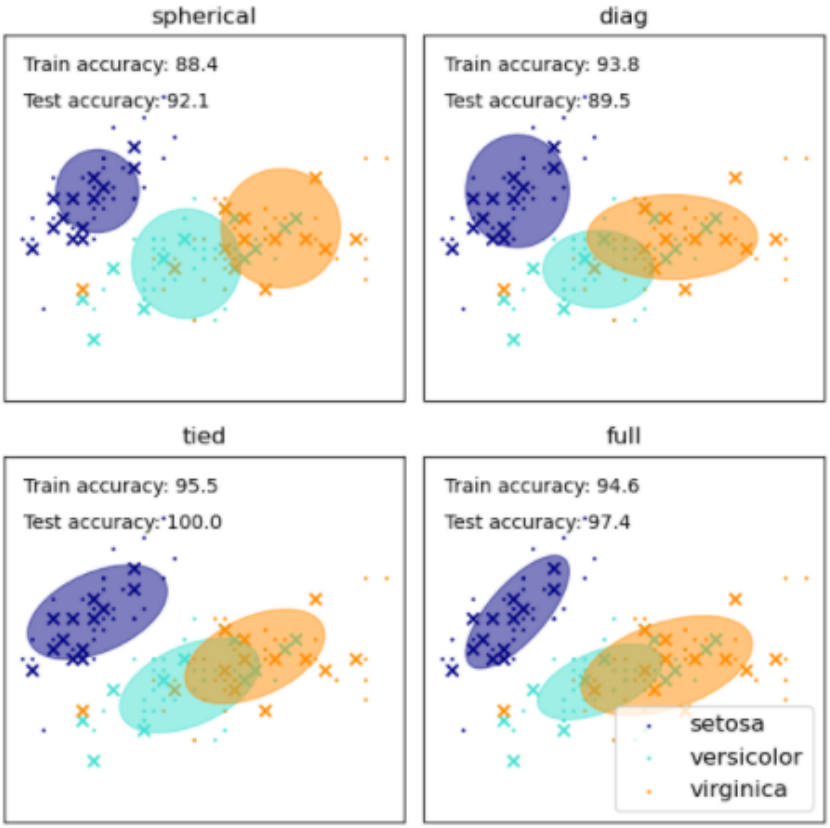
Donde α denota los coeficientes de peso, L1 es la probabilidad GMM de la primera función seleccionada, L2 es la probabilidad GMM de la segunda función seleccionada.

2.5.1. Covarianzas GMM. Según la teoría que se muestra en , existen al menos 4 covarianzas con las que se puede entrenar un modelo, las cuales son; esféricas, diagonales, ligadas y completas (estas se listan de menor a mayor rendimiento, siendo las esféricas las de menor y las completas las de mayor). Nota: Este diagrama expone el criterio y forma de estimación para el tratamiento de muestras en cada una de las covarianzas, para mas información Pedregosa et al. (2011).

2.6. Modelos de fusión

Nota: Este diagrama se presenta como propuesta para fusionar dos modelos distintos en donde el cálculo de su puntaje (Entrenamiento y testeo) se realizan por separado y al final mediante

