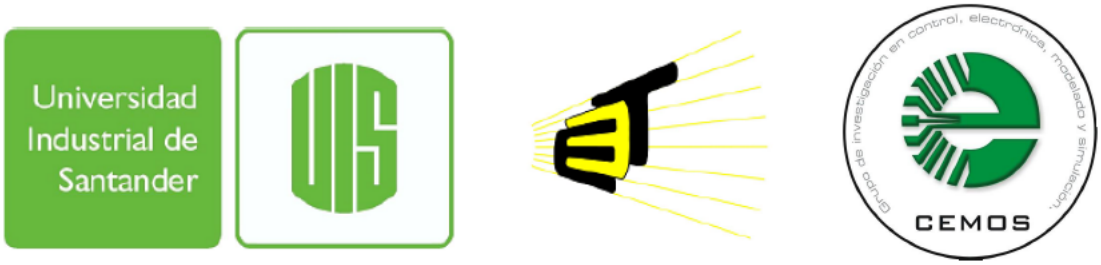
DESARROLLO DE UN SISTEMA DE VERIFICACIÓN DEL HABLANTE BASADO EN MODELOS DE MEZCLAS GAUSSIANAS

ANDRÉS FELIPE MENDOZA MARÍN
DAGOBERTO PORRAS PLATA



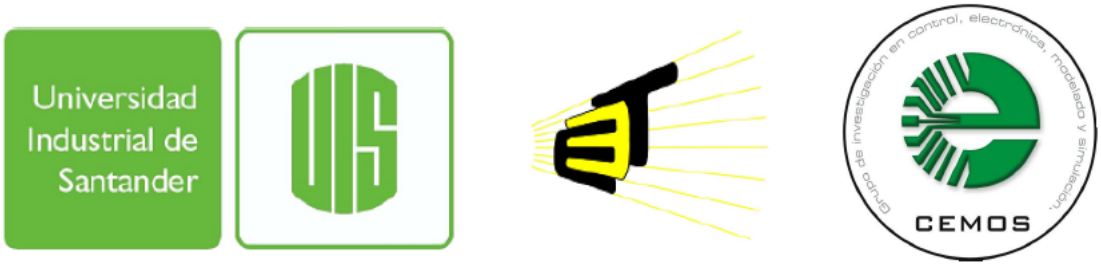
UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FÍSICO-MECÁNICAS
ESCUELA DE INGENIERÍAS ELÉCTRICA, ELECTRÓNICA Y DE TELECOMUNICACIONES
BUCARAMANGA
2016

DESARROLLO DE UN SISTEMA DE VERIFICACIÓN DEL HABLANTE BASADO EN MODELOS DE MEZCLAS GAUSSIANAS

ANDRÉS FELIPE MENDOZA MARÍN
DAGOBERTO PORRAS PLATA

Trabajo de grado presentado como requisito parcial para obtener el título de ingeniería electrónica

Director
FRANKLIN ALEXÁNDER SEPÚLVEDA SEPÚLVEDA
PhD en Procesamiento de Señales



UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FÍSICO-MECÁNICAS
ESCUELA DE INGENIERÍAS ELÉCTRICA, ELECTRÓNICA Y DE TELECOMUNICACIONES
BUCARAMANGA
2016

En agradecimiento dedico este trabajo

En primer lugar a mi madre Victoria Plata y mi padre Dagoberto Porras, que a lo largo de la vida me han formado y educado para afrontar grandes desafíos y ser un gran ser humano. Además de brindarme su cariño y apoyo todo el tiempo.

A mi hermana, Stefany Porras, que me ha aconsejado y me ha servido como ejemplo, para crecer y tomar buenas decisiones.

A mi hermano, Nicolás Porras, que ha sido un ser humano maravilloso además he contado con todo su apoyo, cariño y gracias por recordarme nunca renunciar a pesar de los obstáculos.

A mi compañero y amigo Felipe Mendoza, por su dedicación y hacer parte en el desarrollo de este trabajo y la presentación del articulo.

A mi director de este trabajo, el profesor Franklin Alexander Sepúlveda, que con su gran calidad humana y su dedicación, ha hecho de esta una experiencia extraordinaria. Además de brindarme la oportunidad de trabajar en este proyecto.

A Nía y A todos mis amigos que han sido parte y han hecho posible de cualquier manera la realización de éste proyecto.

Gracias a todos.

Dagoberto Porras Plata

Dedico este trabajo,

Inicialmente a Dios por darme la fortaleza y salud para culminar esta importante meta en mi vida

A mi madre Neyla Marin, a mi padre Edilberto Mendoza y a mi hermana Daniela , por los consejos, valores y virtudes que me han hecho una mejor persona, además de su esfuerzo y amor que los convierten en mi principal motivación

A mi tía Elsa Marin, por ser tan especial para mi con su sabiduría, apoyo y amor en sus consejos que día a día me permiten enfocarme y cumplir mis sueños.

A mi compañero y amigo Dagoberto Porras, por su gran aporte y dedicación a este proyecto

Al director de este proyecto, Franklin Sepulveda, por sus conocimientos fundamentales para el trabajo, su experiencia y dedicación, que me hicieron crecer profesional y personalmente.

A mis amigos y demás personas que en algún modo influyeron para hacer de este proyecto una gran experiencia.

Sinceramente gracias.

Andres Felipe Mendoza

Índice general

Índice de figuras

Pág

RESUMEN

Título: "Desarrollo de un sistema de verificación del hablante basado en modelos de mezclas Gaussianas "*

Autores:
Andrés Felipe Mendoza Marín
Dagoberto Porras Plata**

Palabras claves: GMM, UBM, Python, ALIZE, LIA_RAL, Verificación del Hablante

Descripción

Este proyecto presenta y describe la implementación de un sistema de verificación automático del hablante basado en modelos de mezclas Gaussianas bajo el paradigma de la razón de verosimilitud. Para este objetivo se utilizaron las señales de voz proveídas por la base de datos MIT *Device Speaker Verification Corpus*. Una base de datos recolectada por el MIT, la cual se encuentra destinada a problemas de verificación robusta del hablante en dispositivos móviles en ambientes ruidosos y con limitación de datos para entrenamiento. De otra parte, se recogió una base de datos de 30 hablantes masculinos con el fin de realizar pruebas adicionales. El sistema se desarrolla mediante el uso del motor de entrenamiento de código abierto *ALIZE*.

A modo de trabajo adicional, se plantea el desarrollo de un experimento en el que se evalúa la influencia del uso de un modelo de referencia por aparte para hombres y mujeres, respecto al uso de un solo modelo global para los dos géneros. Los resultados indican que el no alimentar el sistema de verificación del hablante con la información acerca del género de la persona en cuestión afecta el desempeño del sistema. Las pruebas se realizan usando señales proveniente de diferentes ambientes y niveles de ruido. Por último observar el desempeño del sistema debido a la cantidad de distribuciones Gaussianas seleccionadas para el entrenamiento del sistema global de referencia.

*Trabajo de grado

**Facultad de Ingenierías Físico-mecánicas, Escuela de Ingenierías Eléctrica, Electrónica y Telecomunicaciones. Director. PhD Franklin Alexánder Sepúlveda Sepúlveda

ABSTRACT

Tittle: "Development of a speaker verification system based on Gaussian mixture models"*

Authors:
Andrés Felipe Mendoza Marín
Dagoberto Porras Plata**

Keywords: GMM, UBM, Python, ALIZE, LIA_RAL, Speaker Verification

Description
This work presents and describes the implementation of an automatic speaker verification based on Gaussian mixture models under the paradigm of the likelihood ratio. Voice signals provided by the database MIT Device Speaker Verification Corpus was used for this purpose. A database collected by MIT, which is intended to problems robust speaker verification on mobile devices in noisy environments and with limited training data. Furthermore, it collected a database of 30 male speakers in order to further testing. The system is developed using the engine open source training ALIZE.

By way of further work, the development of an experiment in which the influence of using a reference model for evaluating male and female apart, on the use of a single global model for both genders. The results indicate that not feeding the speaker verification system with information about the gender of the person in question affects system performance. The tests are performed using signals from different environments and noise levels. Finally observing system performance due to the amount of Gaussian distributions for training selected global reference system.

*Degree work

**Faculty of Physical-Mechanical Engineering, School of Electrical Engineering, Electronics and Telecommunications. Director. PhD. Franklin Alexander Sepulveda

INTRODUCCIÓN

La señal acústica de voz además de portar información relacionada con la identidad del hablante, posee información de tipo emocional, lingüístico, fisiológico y sociolingüístico, entre otros tipos. Dentro de toda la información que contiene la señal de voz, se busca modelar aquella que atañe a la identidad de la persona, con la que se implementan sistemas biométricos. Siendo así los seres humanos han desarrollado en el campo forense la capacidad para realizar las tareas de reconocimiento oral que a su vez son difícilmente emuladas por los sistemas automáticos [41], sin embargo este patrón biométrico ha tenido gran acogida en distintos campos de la industria que ha sido un tema importante en materia de investigaciones, lo que ha conllevado al desarrollo de sofisticados algoritmos de reconocimiento automático del hablante que han mejorado el desempeño de estos sistemas en relación a hace un par de décadas.

El reconocimiento de hablantes se divide en dos partes: la identificación y la verificación. La primera consiste en reconocer a un hablante de entre un grupo de diferentes hablantes previamente modelados por el sistema; la segunda, que corresponde al campo del presente trabajo de grado, determina si el hablante es quien dice ser. En el primer caso es un sistema de tipo cerrado; es decir, su respuesta consiste en determinar cual de entre la cantidad reducida de N posibilidades (personas) es el respuesta. Realiza N comparaciones. De otra parte, en el caso de verificación del hablante se busca determinar si es o no la persona en cuestión, dentro de las infinitas posibilidades. Con el avance de la tecnología, ha incrementado el potencial de aplicación de sistemas de verificación del hablante, los cuales se pueden encontrar en sistemas como: el acceso a computadores por medio de un log_in acústico, operaciones bancarias a través de líneas telefónicas [33, 23], mecanismos de autenticación remota [13, 2], aplicaciones de seguridad en dispositivos móviles y finalmente en la línea acústica forense como apoyo a procesos judiciales donde se requieren procesos de verificación del hablante (cotejos de voz, como es llamado en Colombia).

CAPÍTULO 1

PROPUESTA DE INVESTIGACIÓN

1.1. JUSTIFICACIÓN

Las comunicaciones por medio de la telefonía móvil se ha ido incrementando de forma notable, tal es punto que países como Colombia y Ecuador la cantidad de líneas activas de celular supera a la población [11]. Sin embargo, son cada vez más los crímenes que son cometidos a través de llamadas realizadas por teléfonos móviles (extorsiones, secuestros, amenazas, casos de corrupción, entre otros). Por lo tanto, en materia de seguridad los sistemas biométricos, en particular los basados en la voz, se convierten en una importante herramienta para verificar e identificar los autores de dichos crímenes, en caso de que se hayan cometido. Además, la voz como parámetro biométrico es un buen candidato para ser usada en la identificación de usuarios ya que contiene información tanto de carácter físico como emocional, la comunicación oral suele ser natural para el usuario, y, su adquisición es bastante natural y económica. Esto ha llevado a que se incremente la importancia y el uso de sistemas de reconocimiento del hablante en mecanismos de seguridad.

Los métodos de verificación del hablante se clasifican, comúnmente, en las siguientes cuatro categorías: auditiva, auditiva-espectrográfica, fonético-acústico y automático [32]. A pesar de los comentarios desfavorables ofrecidos por la comunidad científica respecto al método de verificación de hablantes basado en el espectrograma, el mencionado método aún hace parte del repertorio de procedimientos en varios países del mundo; dentro de los que se incluye Colombia [4, 40]. En Colombia, además del método espectrográfico se incluye también el método auditivo-perceptivo y el acústico-fonético, dando lugar al método combinado clásico [4]. Estos métodos

usan observaciones y mediciones que dan lugar a la subjetividad, especialmente en el caso del método auditivo-perceptivo y espectrográfico, lo cual afecta la entrega de veredictos respecto a la identidad de una persona de forma objetiva y confiable. Además, dificulta en gran manera la repetibilidad y la verificación de los experimentos por parte de la defensa y/o la comunidad científica en caso de ser requerida. En síntesis, los tipos de anteriormente mencionados son susceptibles de mejora en los siguientes ámbitos: 1) Incremento en la objetividad; 2) Incremento de la confiabilidad; 3) Incremento en la eficiencia.

Otro elemento que afecta el desempeño de este tipo de sistemas corresponde al género. El ingresar al sistema a modo de información a priori el género de la persona, en cuestión es una acción que reduciría la variabilidad causada por este mismo elemento, lo cual es una práctica común al menos en el ámbito forense; sin embargo, su efectividad podría verse afectada en casos en los que no se cuente o no se pueda confiar en la información de género. En particular, en Colombia el decreto 1227 del 4 de junio de 2015 permite corregir la información acerca del sexo en el Registro Civil [*], y por consiguiente también en la identificación. En el presente trabajo se analiza la influencia del género en sistemas de verificación del hablante.

Buena parte de estas falencias son suplidas mediante el uso del método automático, los cuales son sistemas que utilizan modelos probabilísticos para representar el comportamiento acústico de la señal de voz perteneciente a los hablantes. Estos sistemas pueden requerir una considerable cantidad de datos. De otra parte, el costo de los sistemas de verificación es elevado. En particular, en el sistema información de contratación nacional de la República de Colombia se reporta la licitación de compra de equipo de cotejo de voces. Se adquiere el software BatVox[**] fabricado por la empresa *Agnitio* a un alto costo. Por tanto, el contar con un sistema de costo accesible es un problema por resolver de notable importancia.