Figura 8. Diagramas de las distintas covarianzas



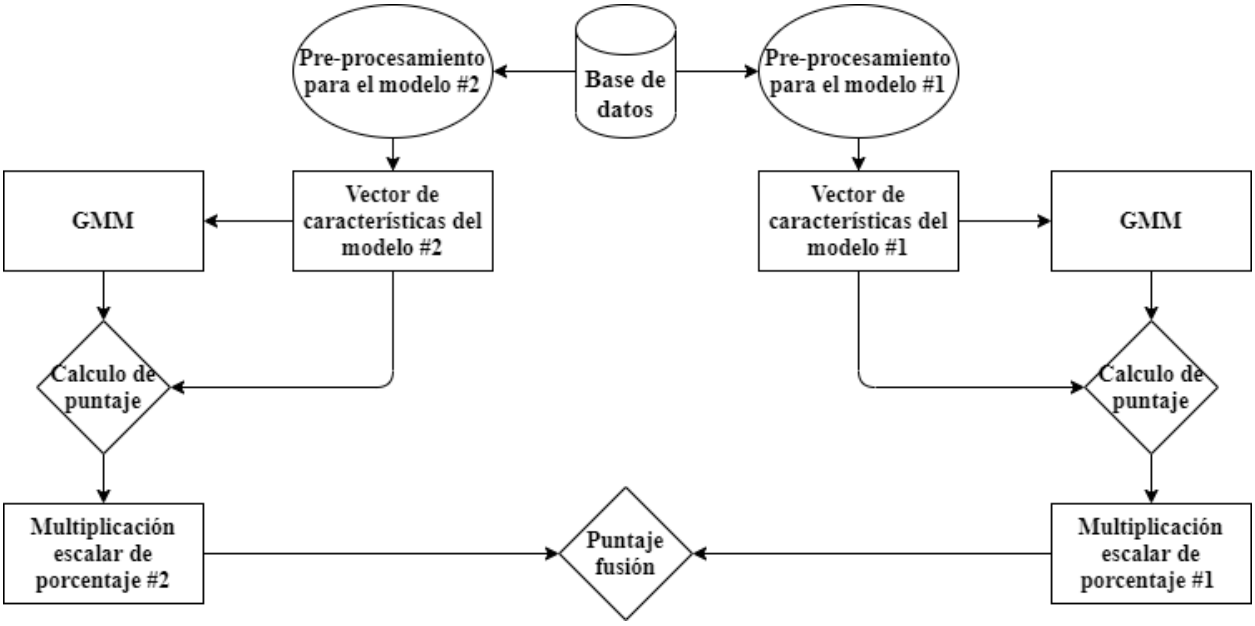
una sumatoria de pesos, se obtiene un puntaje de fusión (Chakroborty and Saha (2009)).

2.7. Criterio de evaluación

La tasa de error igual (EER) corresponde al punto de operación en donde son iguales las tasas de falso rechazo y falsa aceptación, este se utiliza como criterio de rendimiento. Además, en el código baseline obtenido de ASVspoof utilizan este criterio de rendimiento para evaluar este tipo de sistemas. La librería utilizada en Python para obtener este resultado se llama bob referencia

Nota: Se presentan las carpetas y códigos usados en el presente trabajo. En las carpetas de

Figura 9. Diagrama de flujo para el modelo propuesto de fusión

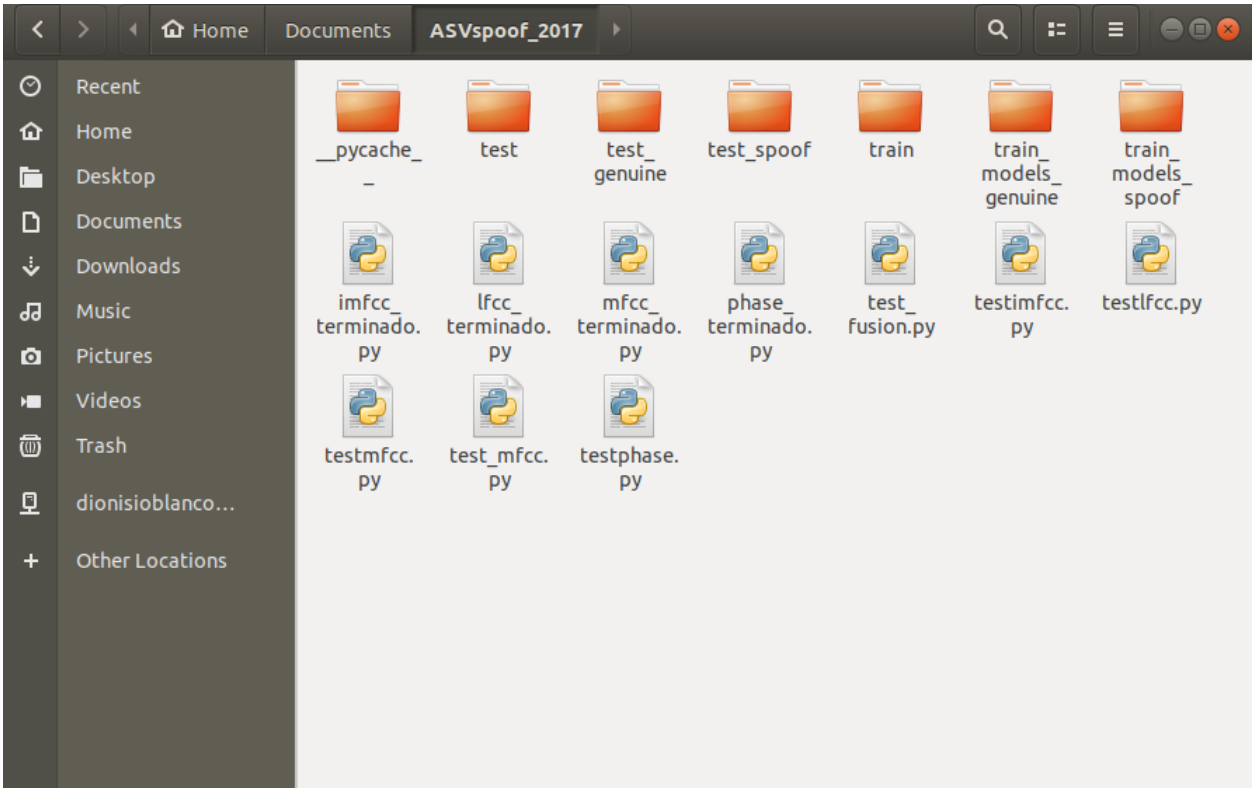


Train y Test se encuentran los archivos de audio de la base de datos ASVspoof 2017 (Versión 2), las carpetas que contengan en su nombre las palabras (*train* , *models*) son aquellas que guardan las cadenas de bytes de los modelos entrenados, cada uno de los modelos tiene un código en primera instancia con sus siglas por nombre, el entrenamiento y testeo se encuentran en los otros códigos como segunda instancia.

2.8. Plataforma

Como medio para compartir los códigos aquí presentados se uso GitHub el cual es un sistema ampliamente conocido para la gestión de códigos en linea. La primera versión del repositorio recibirá el nombre de andioe3t/ASVspoof-2017.

Figura 10. Distribución de carpetas presentadas para la portabilidad



3. Método

3.1. Modelos base

Todos los resultados de los modelos aquí evaluados, se realizaron con los datos de las sub-carpetas de entrenamiento y evaluación respectivamente, para estas pruebas no se tuvo en cuenta ningún dato del subconjunto de desarrollo. Para el estudio expuesto en este documento, se emplearon los CQCCs como baseline, al igual que otros documentos en los que se exponen estos desafíos (Kinnunen et al. (2017)), y los MFCCs como funciones de referencia para la tarea de detección de

ataques de repetición. Todas las funciones de extracción de características en magnitud, se obtienen utilizando un filtro pre énfasis con factor igual a 0.97, una longitud de cuadro de 20 ms, un desplazamiento de cuadro de 10 ms y aplicando la ventana Hamming. El número de muestras en la Transformada discreta de Fourier es de 512 con el que se calculo el espectro de magnitud con 256 componentes. El espectro en magnitud se procesa a través de un banco de 13 filtros triangulares, y el espaciado en frecuencia de cada uno de estos filtros está determinado por la ecuación característica de cada modelo. Al final del proceso se aplica la transformada discreta de coseno para obtener los vectores.

Las características Mel-RP se calculan utilizando una longitud de cuadro de 12,5 ms y un desplazamiento de cuadro de 5 ms. El número de muestras en la Transformada discreta de Fourier es de 256 con el que se calcula el espectro de fase con 128 componentes. El espectro de fase se normaliza mediante la función coseno y seno para obtener características de fase relativa (RP). Al final del proceso estas características RP se procesan a través de un banco de 19 filtros triangulares en la escala de frecuencia de Mel.

Todos los vectores de características para los modelos en magnitud tienen una dimensión de 13 y el modelo de fase de 19, ademas de esto la covarianza elegida en el entrenamiento como base o referencia fue la esférica.

En este trabajo se tuvieron en cuenta los componentes de velocidad (delta) y aceleración (doble delta) para los modelos en magnitud y las coordenadas en circulo unitario (seno y coseno) para el modelo en fase, debido a esto los vectores quedan del tamaño (13,13Δ,13ΔΔ) y (19sin,19cos).

3.2. Modelos de fusión

En este apartado se realizaron las tres fusiones propuestas en los objetivos, con los mismos valores en las variables de cada bloque como se especificó en la sección anterior, con la única diferencia que en este caso los componentes deltas fueron excluidos para probar el rendimiento puro y duro de cada uno de los modelos. El diagrama de bloques presentado en la figura 9 es el que se aplico para las pruebas en este segmento.