1.2. ANTECEDENTES

Un sistema de verificación se enfoca solamente en verificar si el hablante proporciona un patrón que coincida con uno conocido, por lo tanto la respuesta del sistema simplemente tendrá dos opciones, verdadero o falso. Existen varios métodos

[*] http://wp.presidencia.gov.co/sitios/normativa/*decretos/2015/*
[**] http://www.agnitio-corp.com/products/government/voice-recognition-system

de VH (*verificación del hablante*): método auditivo, método auditivo-espectrográfico, fonético-acústico y automático.

Método Auditivo: Este método consiste en aplicar las capacidades que desarrollan profesionales para identificar personas por medio de la voz, en Colombia es comúnmente utilizada por ligüistas y fonoaudiólogos. Los parámetros que caracterizan este método se denominan de alto nivel tales como el dialecto, el tipo de fonemas utilizados, la velocidad del habla y la entonación. Sin embargo se encuentran factores que ponen en duda la eficiencia del método tales como la familiaridad con el hablante, la duración de la muestra, el contexto, el grado de similitud entre hablantes en el enfoque auditivo y el grado de entrenamiento del personal profesional en este método.

Método Auditivo-espectrográfico: Este método consiste en comparar la voz de prueba y la voz cuya fuente es conocida, por medio de los segmentos de voz de cada una. El objetivo de este es encontrar las diferencias y similitudes, y resaltar aquellos segmentos que pueden brindar información relevante para el análisis, esto se realiza por medio de un examen visual en donde los patrones de la voz están representadas por un espectrograma que está compuesto por los segmentos adquiridos, este examen es desarrollado por una persona entrenada para tal tarea. Por otro lado la comunidad científica descalifica este método para implementar un sistema de verificación del hablante ya que no se tiene certeza de su confiabilidad.

Método Acústico-Fónetico: Es principalmente utilizado en el campo de la fonética, los profesionales que se desempeñan en este, realizan mediciones de la señal de la voz que corresponden a información relevante para facilitar la identificación de la personas y logran analizar estadísticamente los resultados de la misma. De este método se resalta que la información obtenida es en algunos casos robusta ante efectos de distorsión que puede provocar el canal. Por otro lado el principal problema de este método es que para llevar a cabo todo el proceso se requiere de gran cantidad de horas para su desarrollo.

Método Automático: Para el desarrollo de este método se crean modelos estadísticos de la voz del hablante y de la voz de prueba con el objetivo de ser

comparados. Para esto comúnmente se usa un tercer modelo que es usado como referencia, este es denominado como UBM (*Universal background Model*), que en general está compuesto por un amplio modelo de mezclas Gaussianas que contienen patrones de la voz de la población que es utilizada como referencia. De esta forma el método automático consiste en comparar la señal de voz del hablante con una voz conocida y con un modelo de referencia con el fin de encontrar con cual se obtiene mayor similitud. Dentro de este método se destacan las etapas de representación y extracción de características, entrenamiento y comparación de los modelos. Es este el tipo de métodos en el que se enmarca el presente trabajo de grado.

Con el desarrollo de la tecnología se ha ido incrementando el potencial de los sistemas de verificación oral como una herramienta biométrica, de manera que este campo se ha centrado en desarrollar técnicas para el reconocimiento automático del hablante por medio de algoritmos, que principalmente se aplicaron en el campo de las telecomunicaciones (1985) [14], por ejemplo en (1993) *Sprint Corporation* ofreció "*the Voice FonceCard calling card*", donde cada usuario que requería acceso a una red de larga distancia pasaba por un sistema de reconocimiento del hablante. Cerca al siglo XXI, se trataron técnicas de modelado probabilístico para seleccionar la información de las características del hablante y de esta manera ser útil para el sistema de verificación, quizás una de las más relevantes ha sido el modelo de mezclas gaussianas (GMM) y alguna muestra de estos trabajos se puede encontrar en [39].

Después del 2000, numerosas empresas se dedicaron al desarrollo de sistemas de verificación e identificación del hablante anunciando un gran potencial para esta tecnología en distintas aplicaciones [35]. Aunque en el 2003, se reportaba en [7] la imposibilidad para caracterizar de forma única a una persona por su voz con absoluta certeza, luego de esto, como indica la revista [8], se observó un progreso impresionante. Esto se puede apreciar en las campañas de evaluación [1, 37, 36], del *National Institute of Standards and Technology* (NIST). Allí se presentan los grandes avances en este campo, entre ellos se encontraban las técnicas para modelar la variabilidad entre sesiones como *factor analysis* (FA) o *nuisance attribute projection* (NAP) [26, 9], y finalmente la aparición del modelo estadístico *Gaussian mixture model universal background model* (GMM-UBM), ampliamente usado en sistemas que se encuentran en el estado del arte.

Este tipo de sistemas están sometidos a varios factores que disminuyen el desempeño de los mismos. Entre ellos se mencionan: variabilidad intra-hablantes [40], ruido

acústico, condiciones emocionales, manera de hablar [38], el medio a través del cual se capta la señal de voz [3] y el medio ambiente, entre otros. Debido a esto es latente la preocupación por hacer los sistemas de verificación del hablante cada vez más robustos.

Una configuración robusta consiste en la extracción de coeficientes MFCC, seguido de un filtrado RASTA (*Relative Spectra*), cálculo de parámetros de primera derivada, detección activa de voz, mapeo de características y finalmente normalización respecto a la media y varianza. Existen métodos adicionales de reconocimiento del hablante de tipo robusto. Uno de ellos es el de proyección de atributos no convenientes (*Nuisance Attribute Projection*, NAP), que corresponde a una técnica de compensación que actúa sobre supervectores. Para mayor información ver [16, 10]. De otra parte Se ha utilizado el análisis factorial conjunto (*Joint Factor Analysis, JFA*) en [27, 12] con el fin de ofrecer una posible solución al problema de variabilidad entre sesiones. En [22] por otra parte, se plante el uso de análisis factorial sobre los vectores de parámetros acústicos en lugar de aplicarlos sobre los parámetros de supervectores, y, en [18] se utiliza el análisis de factores sobre modelos auto-asociativos de redes neuronales en lugar de aplicarlo sobre el modelo de mezclas Gaussianas que conforma el modelo de referencia. Sin embargo, desde el punto de vista de reconocimiento del hablante no se ha concluido exactamente en cuáles características buscar dentro de la señal de voz con fines de robusticidad [28]. Son dos elementos fundamentales en el proceso de verificación de hablantes: 1) el método utilizado para el modelado de los hablantes; 2) la estrategia utilizada para la representación de la señal de voz.

A modo general, existen dos elementos que hacen parte de los sistemas VH. La primera está relacionada con la representación de la señal de voz y la segunda con el modelado estadístico de los datos. Idealmente los parámetros utilizados en voz, deberían cumplir con las siguientes condiciones [40]: a) gran variabilidad entre-hablantes y baja variabilidad intra-hablante; b) ser robustos ante condiciones de ruido y distorsión por efectos del canal; c) que ocurran de manera frecuente y natural en el habla; d) fáciles de medir; e) difíciles de imitar por otras personas; f) que no se vean afectados por cambios en la salud y en la edad. Una forma de clasificar las características es como sigue [28]: 1) características de tiempo corto y relacionadas con la fuente de voz; 2) características espectro-temporales; 3) características prosódicas y de información de alto nivel.

En cuanto a la construcción del modelado acústico de los hablantes, el método comúnmente usado corresponde al basado en UBM (*Universal Background Model*) [21]. Donde el elemento UBM es básicamente un gran modelo de mezclas Gaussianas (GMMS) que representa las propiedades de la voz de la población que se utiliza como referencia. Para mejorar el desempeño se han desarrollado estrategias adicionales: supervectores GMM en [9], presentando mejoras; análisis de conjunto de factores (JFA) en [46] para modelar la variabilidad entre los modelos GMM, del hablante y la sesión con el fin de mejorar la robusticidad de los sistemas de verificación del hablante.

Aunque existe aplicaciones de software que permiten realizar verificación del hablante, *e.g. Batvox*, su precio es alto. Además es difícil contar con plataformas de rápida implementación que permitan probar nuevos algoritmos y realizar nuevos avances en tareas de verificación del hablante. En consecuencia, la comunidad científica se ha enfocado en la implementación de éste tipo de sistemas en multiplataformas de tipo open source, logrando reducir costos y aportando sus resultados al alcance de cualquier persona, de esta manera se ha logrado posicionar en el estado del arte para sistemas de verificación del hablante [13]. Además la plataforma *ALIZE* (http://mistral.univ-avignon.fr/) ofrece ventajas de tipo practico y puede ser utilizado para realizar la estimación de la razón de verosimilitud, en donde, los modelos del hablante dubitado, indubitado y la referencia corresponden a GMMs y adicionalmente contiene distintas librerías que son usadas tanto para la verificación como la identificación y reconocimiento del hablante que se encuentran en el estado del arte.

1.3. OBJETIVOS GENERALES

Implementar y evaluar un sistema de verificación del hablante basado en parámetros de tiempo corto y modelos de mezclas Gaussianas bajo el paradigma de la razón de verosimilitud.

1.4. OBJETIVOS ESPECÍFICOS

- Implementación del sistema de estimación, extracción de características y modelado para los hablantes y la población de referencia.

- Calibración del sistema de verificación del hablante bajo el paradigma de la razón de verosimilitud.
- Evaluar el sistema de verificación del hablante ante condiciones de cambio en el canal y ruido.

1.5. ALCANCES

Con la realización del siguiente proyecto de grado se busca lograr la implementación de un sistema de verificación automática del hablante con un desempeño comparable a sistemas desarrollados recientemente. Para ello se plantea utilizar el motor de entrenamiento ALIZE, el cual cuenta con librerías previamente utilizadas en este tipo de tareas [29]. En consecuencia, el desempeño depende de buena parte de la eficiencia del mismo.

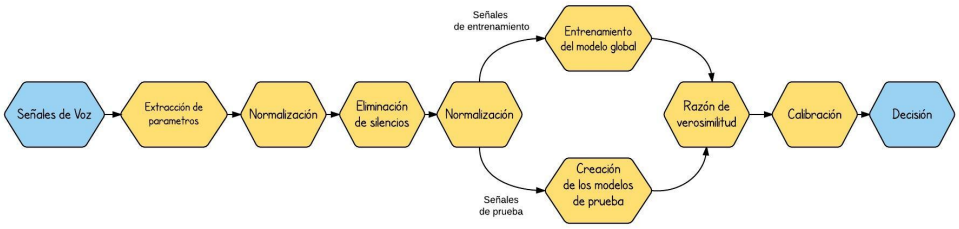
En principio, la evaluación se podría realizar sobre bases de datos de señales de voz disponibles en internet, como *The MIT Mobile Device Speaker Verification Corpus* (http://groups.csail.mit.edu/sls/mdsvc).

CAPÍTULO 2

MÉTODO

Dentro del proceso de verificación del hablante se distinguen las etapas de extracción de características, entrenamiento de los modelos y finalmente comparación de los mismos. La extracción de características tiene por fin obtener una representación adecuada del comportamiento de la señal perteneciente a los hablantes. Luego, el comportamiento de las características es sintetizado y representando mediante modelos probabilísticos. Se requieren tres modelos: el modelo acústico de hablante conocido (voz indubitada), modelo de la población de referencia (representa la entidad de nombre "los demás hablantes") y el modelo de voz procedente de hablante desconocido (voz dubitada).

Figura 2.1: Diagrama del método para el desarrollo de un sistema de verificación del hablante.



Finalmente, se requiere comparar los modelos mediante el uso de alguna medida de distancia en particular. Cada modelo, de los tres en total que se requieren, está en capacidad de generar valores de probabilidad para observaciones particulares.

Con el fin de entregar un veredicto, estos valores son usados para el cálculo de la razón de verosimilitud, la cual podría llegar a verse a modo de una distancia entre modelos. Si el modelo de la señal bajo análisis (voz cuestionada) está más cerca al

modelo de la voz cuyo hablante es conocido, que al modelo de referencia, entonces la evidencia estaría inculpando a la persona cuestionada.

2.1. EXTRACCIÓN DE CARACTERÍSTICAS

Durante el proceso de extracción de características se obtienen aquellos parámetros, de cantidad reducida, que representan el comportamiento de la señal de voz. Sin embargo, una vez se adquiere la señal de voz se tienen problemas que modifican la señal tales como el ruido, sea este producido por el ambiente o por el canal. En tal sentido, la extracción de las características relevantes de la señal se convierte en una tarea realmente importante para un sistema de reconocimiento y verificación de hablantes.

Las características comúnmente utilizadas en sistemas de reconocimiento del hablante son los parámetros MFCC (*Mel-frequency cepstral coefficients*). Estos coeficientes están relacionados con la envolvente espectral, la cual entrega información acerca de la forma del tracto vocal y ha mostrado ser de bastante utilidad en propósitos de verificación de hablantes. Sin embargo, es importante aclarar que no solo los MFCC entregan información de la forma del tracto vocal, están además los coeficientes PLP (*Perceptual Linear Prediction*), LPC (*Linear Predictive Coding*), entre otros.