3.3. Complementos a los modelos

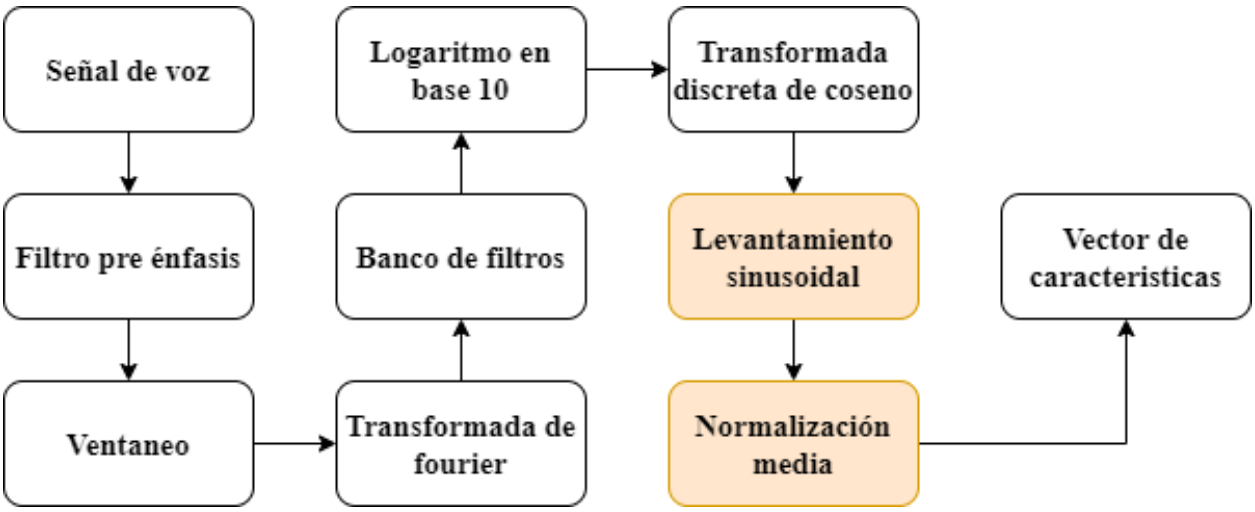
En esta sección se escogió el modelo de MFCC para realizar unas pruebas con algunos bloques complementarios (Levantamiento sinusoidal y normalización) usados en sistemas ASV, de modo que como objeto de experimento se observa la variación que estos bloques causan en el resultado para este modelo, por el contrario no se incluyen lo deltas y dobles deltas en estas pruebas, para evaluar unicamente la influencia de los dos códigos en cuestión sobre el modelo, así mismo estudiar si el rendimiento mejora o por el contrario se ve afectado.

Nota: En este diagrama de flujo se observa los mismos bloques que constituyeron los modelos base visto en la figura 1 , más los bloques complementarios en color.

3.4. Cambio de covarianzas en GMM

Ya para el final se quiso experimentar un factor del que poco se habla en los documentos, y es el cambio de covarianza en los modelos de entrenamientos para sistemas de aprendizaje automáticos. Se tomó la decisión de elegir el modelo de MFCC (por ser ampliamente usado) para así

Figura 11. Diagrama de flujo para el modelo complementario



observar los cambios que podrían generar las demás covarianzas.

4. Resultados

Tabla 2
Resultados en términos de EER para cada uno de los modelos

Modelo	EER(%)
CQCC	28.89
MFCC	35.68
IMFCC	30.81
LFCC	35.18
Mel-RP	49.98

Nota: Esta tabla presenta los porcentajes de error según el criterio de EER, para cada uno de los modelos de manera individual.

Nota: Esta tabla muestra los porcentajes de error según el criterio de EER, para cada uno de

Tabla 3
Resultados en términos de EER para los modelos de fusión

Fusion	EER (%)
MFCC + IMFCC	29.27
MFCC + LFCC	39.52
IMFCC + Mel-RP	38.84

los modelos de fusión propuestos, en este caso el α que definía el peso de cada uno de los modelos para obtener L_{comb} se estableció en el valor de 0.7.

Tabla 4
Resultados en términos de EER para el modelo complementario MFCC

Levantamiento sinusoidal	Normalización media	EER (%)
✓		36.43
	✓	37.88
✓	✓	49.43

Nota: En esta tabla se señala con un ✓el bloque que se introdujo en cada una de las pruebas, el diagrama de bloques implementado aquí es el de la figura 11

Tabla 5
Resultados en términos de EER con las otras 3 covarianzas con el modelo de MFCC

Diagonal	Ligada	Completa	EER (%)
✓			40.69
	✓		38.12
		✓	38.32

Nota: En esta tabla se expone con un ✓la covarianza utilizada en cada una de las pruebas.