Para el cálculo de los MFCC, el primer paso consiste en tomar la onda de entrada y someterla a un banco de filtros de ancho de banda delimitados que generan valores de energía asociados a cada rango de frecuencia del filtro, lo cual genera una aproximación del contenido espectral de la señal. Estos filtros, de tipo pasa-banda, están traslapados y distribuidos a lo largo del rango de frecuencias de la señal de voz. Las frecuencias centrales de estros filtros son determinadas de tal manera que todos los filtros compartan el mismo ancho de banda, pero en la escala MEL. La técnica de coeficientes cepstrales en la escala de Mel es un método ampliamente utilizado para obtener los vectores característicos de la señal de voz [34]. Dicha técnica se inspira en el funcionamiento de la cóclea o caracol durante el proceso de la audición humana. En general, la cóclea no puede discernir la diferencia entre dos frecuencias cercanas; cuyo fenómeno se hace más pronunciado a medida que la frecuencia se hace mayor generando un comportamiento aproximadamente exponencial. Por lo tanto se hace uso de la escala de Mel, la cual se aproxima el comportamiento del oído humano.

La conversión de frecuencias de una escala lineal a la escala Mel esta dada por [24]:

$$M(f) = 1125 \ln(1 + f/700) \tag{2.1}$$

Para el calculo de los MFCC se aplican los siguientes pasos: 1) Se calcula la densidad espectral de potencia; 2) Se aplica, en el dominio de la frecuencia, un banco de R filtros al espectro de potencia y se obtiene la energía proveniente de cada filtro; 3) se toma el logaritmo de las energías del banco de filtros para formar un vector de longitud R; 4) se aplica una transformación discreta del coseno al vector anterior con el fin de decorrelacionar las energías ya que los filtros se traslapan entre si [34]. De otra parte, se ha observado un mayor rendimiento en los sistemas de reconocimiento de voz si se calculan no solo los coeficientes MFCC sino también sus primera y segunda derivadas. Con ello se busca además obtener información acerca de la dinámica de la voz de la persona en cuestión [25]. En particular, en el presente trabajo se calculan además los componentes de velocidad y aceleración mediante el uso de los coeficientes Δ-MFCC y $\Delta\Delta$-MFCC [42]. Junto a los coeficientes MFCC, Δ-MFCC y $\Delta\Delta$-MFCC se calcula además el valor de la energía de la señal. Estos vectores de parámetros son normalizados para obtener media cero y varianza unitaria.

En muestras de audio es común encontrar silencios, los cuales corresponden a información generalmente irrelevante. Por lo tanto, la mayoría de sistemas incluyen un detector de energía que tiene como finalidad eliminar los silencios, el cual esta basado en los valores de energía a través del tiempo. Para seleccionar los segmentos correspondientes a la voz (no silencios), se usa una aproximación basado en el valor del *log* de la energía [5]:

1. Los valores normalizados del *log* de la energía son modelados mediante tres componentes Gaussianas.

2. Aquellos valores de mayor energía son tomados para pasos posteriores mientras que los de menor energía se descartan.

Luego, las características son nuevamente normalizadas por medio de la media y varianza de cada vector de características.

De otra parte, para mejorar el desempeño del sistema de reconocimiento se emplean técnicas de transformación sobre las características ya obtenidas. Entre estas técnicas se encuentran el JFA (*Joint Factor Analysis*), FA (*Factor Analysis*) y PCA (*Principal Component Analysis*) entre otras. En general, estos métodos realizan

transformaciones de tipo lineal sobre el espacio de entrada de características de voz con el fin de obtener representaciones que cumplan con criterios adecuados para ser usados en la representación de las distintas clases: modelo de referencia, modelo del hablante y modelo de voz dubitada. Uno de los criterios más útiles para este tipo de tareas es aquel en el que se busca un espacio de salida en el cual las nuevas características están no correlacionadas entre sí. Es decir, cada característica de salida representa una porción de información contenida en la señal de voz. En el presente trabajo no se utilizan este tipo de transformaciones.

2.2. MODELADO ACÚSTICO DE LA SEÑAL

Para el modelado de una señal acústica se emplean varios métodos entre los cuales se encuentra el modelo de mezclas gaussianas. En los últimos años este modelo ha tomado gran auge en los sistemas de reconocimiento del hablante ya que explota las características espectrales de la voz para discriminar a los hablantes.

En los sistemas de reconocimiento basados en GMM se entrena primero un modelo universal (UBM, *Universal Backgroud Model*) que servirá como referencia para la posterior comparación. Este modelo corresponde a una distribución independiente del hablante, es decir modela los patrones de la población de referencia. El entrenamiento se realiza en base a información generada por una cantidad de hablantes de suficiente variabilidad en sus condiciones acústicas respecto a la población de referencia que se quiere representar.

Además con el fin de calcular el valor de *Likelihood Ratio*(se describe en la sección 2.3), se requiere la estimación de tres modelos probabilísticos: modelo de la voz dubitada (hablante por verificar); modelos de la voz indubitada (hablante conocido); y, el modelo de referencia (UBM). El elemento UBM es en esencia un gran modelo de mezclas Gaussianas (GMMs) que representa las propiedades de la voz de la población que se utiliza a modo de referencia y es ampliamente usado en sistemas de verificación del hablante de hoy en día [21]. En los sistemas de reconocimiento basados en GMM (*Gaussian mixture model*) se entrena primero un medio universal (UBM) que servirá como referencia para la posterior comparación. Para el entrenamiento de este modelo global se emplea el criterio ML (*Maximum Likelihood*) junto al algoritmo *Expectation Maximization* (EM). Este último consiste en realizar iteraciones con el fin de encontrar los parámetros del modelo de referencia planteado.

Este proceso se realiza maximizando el resultado de la probabilidad calculada por el modelo dados los parámetros actuales denotados por θ. El algoritmo encuentra θ, el cual es el conjunto de parámetros optimizados por medio del siguiente criterio:

$$\theta = \arg \max_{\Theta} E(p(z_t, Y/\Theta)/z_t, \theta) \tag{2.2}$$

El algoritmo permite encontrar el *maximun likelihood* (ML) o la probabilidad *maximum a posteriori* estimando los datos bajo una hipótesis. El ML para mezclas Gaussianas resulta de tres ecuaciones que generan los parámetros de la mezcla [17],

$$\theta = (\pi_1, ...\pi_n, \mu_1, ..., \mu_n, \sigma_1, ..., \sigma_n) \tag{2.3}$$

$$\pi_i^{'} = \frac{1}{n} \sum_{j}^{n} p(i|z_j, \theta) \tag{2.4}$$

$$\mu_i^{'} = \frac{\sum_j^n z_j p(i|z_j, \theta)}{\sum_j^n p(i|z_j, \theta)} \tag{2.5}$$

$$\sigma_i^{'2} = \frac{\sum_j^n (z_j - \mu_i^{'})(z_j - \mu_i^{'})^T p(i|z_j, \theta)}{\sum_j^n p(i|z_j, \theta)} \tag{2.6}$$

Donde $\pi_i^{'}$ es el peso para la i-ésima distribución y $p(i|z_j, \theta)$ es el likelihood que genera z_j para la distribución i. Posteriormente se adaptan los modelos de cada hablante por medio del criterio MAP (*Maximum a posteriori*) basados en el modelo global obtenido [17].

Una función de probabilidad en particular, que representa el comportamiento estadístico de los datos $\boldsymbol{z}_t$, se puede aproximar mediante la suma ponderada de J funciones $P(\boldsymbol{z}_t; \boldsymbol{\theta}^j)$, de la forma [6],

$$P(\boldsymbol{z}_t \cdot) = \sum_{j}^{J} \pi^j P(\boldsymbol{z}_t; \boldsymbol{\theta}^j) \tag{2.7}$$

donde $\boldsymbol{\theta}^j$ es el vector de parámetros que describe $P(\boldsymbol{z}_t; \boldsymbol{\theta}^j)$. Los pesos π^j en (2.7) son valores escalares normalizados que cumplen $\sum_{j=1}^{J} \pi^j = 1$ y $\pi^j > 0$. Esto asegura que la mezcla descrita por (2.7) es una verdadera función de densidad de probabilidad articulo [45].

En el modelo dado por la ecuación (2.7), $P(\boldsymbol{z}_t; \boldsymbol{\theta}^j)$ corresponde a la distribución

Gaussiana multivariada dada por la ecuación (2.8),

$$P(\boldsymbol{z}_t;\boldsymbol{\theta}^j) = \mathcal{N}(\boldsymbol{z}_t;\boldsymbol{\mu}_z^j,\boldsymbol{\Sigma}_z^j) = \frac{1}{(2\pi)^{d/2}} \mid \boldsymbol{\Sigma}^j \mid^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\boldsymbol{z}_t-\boldsymbol{\mu}^j)^\intercal(\boldsymbol{\Sigma}^j)^{-1}(\boldsymbol{z}_t-\boldsymbol{\mu}^j)\right) \quad (2.8)$$

En este caso el conjunto de parámetros $\boldsymbol{\theta}^j$ en (2.7) corresponde a la media y la covarianza, por tanto $\boldsymbol{\theta}^j = \{\boldsymbol{\mu}_z^j, \boldsymbol{\Sigma}_z^j\}$.

Si se usa una cantidad suficiente de Gaussianas, además del ajuste adecuado de las medias, las covarianzas y los valores de los coeficientes de la combinación lineal, casi cualquier función de densidad de tipo continuo puede ser aproximada [6]. La función de densidad es parametrizada por los pesos, los vectores de medias y las matrices de covarianza de las funciones Gaussianas . Estos parámetros se representan por,

$$\boldsymbol{\Theta} = \left\{\pi^j, \boldsymbol{\mu}_z^j, \boldsymbol{\Sigma}_z^j\right\}, \quad j = 1,\ldots,J \quad (2.9)$$

Una vez se establece la estructura modelo, lo que resta es determinar los parámetros del mismo modelo Θ; para lo cual se aplican procesos de optimización.

Con el fin de crear un modelo global de referencia, con hablantes para tal propósito, se utilizan funciones relacionadas con las librerías ALIZE y LIA_RAL, para los cuales es necesario generar una lista de archivos que contienen los parámetros asociados a cada pronunciación para cada hablante. El modelo de mezclas Gaussianas es estimado usando el algoritmo EM (*Expectation Maximization*).

En el estado del arte de los métodos de verificación del hablante predomina el enfoque hacia los efectos que produce la variabilidad del hablante y derivado de este se presenta el problema de la variabilidad entre sesiones, con el fin de afrontar este efecto se emplea la técnica JFA (*Join Factor Analysis*) la cual consta de generar un modelo de la variabilidad de la sesión a través de la técnica llamada *Eigenchannel MAP*, y además integrándolo con modelos estándares de la variabilidad del hablante por medio de las técnicas *Classical MAP* y *Eigenvoice MAP*, y con esto se genera un modelo tanto del hablante como de la variabilidad entre sesiones que permitirá tener estas características en cuenta en el momento de tomar la decisión y de esta manera generar un resultado con mayor confiabilidad.

2.3. EVALUACIÓN

Obtenidos los modelos GMM del hablante y los modelos GMM-UBM se calcula la razón de verosimilitud entre los modelos. La razón de verosimilitud o *likelihood ratio*, es una medida que compara las diferencias y similitudes entre las muestras conocidas y las voces que se cuestionan, tomando la razón entre la probabilidad que las dos muestras sean del mismo origen y la probabilidad que sean de distintas fuentes, cuantificando de esta manera la diferencia entre las dos voces que finalmente será un valor que permita según su magnitud concluir si el hablante es o no, quien dice ser.

Razón de verosimilitud: La razón de verosimilitud (LR) se define a partir de la aplicación del teorema de Bayes. Primero que todo se establecen dos hipótesis: 1) H_0 : el sospechoso es la fuente de la grabación analizada, y 2) H_1 : alguna otra persona es la fuente de la voz analizada. E corresponde a la evidencia. Para tal caso, la expresión que relaciona las hipótesis con la evidencia es la siguiente [19]:

$$\underbrace{\frac{Pr(H_0/E)}{Pr(H_1/E)}}_{\text{a posteriori}} = \underbrace{\frac{Pr(E/H_0)}{Pr(E/H_1)}}_{LR} \underbrace{\frac{Pr(H_0)}{Pr(H_1)}}_{\text{a priori}} \tag{2.10}$$

donde $Pr(A/B)$ denota la probabilidad condicional,siendo A y B cualquiera de las dos hipótesis H_0 o H_1 o la evidencia E . La probabilidad a priori representa la creencia antes de realizarse el experimento, la cual podría ser descrita por el juez, mientras que la razón de verosimilitud es una medida generada por la evidencia. El LR está definido por,

$$LR = \frac{Pr(E/H_0)}{Pr(E/H_1)} \tag{2.11}$$

Un elemento aún más interesante de la expresión (2.11) es que el método en conjunto no es absolutista, es decir, los resultados de la estimación de la LR pueden ser incorporados junto con otras pruebas provenientes de otras fuentes, lo cual es lo típico en las diligencias judiciales. De esa forma, se entrega la posibilidad de dar a la audiencia un resultado aún más confiable en los veredictos, los cuales por ley y simple sentido común solo un juez puede proferir, no una máquina.

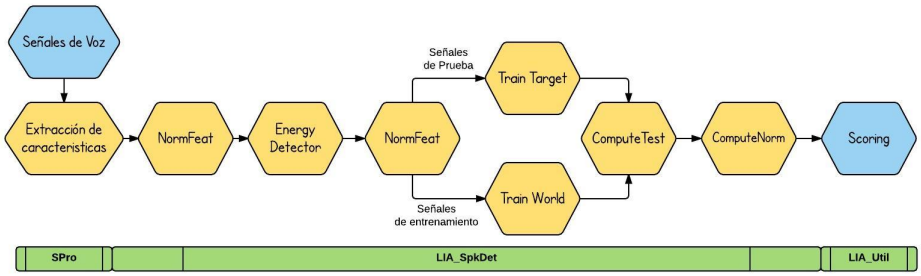
La expresión LR involucra el cálculo de funciones de densidad de probabilidad a partir de los datos proveídos por la evidencia. Existen varios métodos que permiten la estimación de funciones de densidad de probabilidad [15], dentro de los cuales se están los modelos de mezclas Gaussianas, el cual ha mostrado ser adecuado para

el modelado de funciones de densidad de probabilidad de una mediana cantidad de variables con un costo computacional moderado [43].

Para evaluar el desempeño del sistema se emplean las curvas DET (*Detection Error Tradeoff*) y el EER (*Equal Error Rate*), las curvas DET permiten visualizar el desempeño del sistema en base a la relación que se presenta entre los dos tipos de error que se pueden generar: Falsas aceptaciones y falsos rechazos. Los ejes de las curvas DET son representados usando frecuentemente una escala logarítmica [31]. El EER es una de las medidas más comunes en sistemas biométricos reales para valorar el desempeño de un sistema de verificación del hablante y corresponde al punto de las curvas DET en donde el porcentaje de falsos rechazos y falsas aceptaciones es igual. Aunque, LIA_RAL ofrece funciones para la estimación del valor LR, se requieren rutinas adicionales para calcular y graficar las curvas DET.

2.4. ALIZE Y LIA_RAL

Figura 2.2: Diagrama del método para el desarrollo de un sistema de verificación del hablante con ALIZE y LIA_RAL.



En el presente trabajo se desea emplear el paquete ALIZE/LIA_RAL, una plataforma de código abierto escrita en C++, posicionada como una herramienta en el estado del arte para la tarea de reconocimiento del hablante. Esto con el fin de implementar un sistema GMM-UBM y además utilizarse en la prueba de nuevos métodos [29].

El paquete ALIZE en conjunto con LIA_RAL contienen aquellas aplicaciones de bajo nivel necesarias y que son usadas para el desarrollo de un sistema de verificación del hablante. En particular, LIA_RAL hace uso de las librerías contenidas en ALIZE. A continuación, se dará una breve descripción de las herramientas empleadas para desarrollar el método mostrado en el diagrama 2.4.

Extracción de características: Para las señales de voz almacenadas en archivos de audio es necesario la extracción de características, para éste fin ALIZE no contiene alguna función que lo haga, por lo tanto la extracción de parámetros es realizada por *sfbcep* un comando de SPRO4, dónde la parametrización del habla se hace mediante MFCC, las demás opciones que se pueden utilizar se encuentra documentadas en [20].

Eliminación de Silencios: Una vez son calculados los parámetros de cada señal de voz, el siguiente paso es encontrar los vectores adecuados o útiles para nuestro trabajo, una manera es determinando las tramas que corresponden a segmentos hablados de la señal y las que corresponden a silencios, de acuerdo al nivel de energía que presente esa trama. A partir de esto, se seleccionan los segmentos que presente la más alta energía con la herramienta *EnergyDetector* de LIA_RAL (De aquí en adelante todos los ejecutables pertenecen al paquete de LIA_RAL), que entrega un archivo con las etiquetas *speech* y *non-speech*, de acuerdo al nivel de energía de cada trama de la señal de voz. Por otra parte, la normalización es realizada antes y después de la detección de silencios con *NormFeat*, que usa la normalización media cero y varianza uno, a cada archivo de parámetros. En particular, se calculan 19 coeficientes MFCC sobre ventanas de 20 ms de ancho del tipo Hamming. Adicionalmente, a partir de los mismos, se calculan los coeficientes Δ-MFCC y $\Delta\Delta$-MFCC además del valor de la energía de la señal, para un total de 60 parámetros de representación. Adicionalmente, las secuencias de parámetros son normalizados [17].

Entrenamiento: Para llevar a cabo la verificación de hablante normalmente se necesitan dos partes. Primero entrenar un modelo global de referencia (UBM), y luego a partir de él se crean los modelos GMM de cada hablante.

Entrenamiento del UBM: El entrenamiento del UBM, se lleva acabo con la herramienta *TrainWorld*, dónde la entrada es la lista de parámetros de todos los segmentos de las señales de voz y la salida es un solo GMM, que representa toda la población. Inicialmente las distribuciones se establecen con la varianza y la media global, luego el modelo es entrenado iterativamente con todos los vectores de pará-

metros usando el algoritmo EM con un criterio de ML [17].

Extracción de los modelos de cada Hablante: Con el fin de realizar la tarea de verificar a cada hablante, se necesita extraer un modelo GMM para cada hablante. Por medio de los vectores de parámetros del hablante y el UBM, el modelo se extrae a través de la herramienta *TrainTarget*, que emplea la estimación máxima a posteriori (MAP).

Pruebas: En el momento de probar un segmento desconocido, un modelo objetivo de un hablante (conocido) y un UBM, la herramienta *ComputeTest*, calcula un puntaje de ese segmento respecto al modelo objetivo con el cual se hace la comparación. Con éste puntaje, el sistema refleja la probabilidad de que el segmento de prueba sea o no el hablante objetivo (modelo con el cual se probó). Para el siguiente paso se necesitan diferentes tipos de puntajes, para esto se desean realizar dos pruebas, en el caso de T-norm: a) Puntajes objetivos, dónde se compara los segmentos de prueba con los modelos objetivos y b) Puntajes de Impostores, dónde se compara los segmentos de prueba con los modelos de los impostores.

Calibración: El siguiente paso una vez se tiene los resultados, es aplicar una normalización a estos puntajes para así tratar de alguna manera con la variabilidad de los puntajes, LIA_RAL posee una herramienta llamada *ComputeNorm*, para realizar ésta tarea, cuenta con diferentes normalizaciones basadas en *T-norm*, y *Z-norm*. Luego de encontrar la razón de verosimilitud es recomendable realizar una normalización a los valores obtenidos ya que de esta manera se puede lograr una mejora en el desempeño del sistema. Para esto existen varias técnicas, de las cuales las mas utilizadas son *Z-norm*, *T-norm* y *H-norm*. En éste trabajo, se utilizó *T-norm*, y para esto es necesario ingresarle a este ejecutable los puntajes de los impostores y los que se deseen normalizar (éstos se calculan en el paso anterior, con el ejecutable *ComputeTest*). Para T-norm (*Test Normalization*), los *scores* o puntajes obtenidos de los modelos de los hablantes son normalizados usando las muestras de los impostores. Se realiza una prueba en la que se someten las muestras de los hablantes a un test, comparando los modelos de los impostores y los demás hablantes. Los resultados de esta prueba son usados para encontrar los dos parámetros necesarios para realizar la normalización de los *scores* ($\log_{\Lambda_M}$); es decir, el valor medio μ_1 y la

desviación estándar σ_1. Se normaliza de la siguiente manera:

$$\log_{\Lambda_M,T} = \frac{\log_{\Lambda_M} - \mu_1}{\sigma_1} \tag{2.12}$$

Decisión: El paso final es la toma de decisiones, esto se lleva a cabo con la herramienta *Scoring*, dónde se ingresan los puntajes normalizados y un umbral de decisión, de ésta manera si el puntaje supera el umbral, se dice que el segmento de prueba pertenece al hablante objetivo y si se encuentra por debajo, el segmento de prueba no corresponde al ese hablante, así que finalmente se tienen dos posibles resultados t si es verdadero (*True*) o f si es falso (*False*).

2.5. BASE DE DATOS DE VOZ

En el presente, se usan dos bases de datos:

MIT Device Speaker Verification Corpus: Con el fin de reproducir un experimento en el cual se logre evaluar el sistema antes posibles cambios de ambiente y micrófono se adquirió la base de datos *MIT-DSVC, Device Speaker Verification Corpus* [44], la cual está adaptada para ser aplicada en sistemas de verificación del hablante dependiente del texto. La base de datos se divide en dos grupos: Un grupo de hablantes verdaderos y un grupo de impostores. La recolección de los datos de voz para el grupo de hablantes verdaderos se realizó en dos sesiones con el fin de asignar una sesión para entrenamiento del sistema y otra para la evaluación. Para cada sesión, la recolección de datos tuvo lugar con muy bajo nivel de ruido, un pasillo con un nivel de ruido intermedio y una intersección donde se presenta tráfico vehicular y peatonal que genera un alto nivel de ruido. En general, durante cada sesión se tomaron 54 frases por usuario, las cuales completarían 5,184 muestras de los hablantes verdaderos (2,592 por sesión), y 2700 frases de impostores. EL grupo de hablantes verdaderos es de 48 personas, de las cuales 22 son mujeres y 26 hombres y para el grupo de impostores se tiene un total de 40 hablantes, 17 mujeres y 23 hombres.

Base de datos de adultos hombres: También se decidió crear una base de datos en la cual se reunieron 30 hablantes masculinos que se encuentran entre los 18 y los 50 años de edad. Esto con el fin de emplear esta base de datos en las pruebas que se muestran en el parágrafo dos de la sección 3.2. Para esto, la recolección de

datos se realizó en diversos ambientes y en 3 sesiones diferentes para cada hablante. La duración de cada sesión consta de 3 minutos aproximadamente, en la que los hablantes leen tres hojas diferentes de la sección deportiva de un periódico local, en español.Todas las grabaciones se llevaron a cabo por medio del micrófono de un dispositivo móvil de referencia Huawei G630-U251, con frecuencia de muestreo de 8 KHz y en formato de audio wav. Para entrenar el modelo de referencia se utilizaron las 3 sesiones de 15 hablantes, mientras que para los 15 hablantes restantes se tomó 1 sesión para el entrenamiento de cada modelo y las 2 sesiones restantes para realizar las respectivas pruebas.

CAPÍTULO 3

RESULTADOS

El presente capítulo se encuentra principalmente organizado en dos secciones. En 3.1 se presenta y se describe el desarrollo de una Interfaz gráfica que se realizó en este trabajo para facilitar la ejecución de los experimentos. Luego en 3.2 se presentan los resultados obtenidos de acuerdo a los experimentos que se realizaron con las dos distintas bases de datos.

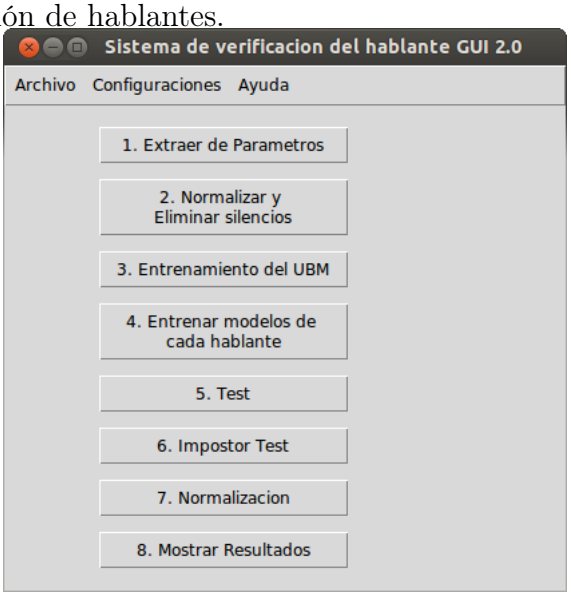
3.1. INTERFAZ GRÁFICA - GUI

Con el fin de facilitar el desarrollo de sistemas de verificación de hablantes a personas que no estén familiarizadas con ALIZE, se desarrolla una interfaz gráfica (Graphical User Interface, GUI). A continuación se presenta la interfaz realizada.

Desarrollo de la Interfaz: La Interfaz gráfica fue desarrollada usando una librería de *Python* llamada *Tkinter*, la cual es típicamente utilizada para este tipo de tareas. En particular, en el presente proyecto se utiliza la versión 2.7.6 de *Python*. Lo que se busca con esta GUI, es realizar de manera más fluida y un poco más visual el desarrollo de sistemas de verificación de hablantes así como la realización de experimentos, como los que se presentan más adelante en la sección 3.2. La GUI permite variar los parámetros de configuración del sistema tales como el número de gaussianas del modelo de referencia, entre otros. Por lo tanto, se plantea que en una ventana principal se tengan todos los pasos necesarios del método.

Ventana Principal: En la figura 3.1, se muestra la ventana principal de la interfaz gráfica, que se ha desarrollado en el proyecto y que sirve para la realización de experimentos y sistemas de verificación de hablante.

Figura 3.1: Ventana Principal de la Interfaz gráfica realizada para el desarrollo de sistemas de verificación de hablantes.



Como se puede observar en la ventana principal se tienen 8 pasos para desarrollar el sistema y realizar los experimentos. Por otra parte se tiene una barra de tareas que cuenta principalmente con tres menús: Archivo, Configuraciones y Ayuda. En *Archivo* se puede seleccionar si se desea un nuevo o antiguo proyecto, de allí se despliegan dos opciones para escoger alguna de las dos bases de datos con las que se trabajo el proyecto enseguida se abrirá un cuadro de texto para seleccionar la dirección (cabe decir que si selecciona nuevo proyecto, la aplicación automáticamente crea, organiza, copia y pega los archivos necesarios), por otra parte se encuentra una opción para salir inmediatamente de la aplicación. Mediante la opción *configuraciones* se permite leer o editar los archivos de configuración de los ejecutables empleados por ALIZE. Finalmente en *Ayuda*, se encuentra una opción para abrir el léeme del *script* desarrollado para correr los ejecutables de ALIZE, además contiene un *acerca de ...* que abre un cuadro de texto dando una breve explicación de la versión de la interfaz y los desarrolladores.