5. Discusión

Esta sección iniciara hablando de la librería que se usó para obtener el EER (Equal Error Rate), recibe el nombre de bob (Anjos et al. (2017) ; Anjos et al. (2012)); la cual es una herramienta gratuita para el procesamiento de señales y machine learning, soporta los lenguajes de Python (utilizado en este trabajo) y C++. La ventaja de usar esta herramienta es; su buena optimización y su amplio catálogo de paquetes para desarrollo de machine learning. Dicho lo anterior existe una garantía y buena fiabilidad de confianza en la obtención de los resultados. Acerca de los resultados observados en la tabla 2, en ella se muestran los puntajes obtenidos por cada uno de los modelos, y es aquí donde se hace visible la propuesta de evaluar estos modelos ya que estos resultados son aún más deficientes en general que el obtenido por el modelo baseline (CQCC) que con un puntaje de 28.89 es el mejor y 49.98 para Mel-RP como el peor. Debido a esto se proponen las fusiones ofreciendo una alternativa de mejora a los resultados individuales.

En la tabla 3 , se muestran los valores (EER %) para las fusiones propuestas en este documento, una de las preguntas formuladas para este estudio era si dos métodos (escala y su inversa) podían tener un mejor desempeño; para ello se presento la fusión MFCC e IMFCC (modelo auditivo humano y su modelo evolutivo contrario), la cual con su puntaje de 29.27 logro ser la mejor opción en el apartado de fusiones y también supero los puntajes individuales de los modelos implicados, por tanto la respuesta a la pregunta es asertiva, realmente se obtiene un buen comportamiento comparándose con los resultados individuales de MFCC e IMFCC. Sin embargo, su rendimiento no logro superar el propuesto por el baseline (CQCC), haciendo que al menos en el apartado de es-

tas fusiones no se supere el rendimiento del modelo referencia (baseline). Otra de las preguntas en este apartado era si era necesario extraer todas las características de una señal, es decir información tanto en fase como en magnitud, a lo cual también obtuvo una respuesta algo controvertida, con su puntaje de 38.84 la fusión de (IMFCC y Mel-RP) obtiene un rendimiento superior al modelo de fase individual presentado pero no el de magnitud (49.98 para Mel-RP y 30.81 para IMFCC), sin embargo sigue siendo mejor que la fusión propuesta para corroborar la hipótesis (39.52 para la fusión de MFCC y LFCC) las cuales son solo ofrecen información en magnitud.

A medida que se avanzaba en el desarrollo del trabajo se observaron distintos componentes que podrían cambiar el rendimiento de los modelos propuestos, para la sección de estudio que arrojo como resultados los expuestos en la tabla 4, se estudiaron dos bloques presentados como alternativas para sistemas de verificación del hablante, la pregunta que se planteo en este caso es; ¿Estos bloques ayudaran contra ataques de tipo reproducción?, con esta premisa se realizaron 3 pruebas; la primera de ellas agregando el bloque de levantamiento sinusoidal, la segunda con el bloque de normalización media y por ultimo los dos bloques agregados al diagrama base. De los resultados obtenidos por estos experimentos se observa un deterioro en términos de EER en comparación con el modelo principal; 36.43 con levantamiento sinusoidal, 37.88 con la normalización media y 49.43 con ambos bloques agregados. En comparación con el modelo referencia o de experimento que se tomo para estas pruebas (MFCC) que de manera individual obtuvo un puntaje de 35.68, se vuelve un opción poco favorable el optar por agregar estos bloques.

Por lo mostrado en la tabla 5 erróneamente se podría llegar a pensar que elegir la covarianza ligada sería la mejor opción, sin embargo, se quiso ir más allá y se realizo el mismo experimento

para los demás modelos en magnitud, obteniendo los siguientes resultados:

Tabla 6
Resultados en términos de EER de los modelos en magnitud individuales con covarianza ligada

Modelo	EER (%)
MFCC	38.12
LFCC	43.98
IMFCC	40.13

Nota: En esta tabla se exhiben los resultados obtenidos por los tres modelos en magnitud con una covarianza ligada en el modelo de mezclas gaussianas para los datos de entrenamiento, los cuales se pueden comparar con los enseñados en la tabla 2 que tiene los resultados de estos modelos con una covarianza esférica.