3.2. EXPERIMENTOS

En los parágrafos que se presentan a continuación, se organizan los resultados y se presentan de acuerdo al experimento realizado.

Sistema En Diversos Ambientes y Población de referencia del UBM. En el primer experimento se tomo la base de datos proporcionada por el MIT, este se

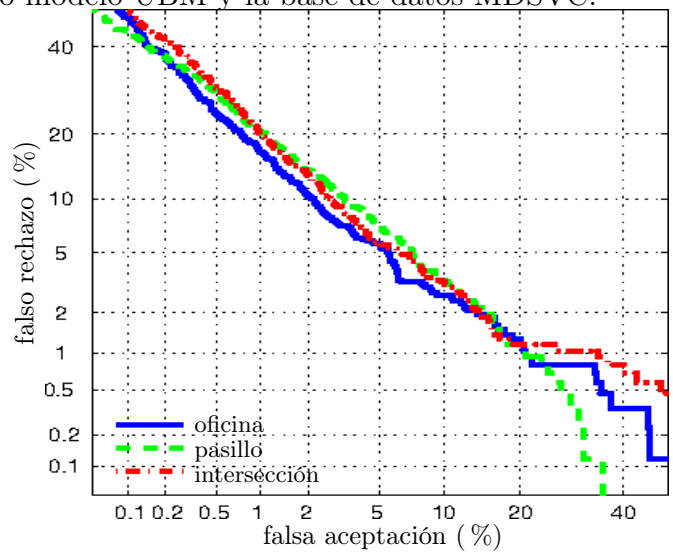
realizo con el fin de evaluar el sistema ante diversos ambientes y diferentes micrófonos mediante los cuales se capturaron las señales. Las señales son representadas mediante 19 coeficientes cepstrales y el valor de la energía, además de la primera y segunda derivadas de este vector de parámetros. Es decir, se tiene un total de 60 parámetros. Luego se realiza el proceso de normalización de características. Para la eliminación de silencios se usan 8 iteraciones del algoritmo EM para estimar la distribución de la energía formada por una mezcla de 3 distribuciones gaussianas.

De otra parte, para el entrenamiento del modelo global (modelo de referencia) se usaron 80 distribuciones Gaussianas en la mezcla y un total de 15 iteraciones EM. El modelo del hablante en cuestión está formado por 10 gaussianas (valor sugerido por ALIZE/Mistral [30]). Para la adaptación de los modelos de los hablantes del conjunto de prueba se utilizó el método MAP, el modelo de cada hablante se adapta a partir del modelo de referencia de manera que los resultados del método son producto de la combinación de los modelos de referencia y el del hablante. El parámetro de ponderación α de la combinación lineal está controlado por el parámetro $MAPRegFactorMean$ dentro del archivo de configuración correspondiente. Se selecciona 14 como el valor para $MAPRegFactorMean$ (valor sugerido por [5]). Para la configuración del sistema en su totalidad se requieren parámetros adicionales. Los valores usados se pueden observar en los anexos B. Finalmente se le aplica la técnica *T-norm* con el fin de normalizarla los *scores*.

En las figuras (3.2, 3.2, 3.2) se muestran los resultados cuando se usa un único modelo de referencia en el que se combinan tanto hombres como mujeres con diferentes tipos de ambiente de ruido (oficina, pasillo e intersección). A partir de las figuras (3.2, 3.2) se observa que existe menor variabilidad respecto al tipo de ambiente en el caso de las mujeres respecto al resultados de los hombres. Para estos últimos, aunque ninguno obtiene un EER por arriba de 7 % se aprecia una variabilidad aproximada de 2 puntos porcentuales de EER. Adicionalmente, se observa que en el caso de los hombres el valor del EER es inferior. En trabajos anteriores [44] en los que se utiliza la base de datos MDSVC, se obtienen resultados similares al usar voces con varios niveles de ruido.

De otra parte, en las figuras (3.2, 3.2) se presentan los resultados obtenidos de la evaluación usando un modelo independiente para cada género; es decir, UBM para hombres y otro UBM para mujeres. Comparando estos resultados con los mostrados en las figuras (3.2, 3.2) (usando un único UBM) se observa más variabilidad entre los diferentes ambientes cuando es utilizado el mismo modelo global, tanto para

Figura 3.2: Curva DET, sin hacer distinción entre hombres y mujeres, resultado de utilizar un único modelo UBM y la base de datos MDSVC.



Cuadro 3.1: Valores EER resultado de utilizar un único modelo UBM y la base de datos MDSVC.

Género	Ambiente de prueba			Promedio
	Oficina	Pasillo	Intersección	
Masculino	3.63 %	5.56 %	4.51 %	4.57 %
Femenino	5.65 %	6.31 %	6.31 %	6.09 %
Ambos	5.21 %	6.13 %	5.48 %	5.61 %

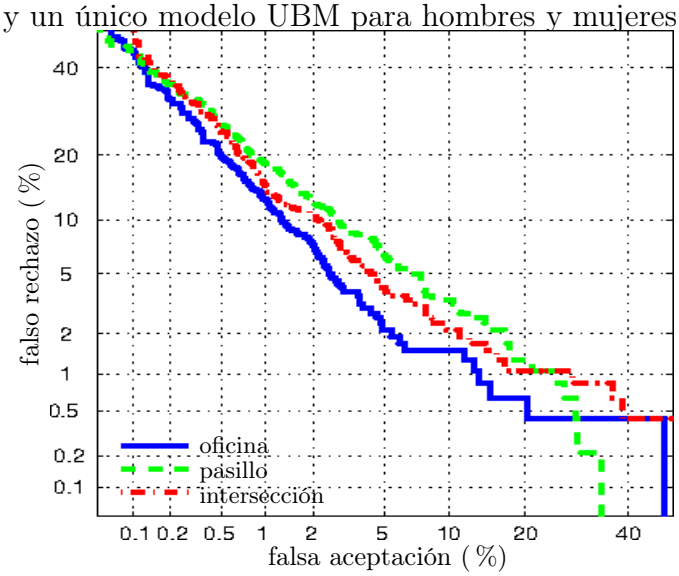
hombres como para mujeres. Adicionalmente, de las figuras (3.2, 3.2, 3.2, 3.2) y de los cuadros 3.2 y 3.2, se puede apreciar que el hecho de utilizar una única población de referencia para hombres y mujeres disminuye el desempeño del sistema de verificación del hablante en aproximadamente 0.06 puntos de diferencia en EER para el caso masculino, y para el caso femenino es de 1.23 puntos de diferencia en EER, que representa aproximadamente una mejora del 22 %.

Estos resultados permiten entrever el buen desempeño que presenta la plataforma ALIZE en el campo de la verificación del hablante. Sin embargo, la evidente

Cuadro 3.2: Valores EER resultado de utilizar un modelo UBM para cada género y la base de datos MDSVC.

Género	Ambiente de prueba			Promedio
	Oficina	Pasillo	Intersección	
Masculino	4.06 %	5.16 %	4.31 %	4.51 %
Femenino	5.05 %	5.87 %	6.31 %	5.74 %

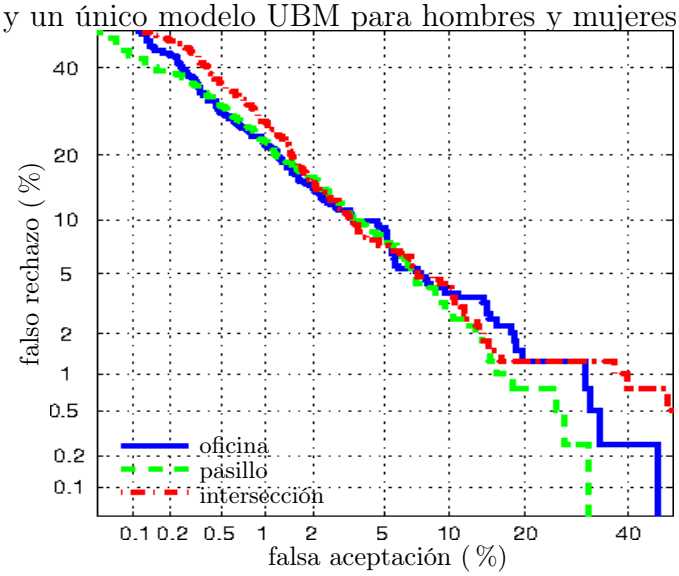
Figura 3.3: Curva DET, de hablantes masculinos, resultado de utilizar la base de datos MDSVC y un único modelo UBM para hombres y mujeres.



poca documentación acerca del uso de las librerías asociadas a la misma dificulta aprovechar de una mejor manera esta plataforma. A pesar de ello, con el desarrollo del sistema presentado en éste trabajo, se pueden apreciar las facilidades proveídas por ALIZE para el desarrollo de sistemas de verificación del hablante. En el presente trabajo, los ejecutables de la librería LIA_RAL, junto con su previa configuración de las librerías de ALIZE, Spro y SPHERE, se invocan mediante el uso de Python. Con ello se busca reducir costos y tiempo de implementación sin sacrificar rapidez en tiempo de procesamiento y de entrenamiento. ALIZE es una herramienta que ha podido llegar a posicionarse en el estado del arte en cuanto a su desempeño. De hecho, los resultados obtenidos en el presente trabajo permiten soportar esta afirmación.

Número de Distribuciones Gaussianas para el entrenamiento del UBM. Por otra parte, en los experimentos que se muestran a continuación se modifica la cantidad de Gaussianas en el modelo de referencia y se evalúa su desempeño. Ello permite evaluar la influencia de este parámetro en el desempeño del sistema. En la gráfica 3.2 se muestran los resultados para el género masculino y en la gráfica 3.2 los resultados obtenidos para el género femenino. Estos resultados se obtuvieron utilizando un UBM para cada género al utilizar la base de datos MDSVC, de lo cual se observa que el número de Gaussianas es un factor determinante en la etapa del modelado . Además en la tabla 3.2 se logra apreciar el desempeño del sistema para cada una de los distintos números de Gaussianas con las que fueron probadas.

Figura 3.4: Curva DET, de hablantes femeninos, resultado de utilizar la base de datos MDSVC y un único modelo UBM para hombres y mujeres.



Por otra parte, en la tabla 3.2, se muestra el desempeño del sistema (EER) ante 14 cantidades diferentes de distribuciones Gaussianas para el entrenamiento del UBM utilizando la base de datos que fue recolectada en este proyecto. Finalmente en las gráficas (3.2,3.2), se observa un comportamiento similar dado que el número de distribuciones Gaussianas es un factor determinante en la etapa de entrenamiento del modelo global de referencia y esto influye finalmente en el desempeño del sistema.

Cuadro 3.3: Valores EER resultado de utilizar un número distinto de Gaussianas al entrenar el UBM, correspondiente a cada género y utilizando la base de datos MDSVC.

Número de Gaussianas	EER Hombres	EER Mujeres
20	8.12 %	11.36 %
30	6.69 %	7.98 %
50	5.20 %	6.40 %
70	4.84 %	5.72 %
80	4.53 %	5.72 %
100	4.13 %	5.47 %
120	4.06 %	4.93 %
140	3.72 %	5.05 %
160	3.35 %	5.13 %
180	3.70 %	5.09 %
200	3.77 %	4.38 %
220	3.85 %	4.71 %
240	3.45 %	4.38 %
1024	3.45 %	4.38 %
2048	4.34 %	5.47 %

Cuadro 3.4: Valores EER resultado de utilizar un número distinto de Gaussianas al entrenar el UBM, correspondiente a cada género, utilizando la base de datos recogida en este proyecto.

Número de Gaussianas	EER
20	2.22 %
30	1.29 %
50	0.70 %
70	0.70 %
80	0.47 %
100	0.59 %
120	0.59 %
140	0.35 %
160	0.35 %
180	0.70 %
200	0.47 %
220	0.35 %
240	0.47 %
1024	0.23 %

Figura 3.5: Curva DET de hablantes masculinos, resultado de utilizar la base de datos MDSVC y crear un modelo de referencia para Hombres.

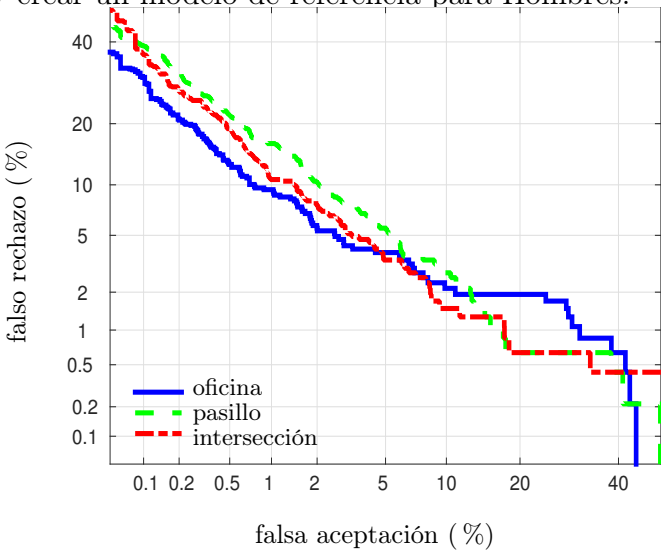


Figura 3.6: Curva DET de hablantes femeninos, resultado de utilizar la base de datos MDSVC y crear un modelo de referencia para Mujeres.

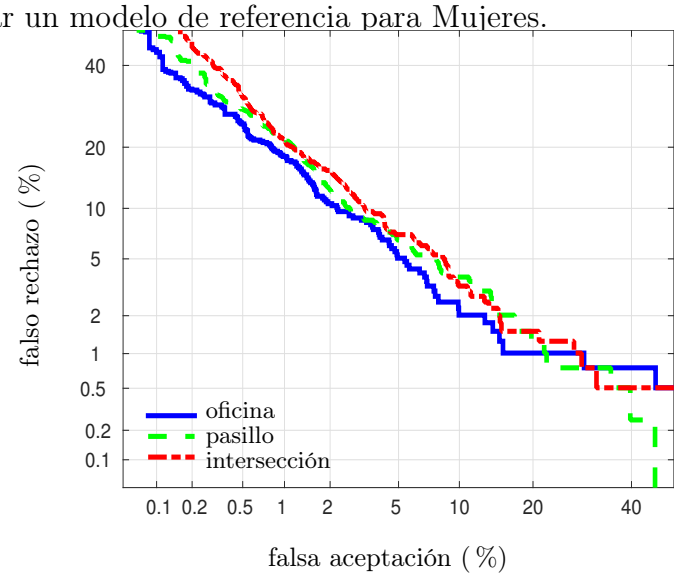


Figura 3.7: Curva DET de hablantes masculinos resultado de variar el número de Gaussianas y utilizar la base de datos MDSVC.

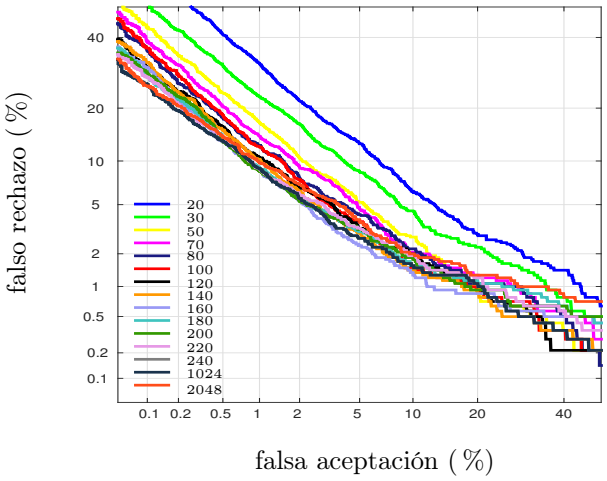


Figura 3.8: Curva DET de hablantes femeninos, resultado de variar el número de Gaussianas y utilizar la base de datos MDSVC.

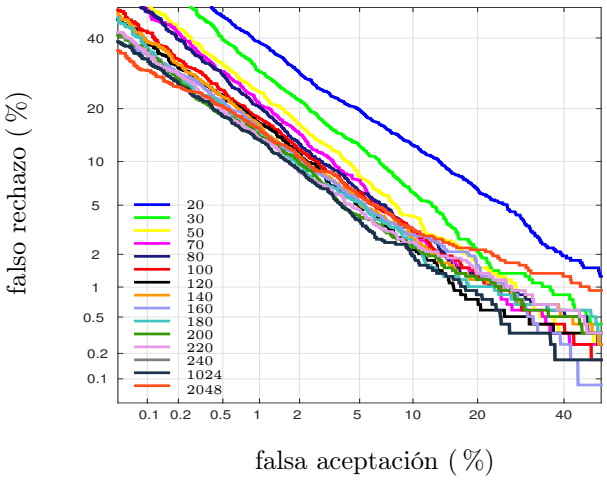


Figura 3.9: Gráfica que evalúa el desempeño del sistema de acuerdo al número de distribuciones Gaussianas y además emplear la base de datos MDSVC para la creación de un UBM para cada género.

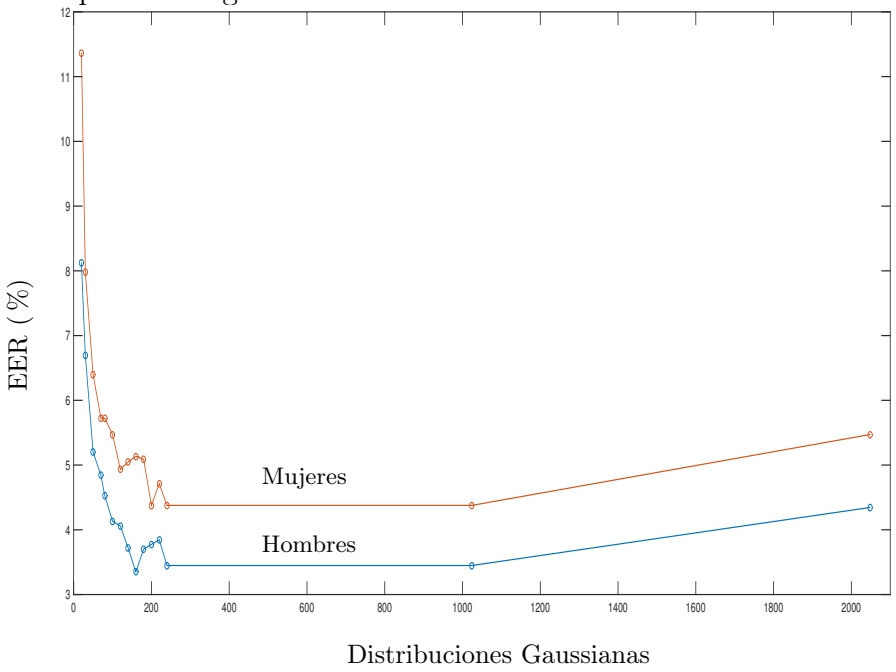
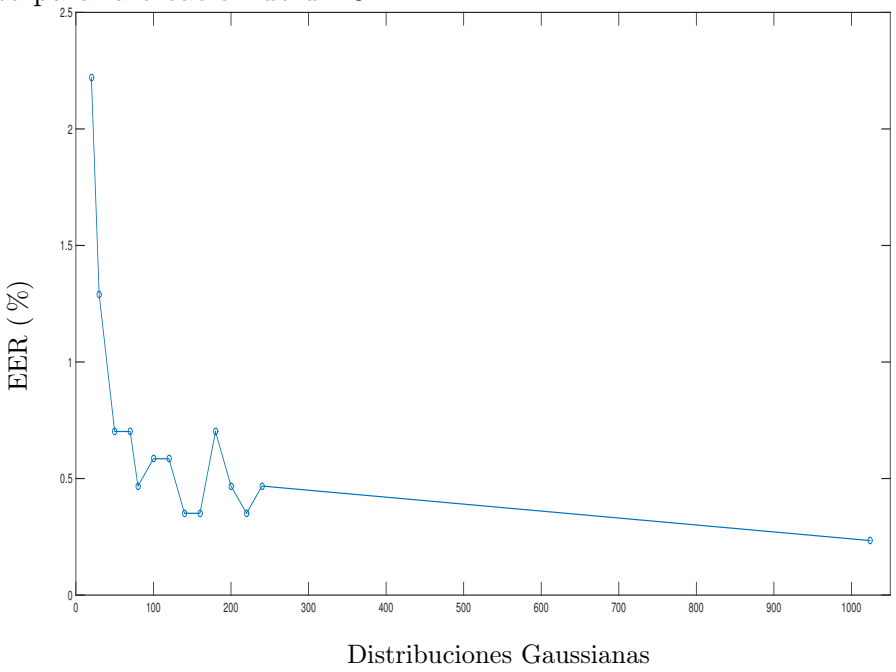


Figura 3.10: Gráfica que evalúa el desempeño del sistema de acuerdo al número de distribuciones Gaussianas y además emplear la base de datos recolectada en este proyecto para la creación de un UBM.



CAPÍTULO 4

CONCLUSIONES

En el presente trabajo se desarrolla un sistema de verificación del hablante usando en parámetros de tiempo corto a modo de representación y modelos de mezclas Gaussianas para el modelado. Esto bajo el paradigma de la razón de verosimilitud y basado en la librería de desarrollo ALIZE. Además se realizó una interfaz gráfica, la cual permitiría efectuar pruebas de verificación de hablantes a personas poco familiarizadas con ALIZE. El sistema se desarrolló sobre el sistema operativo Ubuntu-Linux. Finalmente, el sistema se evalúa bajo diversas condiciones y usando dos bases de datos. La primera corresponde a la base de datos MIT-DSVC; y la segunda, corresponde a una base de datos recolectada por los autores del presente trabajo.

De otra parte, se analizó el efecto del ruido de diversos ambientes en el entrenamiento de los modelos. La diferencia en cuanto al desempeño EER entre los tres ambientes (oficina, pasillo, esquina) es de 2.8, lo cual permite afirmar que el sistema es moderadamente robusto. En particular, el mínimo valor que se obtuvo para oficina 3.63 %; y, el mayor valor se obtuvo para la interseccion de calle 6.31 %.

A modo de trabajo adicional (trabajo no planteado dentro de los objetivos específicos), se evalúa la influencia del entrenamiento del modelo global de referencia (UBM) dependiendo del género. Para ello se realizaron dos experimentos: el primero dónde se entrenó un UBM con hablantes de ambos géneros (tanto masculino como femenino); y, luego se desarrolla otro experimento que se realiza al entrenar un modelo de referencia UBM para cada género. Se concluye una mejoría cercana al 5 %. Los resultados de este experimento fueron presentados en el XX Simposio de Tratamiento de Señales, Imágenes y Visión Artificial (STSIVA-2015).

Adicionalmente, se elaboró otro experimento, en el que se puede mostrar la influencia del numero de distribuciones Gaussianas en el entrenamiento del modelo UBM. De esta manera, se logra observar un rango a partir del cual el sistema no presenta variaciones mayores a 1.4 puntos porcentuales en su desempeño (EER) para la base de datos MIT-DSVC. Para la base de datos de hombres adultos las variaciones no superan los 0.3 puntos porcentuales en EER, siendo recomendable usar un numero de gaussianas por encima de un total de 70.

Finalmente, a modo de trabajo futuro se plantea investigar la influencia del género en sistemas que incluyen la técnica JFA *(Joint Factor Analysis)* y SVM *(Support Vector Machines)*, las cuales han mostrado que mejoran el desempeño de los sistemas de verificación del hablante, además de disminuir el efecto del ruido que se presenta en la captación de la señal. Se plantea además mejorar la interfaz gráfica.

Bibliografía

A. Martin and M. Przybocki. The NIST speaker recognition evaluation series, National Institute of Standards and Technology's. Available from Internet: http://www.nist.gov/speech/tests/sre.

Alegre, F. L. (2007). Application of ANN and HMM to Automatic Speaker Verification. *IEEE Latin America Transactions*, 5.

Alexander, A., Botti, F., Dessimoz, D., and Drygajlo, A. (2004). The effect of mismatched recording conditions on human and automatic speaker recognition in forensic applications. *Forensic Science International*, pages 95–99.

Arana, M. G. I. (2006). *Manual Único de Criminalística*. Fiscalía General de la Nación.

Biosecure (2007 [cited 2015]). Reference system based on speech modality alize/lia_ral [online]. Available from Internet: http://www-clips.imag.fr/geod/User/laurent.besacier/NEW-TPs/TP-Biometrie/tools/CommentsLBInstall/doc.pdf.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer.

Bonastre, J.-F. F., Bimbot, F., Boë, L.-J. J., Campbell, J. P., Reynolds, D. A., and Magrin-Chagnolleau, I. (2003). Person authentication by voice: A need for caution. In *{E}urospeech 2003 - {I}nterspeech 2003. Proceedings of the 8th {E}uropean Conference on Speech Communication and Technology*, pages 33–36.

Campbell, J. P., Shen, W., Campbell, W. M., Schwartz, R., Bonastre, J. F., and Matrouf, D. (2009). Forensic speaker recognition. *IEEE Signal Processing Magazine*, 26:95–103.

Campbell, W. M., Sturim, D. E., and Reynolds, D. A. (2006). Support vector machines using gmm supervectors for speaker verification. *IEEE Signal Processing Letters*, 13(5):308–311.

Castaldo, F., Colibro, D., Dalmasso, E., Laface, P., and Vair, C. (2007). Compensation of nuisance factors for speaker and language recognition. *IEEE Transactions on Audio, Speech & Language Processing*, 15(7):1969–1978.

Comunidad Andina (2014 [cited 2015]). Telefonía móvil en la comunidad andina [online]. Available from Internet: http://estadisticas.comunidadandina.org/eportal/contenidos/2434_8.pdf.

Dehak, N., Kenny, P. J., Dehak, R., Dumouchel, P., and Ouellet, P. (2011). Front-end factor analysis for speaker verification. *IEEE Transactions on Audio, Speech and Language Processing*, 19:788–798.

D'Haro, L. F., de Córdoba, R., Rojo, J. I., Díez, J., Avendaño, D., and Bermudo, J. M. (2014). Low-Cost Speaker and Language Recognition Systems Running on a Raspberry Pi. *IEEE Latin America Transactions*, 12, No 4:9.

Doddington, G. R. (1985). Speaker Recognition- Identifying People. *Proceedings of the IEEE*, 73:1651–1664.

Duda, R., Hart, P., and Stork, D. (2001). *Pattern Classification.* Jhon Wiley and Sons, Inc.

Fauve, B. G. B., Matrouf, D., Scheffer, N., Bonastre, J.-F., and Mason, J. S. D. (2007). State-of-the-art performance in text-independent speaker verification through open-source software. *IEEE Transactions on Audio, Speech & Language Processing*, 15(7):1960–1968.

Gannert, T. (2007). A speaker verification system under the scope: Alize. Master's thesis, KTH Computer Science and Communication, Stockholm, Sweden.