La metodología usada para extraer los resultados de las tablas 5 y 6 es la postulada en la sección 3.4 (Cambio de covarianzas en GMM) de este documento. Esta consiste en modificar una fracción de la línea de código numero 65 de los algoritmos con el prefijo test, allí se cambia la variable $covariance\_type =' spherical'$ por: 'diag' (diagonal), 'tied' (ligada) o 'full' (completa). Las variables para cada uno de los modelos fueron las mismas que se eligieron para las pruebas de la sección 3.1 (Modelos base), solo cambió el tipo de covarianza, para evaluar su efecto en el resultado individual para cada uno de los modelos.

6. Conclusiones

En este trabajo se plantearon distintos recursos además durante el desarrollo de este se fue experimentando y probando algunos otros factores que nos ofrecía la literatura anexada; con los resultados obtenidos podemos inferir distintas circunstancias que quedaron en evidencia. Como

objetivos se planteo el estudio, desarrollo, implementación y evaluación de los distintos modelos aquí propuestos, como punto de partida las fusiones ofrecieron un gran desempeño sin ser el mejor, pero representando un acierto de elección para responder las preguntas que se plantearon al inicio de proyecto, el desempeño de los modelos fusionados fue mejor en términos generales a excepción de la fusión de MFCC y LFCC ofreciendo dos alternativas (MFCC + IMFCC, IMFCC + Mel-RP) a los modelos individuales. El resultado de MFCC e IMFCC fusionados fue sin lugar a dudas aquel que mas se acerco al registro del rendimiento conseguido con el Baseline (CQCC), es posible que el resultado de L_{comb} sea aún mejor cambiando el factor de mezcla α por uno mas adecuado.

El desempeño que se presentó al agregar los dos bloques complementarios (levantamiento sinusoidal y normalización media) no fue del todo satisfactorio, esto se debe a que en sistemas de verificación del hablante la prioridad es el reconocimiento de una señal de voz en específico, es decir esta señal tiene que ser lo mas limpia posible; el levantamiento sinusoidal quita el énfasis en los coeficientes lo que se traduce a un mejor reconocimiento en entornos ruidosos, sin embargo en un ataque de reproducción de voz, la detección del ruido es sumamente importante. Por otro lado cuando se normalizan características o los datos obtenidos de una señal de voz, este aporta un equilibrio al espectro de la señal y ayuda a mejorar la relación señal a ruido (SNR) por sus siglas en ingles, aquí volvemos a identificar el como se elimina parcialmente el ruido lo cual es extremadamente necesario para este tipo de sistemas, pero poco beneficioso, y ofreciendo información innecesaria para la detección de ataques de tipo reproducción.

En cuanto a las covarianzas evaluadas en este documento seria preciso decir que según como sea la distribución de las muestras al aplicar el modelo de mezclas gaussianas, como se observa

en la figura 8, se debe escoger el tipo de covarianza que mejor se ajuste a los datos, para sacar el resultado mas optimo a cada modelo.

7. Recomendaciones

Una buena alternativa para continuar con el trabajo aquí presentado, consiste en cambiar variables: como el número de filtros o la elección del numero de coeficientes en el bloque de la transformada discreta de coseno, que pueden ofrecer una variante en los resultados obtenidos, considerándose como propósito de estudios posteriores. Además de estas variables en la literatura se pueden encontrar un gran número de modelos aparte de los aquí planteados; estos se pueden fusionar bajo el concepto presentado en la sección 3.2 (Modelos de fusión) y exponer sus resultados. Por último, un bloque que puede ayudar a mejorar el rendimiento del modelo Mel-RP según la literatura es el de Pseudo Pitch Synchronization.

Referencias Bibliográficas

Anjos, A., El-Shafey, L., Wallace, R., Günther, M., McCool, C., and Marcel, S. (2012). Bob: a free signal processing and machine learning toolbox for researchers. In *Proceedings of the 20th ACM international conference on Multimedia*, pages 1449–1452.

Anjos, A., Günther, M., de Freitas Pereira, T., Korshunov, P., Mohammadi, A., and Marcel, S. (2017). Continuously reproducing toolchains in pattern recognition and machine learning experiments.

Chakroborty, S. and Saha, G. (2009). Improved text-independent speaker identification using fused mfcc & imfcc feature sets based on gaussian filter. *International Journal of Signal Processing*, 5(1):11–19.