Garimella, S. and Hermansky, H. (2013). Factor analysis of auto-associative neural networks with application in speaker verification. *IEEE Trans. Neural Netw. Learning Syst.*, 24(4):522–528.

Gonzalez-Rodriguez, J., Drygajlo, A., Ramos-Castro, D., Garcia-Gomar, M., and Ortega-Garcia, J. (2006). Robust estimation, interpretation and assessment of likelihood ratios in forensic speaker recognition. *Computer Speech & Language*, 20(2-3):331–355.

Gravier, G. (2004 [cited 2015]). Spro [online]. Available from Internet: http://www.irisa.fr/metiss/guig/spro/spro-4.0.1/spro.html.

Hasan, T. and Hansen, J. (2011). A study on universal background model training in speaker verification. *Audio, Speech, and Language Processing, IEEE Transactions on*, 19(7):1890–1899.

Hasan, T. and Hansen, J. H. L. (2013). Acoustic Factor Analysis for Robust Speaker Verification. *IEEE Transactions on Audio, Speech, and Language Processing*, 21:842–853.

Higgins, A., Bahler, L., and Porter, J. (1991). Speaker verification using randomized phrase prompting. *Digital Signal Processing*, 1(2):89–106.

Huang, X., Acero, A., and Hon, H. (2001). *Spoken Language Processing: A guide to theory, algorithm, and system development.* Prentice Hall.

Ichikawa, O., Fukuda, T., and Nishimura, M. (2010). Dynamic features in the linear-logarithmic hybrid domain for automatic speech recognition in a reverberant environment. *IEEE Journal of Selected Topics in Signal Processing*, 4.

Kenny, P., Boulianne, G., and Dumouchel, P. (2005). Eigenvoice modeling with sparse training data. *IEEE Transactions on Speech and Audio Processing*, 13:345–354.

Kenny, P., Ouellet, P., Dehak, N., Gupta, V., and Dumouchel, P. (2008). A study of interspeaker variability in speaker verification. *IEEE Transactions on Audio, Speech and Language Processing*, 16:980–988.

Kinnunen, T. and Li, H. (2010). An overview of text-independent speaker recognition: From features to supervectors. *Speech Commun.*, 52(1):12–40.

Larcher, A., Bonastre, J.-F., Fauve, B. G., Lee, K.-A., Lévy, C., Li, H., Mason, J. S., and Parfait, J.-Y. (2013). Alize 3.0-open source toolkit for state-of-the-art speaker recognition. In *INTERSPEECH*, pages 2768–2772.

LIA_Team (2011 [cited 2015]). Lia_spkdet package documentation [online]. Available from Internet: http://mistral.univ-avignon.fr/doc/userguide_LIA_SpkDet.002.pdf.

Martin, A., Doddington, G., Kamm, T., Ordowski, M., and Przybocki, M. (1997). The det curve in assessment of detection task performance. Technical report, DTIC Document.

Morrison, G. S. (2010). *Expert Evidence*, chapter Forensic voice comparison. Thomson Reuters.

Naik, J. and Doddington, G. (1987). Evaluation of a high performance speaker verification system for access control. *ICASSP '87. IEEE International Conference on Acoustics, Speech, and Signal Processing*, 12.

Patino-Saucedo, A., Sepulveda-Sepulveda, A., and Gómez-Cajas, D. F. (2015). Phoneme recognition system using articulatory-type information. *TECCIENCIA*, 10(19).

Pawlewski, M. and Jones, J. (2001). Speaker verification. *Biometric Technology Today*, 14:9–11.

Przybocki, M. A., Martin, A. F., and Le, A. N. (2006). NIST speaker recognition evaluation chronicles - Part 2. In *IEEE Odyssey 2006: Workshop on Speaker and Language Recognition*.

Przybocki, M. A., Martin, A. F., and Le, A. N. (2007). NIST speaker recognition evaluations utilizing the mixer corpora2004, 2005, 2006. *IEEE Transactions on Audio, Speech and Language Processing*, 15:1951–1959.

Quatieri, T. F., Dunn, R. B., and Reynolds, D. A. (2000). On the influence of rate, pitch, and spectrum on automatic speaker recognition performance. In *Sixth International Conference on Spoken Language Processing, ICSLP 2000 / INTERSPEECH 2000, Beijing, China, October 16-20, 2000*, pages 491–494.

Reynolds, D. A. (1995). Automatic speaker recognition using gaussian mixture speaker models. *The Lincoln Laboratory Journal*, pages 173–192.

Rose, P. (2002). *Forensic Speaker Identification*. Forensic Science Series. Taylor & Francis.

Rosenberg, A. (1973). Listener performance in speaker verification tasks. *IEEE Trans. Audio Electroacoust*, 21.

Schafer, R. (2008). *Homomorphic Systems and Cepstrum Analysis of Speech*. Springer.

Sepúlveda, A., Guido, R. C., and Castellanos-Dominguez, G. (2013). Estimation of relevant time-frequency features using kendall coefficient for articulator position inference. *Speech Commun.*, 55(1):99–110.

Woo, R., Park, A., and Hazen, T. J. (2006). The MIT Mobile Device Speaker Verification Corpus: Data collection and preliminary experiments. In *Proceedings of Odyssey 2006, The Speaker and Language Recognition Workshop*.

Yannis, S. (1996). *Harmonic plus Noise Models for Speech, Combined with Statistical Models, for Speech and Speaker Modification.* PhD thesis, l'Ecole Nationale Supérieure des Télécommunications.

Zhao, X. and Dong, Y. (2012). Variational bayesian joint factor analysis models for speaker verification. *IEEE Transaction Audio, Speech Language Processing*, 20(3):1032–1042.

APÉNDICES

APÉNDICE A

TUTORIAL DE INSTALACIÓN ALIZE Y HERRAMIENTAS REQUERIDAS

Sistema:

- Linux Ubunut 14.04
- Compilador g++ 3.3.3
- Autotools-dev
- Automake
- Libtool

ALIZE 3.0:

- Se descarga la versión correspondiente de:
  http://mistral.univ-avignon.fr/download.html
- Se crea un directorio para la carpeta descomprimida, se ubica allí y se desarrolla la siguiente secuencia:

```
$ sudo apt-get install subversion
$ svn co http://alize.univ-avignon.fr/svn/ALIZE/trunk/
```

- En la carpeta cambiar el nombre del archivo *'configure.in'* por *'configure.ac'*, para evitar la advertencia. Hacer:

```
$ sudo aclocal
$ sudo automake --add-missing
```

- Para evitar que no instale ./compile, sin embargo se obtiene una advertencia, el cual puede ser ignorado.

```
configure.ac:2: warning: AM_INIT_AUTOMAKE: two- and three-
   arguments forms are deprecated. For more info, see:
configure.ac:2: http://www.gnu.org/software/automake/manual/
   automake.html#Modernize-AM_005fINIT_005fAUTOMAKE-
   invocation
configure.ac:5: installing './compile'c
```

```
$ sudo autoconf
$ ./configure
$ sudo make
$ sudo make install
```

- Si se instala correctamente se debe crear un archivo donde se encuentra la carpeta de ALIZE *(/home/../ALIZE/lib)*, llamado *"libalize_Linux_i686.a"* o *"libalize.a"*. En particular, por ser un sistema de 64 bits se crea el archivo *libalize_Linux_x86_64.a*.

LIA RAL 3.0:

- Se descarga la versión correspondiente de:
  http://mistral.univ-avignon.fr/download.html
- Se crea un directorio para la carpeta descomprimida, se ubica allí y se desarrolla la siguiente secuencia:

```
$ svn co http://alize.univ-avignon.fr/svn/LIA_RAL/trunk/
```

- En la carpeta cambiar el nombre del archivo *'configure.in'* por *'configure.ac'*, para evitar la advertencia.

```
$ sudo aclocal
$ sudo automake --add-missing
```

- Para evitar que no instale *./compile*, sin embargo se obtiene una advertencia similar al presentado en ALIZE, el cual se ignora.

```
$ sudo autoconf
```

- se lleva el archivo *libalize_Linux_x86_64*.a a la carpeta de ALIZE

```
$ ./configure --with-alize=/home/andres/Proyecto/ALIZE
```

(Dirección en donde se creó el archivo cuando se instaló ALIZE sin el */lib*)

```
$ sudo make
$ sudo make install
```

- Si se instala correctamente se crearan los ejecutables en la carpeta respectiva de LIA RAL. Para este caso quedan instalados en la carpeta *bin* dentro del LIA_RAL.

SPHERE 2.7: Para la instalación de SPHERE en Ubuntu 14.04, se realizó lo siguiente:

- descargar sphere 2.7, una versión más reciente, de la siguiente URL: http://www.nist.gov/itl/iad/mig/tools.cfm
- cambiar el nombre del archivo comprimido a sphere_2.7.tar.bz2

```
bunzip2 -dc sphere\_2.7.tar.bz2 | tar xvf -
cd nist
sh src/scripts/install.sh
```

- Usar la siguiente secuencia:

```
        1:  Linux
        2:  OSX
        3:  cygwin
        4:  Custom

Please Choose one:
   4

Using the Custom Defaults

What is/are the Compiler Command ? [cc]
```

```
gcc
OK, The Compiler_Command command is 'gcc'.  Is this OK? [yes
   ]
yes
What is/are the Compiler Flags ? [-g]
-g
OK, The Compiler_Flags command is '-g'.  Is this OK? [yes]
yes
What is/are the Install Command ? [install -s -m 755]
install -s -m 755
OK, The Install_Command command is 'install -s -m 755'.  Is
   this OK? [yes]
yes
What is/are the Archive Sorting Command ? [ranlib]
ranlib
OK, The Archive_Sorting_Command command is 'ranlib'.  Is
   this OK? [yes]
yes
What is/are the Archive Update Command ? [ar ru]
ar ru
OK, The Archive\_Update\_Command command is 'ar ru'.  Is
   this OK? [yes]
yes
What is/are the Architecture ? [SUN]
GNU/Linux
```

- Si obtiene errores del siguiente estilo:

```
/home/nagaraju/Documents/Speech/nist/lib/
   libsphereCombinedLibs.a(shorten.o): In function `shorten
   ':
/home/nagaraju/Documents/Speech/nist/src/lib/sp/shorten.c
   :499: undefined reference to `log'
/home/nagaraju/Documents/Speech/nist/src/lib/sp/shorten.c
   :554: undefined reference to `floor'
/home/nagaraju/Documents/Speech/nist/src/lib/sp/shorten.c
   :558: undefined reference to `floor'
/home/nagaraju/Documents/Speech/nist/src/lib/sp/shorten.c
   :559: undefined reference to `exp'
```

```
/home/nagaraju/Documents/Speech/nist/src/lib/sp/shorten.c
   :560: undefined reference to 'pow'
```

- Entonces, debe agregar la librería matemática durante el proceso de compilación. Para ello ingrese al siguiente archivo dentro de la carpeta *nist* :

```
lib/makes/make_spg.txt
```

- Agregue *-lm* es decir, cambiar la linea

```
$(EXECUTABLE): $(OBJ) $(PROJECT\_ROOT)/lib/lib$(COMBINEDLIB)
   .a $(CC) $(CFLAGS) $@.c $(OBJ) $(LLIBS) -o $@
```

  por la siguiente,

```
$(EXECUTABLE): $(OBJ) $(PROJECT_ROOT)/lib/lib$(COMBINEDLIB).
   a $(CC) $(CFLAGS) $@.c $(OBJ) $(LLIBS) -o $@ -lm
```

- Correr nuevamente

```
\item Ejecutar: \begin{lstlisting}[language=bash]
   make check
```

- Al final encontrará el siguiente mensaje,

```
*******  ALL TESTS SUCCESSFULLY COMPLETED *******
```

- Adicionalmente se podrá observar la presencia de 20 ejecutables en la carpeta */bin*

SPRO 4.0:

- Se descarga la versión correspondiente en:
  www.irisa.fr/metiss/guig/spro/download.html
- se descomprime y se ubica en la dirección de la carpeta spro 4.0 y se ejecuta lo siguiente:

```
$ ./configure --prefix=/home/andres/Proyecto/spro-4.0 --with
   -sphere=/home/andres/Proyecto/nist
```

- El comando *prefix* da la dirección donde serán instalados los ejecutables, y el *with* permite encontrar las librerías instaladas por sphere.

```
$ sudo make
$ sudo make install
```

- Se obtienen los ejecutables en las carpetas *bin* y *lib*.
- En este caso, Ubuntu 14.04, se obtuvo este error :

  undefined reference to 'sqrt ... además de otros errores similares. Para ello se hace la siguiente secuencia:

```
$ ./configure CFLAGS="-Wall -O3" --prefix=/home/franklin/
   TOOLS/ALIZE/spro-4.0 --with-sphere=/home/franklin/TOOLS/
   ALIZE/nist
```

- Ir al archivo *Makefile* inmediatamente en la carpeta spro-4.0 y colocar la invocación de la librería matemática para la compilación de último puesto. Para ello buscar la siguiente linea:

```
LDADD = -lm -L. -lspro -L/home/franklin/TOOLS/ALIZE/nist/lib
    -lsp -lutil
```

- cambiarla por

```
LDADD = -L. -lspro -L/home/franklin/TOOLS/ALIZE/nist/lib -
   lsp -lutil -lm
```

- guardar, cerrar y hacer lo siguiente

```
make
make install
```

  Aparecerán archivos ejecutables en la carpeta bin. Y de esta manera se finaliza el proceso de instalación de las herramientas requeridas en este trabajo.