Chen, Z., Xie, Z., Zhang, W., and Xu, X. (2017). Resnet and model fusion for automatic spoofing detection. In *INTERSPEECH*, pages 102–106.

Delgado, H., Todisco, M., Sahidullah, M., Evans, N., Kinnunen, T., Lee, K., and Yamagishi, J. (2018). Asvspoof 2017 version 2.0: meta-data analysis and baseline enhancements.

Ergünay, S. K., Khoury, E., Lazaridis, A., and Marcel, S. (2015). On the vulnerability of speaker verification to realistic voice spoofing. In *2015 IEEE 7th International Conference on Biometrics Theory, Applications and Systems (BTAS)*, pages 1–6. IEEE.

Font, R., Espín, J. M., and Cano, M. J. (2017). Experimental analysis of features for replay attack detection-results on the asvspoof 2017 challenge. In *Interspeech*, pages 7–11.

Hanilçi, C. (2017). Features and classifiers for replay spoofing attack detection. In *2017 10th International Conference on Electrical and Electronics Engineering (ELECO)*, pages 1187–1191. IEEE.

Kinnunen, T., Sahidullah, M., Delgado, H., Todisco, M., Evans, N., Yamagishi, J., and Lee, K. A. (2017). The asvspoof 2017 challenge: Assessing the limits of replay spoofing attack detection.

Lai, C.-I., Abad, A., Richmond, K., Yamagishi, J., Dehak, N., and King, S. (2019). Attentive filtering networks for audio replay attack detection. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6316–6320. IEEE.

Li, D., Wang, L., Dang, J., Liu, M., Oo, Z., Nakagawa, S., Guan, H., and Li, X. (2018). Multiple phase information combination for replay attacks detection. In *INTERSPEECH*, pages 656–660.

Muckenhirn, H., Korshunov, P., Magimai-Doss, M., and Marcel, S. (2017). Long-term spectral statistics for voice presentation attack detection. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 25(11):2098–2111.

Murty, K. S. R. and Yegnanarayana, B. (2005). Combining evidence from residual phase and mfcc features for speaker recognition. *IEEE signal processing letters*, 13(1):52–55.

Nakagawa, S., Wang, L., and Ohtsuka, S. (2011). Speaker identification and verification by combi-

ning mfcc and phase information. *IEEE transactions on audio, speech, and language processing*, 20(4):1085–1095.

Patil, H. A. and Kamble, M. R. (2018). A survey on replay attack detection for automatic speaker verification (asv) system. In *2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 1047–1053. IEEE.

Patil, H. A., Kamble, M. R., Patel, T. B., and Soni, M. H. (2017). Novel variable length teager energy separation based instantaneous frequency features for replay detection. In *INTERSPEECH*, pages 12–16.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Phapatanaburi, K., Wang, L., Nakagawa, S., and Iwahashi, M. (2019). Replay attack detection using linear prediction analysis-based relative phase features. *IEEE Access*, 7:183614–183625.

Todisco, M., Delgado, H., and Evans, N. (2017). Constant q cepstral coefficients: A spoofing countermeasure for automatic speaker verification. *Computer Speech & Language*, 45:516–535.

Witkowski, M., Kacprzak, S., Zelasko, P., Kowalczyk, K., and Galka, J. (2017). Audio replay attack detection using high-frequency features. In *Interspeech*, pages 27–31.

Wu, Z., Khodabakhsh, A., Demiroglu, C., Yamagishi, J., Saito, D., Toda, T., and King, S. (2015). Sas: A speaker verification spoofing database containing diverse attacks. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4440–4444. IEEE.

Wu, Z., Yamagishi, J., Kinnunen, T., Hanilçi, C., Sahidullah, M., Sizov, A., Evans, N., Todisco, M., and Delgado, H. (2017). Asvspoof: the automatic speaker verification spoofing and countermeasures challenge. *IEEE Journal of Selected Topics in Signal Processing*, 11(4):588–604.

Yang, J., Das, R. K., and Li, H. (2018). Extended constant-q cepstral coefficients for detection of spoofing attacks. In *2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 1024–1029. IEEE.

Zhou, X., Garcia-Romero, D., Duraiswami, R., Espy-Wilson, C., and Shamma, S. (2011). Linear versus mel frequency cepstral coefficients for speaker recognition. In *2011 IEEE Workshop on Automatic Speech Recognition & Understanding*, pages 559–564. IEEE.