APÉNDICE B

CONFIGURACIÓN ALIZE

Éste apéndice sirve como referencia para la sección de experimentos. A continuación se muestra un listado de configuraciones necesarias para utilizar las herramientas de ALIZE-LIA_RAL, cabe aclarar que algunas directivas están sujetas a cambios tales como el género del hablante o la dirección donde se encuentran los parámetros, entre otras. Finalmente las configuraciones utilizadas se encuentran organizadas de acuerdo al orden en que se utilizan los ejecutables, ya presentados en el método (2.4).

B.1. NORMFEAT

```
   *** NormFeat config File ***
mode norm
bigEndian false
loadFeatureFileFormat SPRO4
saveFeatureFileFormat SPRO4
loadFeatureFileExtension .prm
saveFeatureFileExtension .enr.prm
featureServerBufferSize ALL_FEATURES
sampleRate 100
saveFeatureFileSPro3DataKind FBCEPSTRA
labelSelectedFrames all
addDefaultLabel true
defaultLabel all
segmentalMode false
writeAllFeatures true
frameLength 0.01
```

```
vectSize 60
featureServerMode FEATURE_WRITABLE
featureServerMemAlloc 50000000
featureFilesPath ./prm/
labelFilesPath ./lbl/
inputFeatureFilename ./lst/all_male_list.lst
debug false
verbose true
```

B.2. ENERGYDETECTOR

```
  *** EnergyDetector Config File ***
loadFeatureFileExtension .enr.prm
minLLK -200
maxLLK 1000
bigEndian false
loadFeatureFileFormat SPRO4
saveFeatureFileFormat SPRO4
saveFeatureFileSPro3DataKind FBCEPSTRA
featureServerBufferSize ALL_FEATURES
featureServerMemAlloc 50000000
mixtureFilesPath ./gmm/
lstPath ./lst/
labelOutputFrames speech
labelSelectedFrames all
addDefaultLabel true
defaultLabel all
saveLabelFileExtension .lbl
frameLength 0.01
segmentalMode file
nbTrainIt 8
varianceFlooring 0.0001
varianceCeiling 1.5
alpha 0.0
mixtureDistribCount 3
featureServerMask 19
vectSize 1
```

```
baggedFrameProbabilityInit 0.001
thresholdMode weight
featureFilesPath ./prm/
labelFilesPath ./lbl/
inputFeatureFilename ./lst/all_male_list.lst
debug false
verbose true
```

B.3. NORMFEAT

```
   *** NormFeat config File ***
mode norm
bigEndian false
loadFeatureFileFormat SPRO4
saveFeatureFileFormat SPRO4
loadFeatureFileExtension .prm
saveFeatureFileExtension .norm.prm
featureServerBufferSize ALL_FEATURES
sampleRate 100
saveFeatureFileSPro3DataKind FBCEPSTRA
labelSelectedFrames speech
segmentalMode false
writeAllFeatures true
frameLength 0.01
featureServerMode FEATURE_WRITABLE
featureServerMemAlloc 50000000
featureFilesPath ./prm/
labelFilesPath ./lbl/
inputFeatureFilename ./lst/all_male_list.lst
debug false
verbose true
```

B.4. TRAINWORLD

El entrenamiento del UBM, se realiza primero entrenando un modelo inicial y luego a partir de ese modelo se obtiene el UBM final, a continuación se muestran las dos configuraciones utilizadas.

```
  *** TrainWorld Configuration File ***
distribType GD
mixtureDistribCount 80
loadMixtureFileExtension .gmm
saveMixtureFileExtension .gmm
loadFeatureFileExtension .norm.prm
maxLLK 200
minLLK -200
bigEndian false
saveMixtureFileFormat RAW
loadMixtureFileFormat RAW
loadFeatureFileFormat SPRO4
featureServerBufferSize ALL_FEATURES
featureServerMemAlloc 100000000
featureFilesPath ./prm/
mixtureFilesPath ./gmm/
labelFilesPath ./lbl/
frameLength 0.01
lstPath ./lst/
labelSelectedFrames speech
baggedFrameProbability 0.2
baggedFrameProbabilityInit 0.5
initVarianceFlooring 0.5
initVarianceCeiling 1
finalVarianceFlooring 0.5
finalVarianceCeiling 5.0
normalizeModel true
nbTrainIt 15
use01 true
verbose true
featureServerMask 0-18,20-50
baggedMinimalLength 1
baggedMaximalLength 1
inputStreamList ./lst/Train_UBM.lst
weightStreamList ./lst/Train_UBM.weight
outputWorldFilename world_init
```

```
debug false
verbose true

   *** TrainWorld Configuration File ***
distribType GD
mixtureDistribCount 80
loadMixtureFileExtension .gmm
saveMixtureFileExtension .gmm
loadFeatureFileExtension .norm.prm
maxLLK 200
minLLK -200
bigEndian false
saveMixtureFileFormat RAW
loadMixtureFileFormat RAW
loadFeatureFileFormat SPRO4
featureServerBufferSize ALL_FEATURES
featureServerMemAlloc 100000000
featureFilesPath ./prm/
mixtureFilesPath ./gmm/
labelFilesPath ./lbl/
frameLength 0.01
lstPath ./lst/
labelSelectedFrames speech
baggedFrameProbability 1
initVarianceFlooring 0.5
initVarianceCeiling 5.0
finalVarianceFlooring 0.5
finalVarianceCeiling 5.0
featureServerMask 0-18,20-50
normalizeModel true
use01 true
nbTrainIt 4
baggedMinimalLength 1
baggedMaximalLength 1
inputStreamList ./lst/Train_UBM.lst
weightStreamList ./lst/Train_UBM.weight
inputWorldFilename world_init
```

```
outputWorldFilename world
debug false
verbose true
```

B.5. TRAINTARGET

```
   *** TrainTarget Configuration File ***
maxLLK 200
minLLK -200
bigEndian false
saveMixtureFileFormat RAW
loadMixtureFileFormat RAW
loadFeatureFileFormat SPRO4
featureServerBufferSize ALL_FEATURES
loadMixtureFileExtension .gmm
saveMixtureFileExtension .gmm
loadFeatureFileExtension .norm.prm
featureFilesPath ./prm/
labelFilesPath ./lbl/
mixtureFilesPath ./gmm/
nbTrainIt 1
labelSelectedFrames speech
normalizeModel true
normalizeModelMeanOnly true
normalizeModelNbIt 5
MAPAlgo MAPOccDep
meanAdapt true
MAPRegFactorMean 14
frameLength 0.01
featureServerMemAlloc 100000000
featureServerMask 0-18,20-50
targetIdList ./ndx/trainmodel.ndx
inputWorldFilename world
debug false
verbose true
```

B.6. COMPUTETEST

```
   *** ComputeTest Configuration File ***
topDistribsCount 10
computeLLKWithTopDistribs COMPLETE
loadMixtureFileExtension .gmm
loadFeatureFileExtension .norm.prm
maxLLK 200
minLLK -200
bigEndian false
loadFeatureFileFormat SPRO4
loadMixtureFileFormat RAW
featureServerBufferSize ALL_FEATURES
mixtureFilesPath ./gmm/
labelSelectedFrames speech
segmentalMode false
gender M
frameLength 0.01
featureServerMemAlloc 100000000
featureFilesPath ./prm/
labelFilesPath ./lbl/
channelCompensation true
featureServerMask 0-18,20-50
inputWorldFilename world
ndxFilename ./ndx/test_H.ndx
targetIdList ./ndx/trainmodel.ndx
outputFilename ./res/H3.res
debug false
verbose true
```

B.7. COMPUTENORM

```
   *** ComputeNorm Configuration File ***
verboseLevel 1
bigEndian false
verbose true
debug true
```

```
normType tnorm
maxSegNb 1000000
testNistFile ./res/H3.res
tnormNistFile ./res/H_imp-seg_gmm.res
outputFileBaseName ./res/Hnorm.res
```

APÉNDICE C

LÉEME Y DESCRIPCIÓN DEL SCRIPT

Este escrito sirve como ayuda para el entendimiento y futura utilización del archivo *Run_front_to_End.py* dónde se desarrolla un sistema de verificación del hablante basado en modelos de mezclas Gaussianas (GMM-UBM), en primera parte se describirá los pasos necesarios para que el proceso funcione y en seguida se darán unas breves explicaciones del *script*.

1. Instalación del software necesario

   En esta sección vamos a remitirnos a seguir los pasos del tutorial de instalación de herramientas que se encuentra como anexo a éste trabajo de grado, entre ellas se encuentra ALIZE 3.0, LIA_RAL 3.0, spro 4.0 y sphere 2.6. Cabe aclarar que esto ha sido probado en Ubuntu 14.04.

2. Creación y Organización de ficheros

   Se crean dos ficheros principales: Corpus, dónde se almacenará la base de datos en formato .waw; Prueba, dónde se encontrará el *script* y los ficheros adicionales para la organización del trabajo. Como ficheros adicionales se crean:

   - *cfg*, donde irán los archivos de configuración de cada uno de los ejecutables utilizados en el script *Run_front_to_End.py*, y que tienen extensión .cfg
   - *gmm*, es el fichero dónde se almacenaran los modelos de mezclas gaussianas creados por el script, y que tienen extensión .gmm
   - *lst*, allí se encuentran las listas de las voces de las base de datos, necesarias para la ejecución del script, con extensión .lst

- *ndx*, en este fichero se encuentran unas listas adicionales con extensión .ndx que se encuentran en formato NIST, utilizadas en la realización de experimentos y creación de modelos.
- *res*, allí aparecerán los resultados una vez se halla completado la ejecución del script, se pueden encontrar dos tipos de extensiones .res o .res.tnorm la primera sin normalizar y la segunda una vez realizada una normalización tipo T, que fue la empleada en este trabajo.
- *lbl*, fichero necesario en la detección de energía del método pues allí crea los archivos que contienen las etiquetas de cada trama de los archivos de audio, se distinguen por su extensión .lbl
- *prm*, por otra parte se crea este fichero y es donde se encontrarán los parámetros de cada uno de los archivos de audio de la base de datos utilizada. Una vez realizada la parte de extracción de parámetros aparecerán archivos con el mismo nombre del audio pero con extensión .prm, enseguida de la primera normalización se crean archivos con extensión .enr.prm y después de la segunda normalización se crean los archivos con extensión norm.prm.

3. Ejecutables de LIA_RAL

   Otro factor importante para el éxito de éste procedimiento es que teniendo una vez instaladas las herramientas, es necesario copiar los ejecutables de LIA_RAL, que se encuentran en la ruta ../LIA_RAL/bin/ en nuestro fichero llamado Prueba, pues el *script* los utiliza en esa dirección.

4. Descripción breve del script *Run_front_to_End.py*

   En la primera sección se definen las rutas de los ficheros creados anteriormente, en seguida se definen las variables que configuran la extracción de parámetros, que se realiza en spro. A partir de allí y hasta el final se emplean los ejecutables del LIA_RAL, y la utilización de estos tienen la misma forma, por tanto con un ejemplo se podrán entender los demás:

```
import os
command = "./TrainWorld --config ./cfg/TrainWorldinit.cfg --
   inputStreamList ./lst/Train_UBM.lst --weightStreamList ./
```

```
    lst/Train_UBM.weight --outputWorldFilename world_init --
    debug false --verbose true";
os.system(command)
```

las directivas que se encuentran en los demás ejecutables como:

- *./TrainWorld* (ejemplo) Nombre del ejecutable de LIA_RAL que se utilizará
- *config* define el archivo de configuración con extensión cfg que utilizará ese ejecutable
- *featureFilesPath* define la ruta del fichero de parámetros
- *labelFilesPath* define la ruta del fichero de etiquetas de los parámetros
- *inputFeatureFilename* lista de los archivos de parámetros
- *debug false verbose true* directivas necesarias para la utilización de las librerías de ALIZE
- *inputStreamList* se indica el archivo que contiene la ruta de la lista
- *weightStreamList* indica el peso para tener en cuenta cada lista que se utilizará
- *outputWorldFilename* nombre de salida del modelo global
- *inputWorldFilename* nombre del archivo de entrada del modelo global que se empleará en ese ejecutable
- *ndxFilename* archivos con extensión .ndx donde se indican los modelos y los archivos con los que se comparan tales modelos
- *targetIdList* archivos con extensión .ndx donde se indican que archivos de parámetros se utilizaran para crear un modelo
- *outputFilename* se indica el nombre del archivo de salida
- *gender* se indica el género correspondiente de las grabaciones utilizadas (M/F)
- *testNistFile* archivo de resultados que se desea normalizar. (tienen extensión .res)
- *tnormNistFile* (extensión .res) archivo que contiene los resultados al comparar los modelos de los que se desean conocer los scores con los modelos de los impostores

- *outputFileBaseName* nombre del archivo de salida para el ejecutable ComputeNorm, se diferencian con su extensión res.tnorm pues en este trabajo se aplica una normalización Tnorm.

Cabe aclarar que la configuración mas relevante que se utiliza en cada ejecutable se encuentra en su respectivo archivo .cfg y además tiene su ayuda ejecutando desde la terminal

```
./NombredelEjecutable --help
```

Y finalmente para mayor información referirse a:

A. Larcher, J.-F. Bonastre, B. Fauve, K.A. Lee, C. Levy, H. Li, J.S.D. Mason, J.-Y Parfait, .ALIZE 3.0 - Open Source Toolkit for State-of-the-Art Speaker Recognition,ïn Annual Conference of the International Speech Communication Association (Interspeech), 2013

APÉNDICE D

SCRIPT EN PYTHON SISTEMA GMM-UBM

A continuación se presenta el código llamado *Run_front_to_end.py* desarrollado en Python para la realización de un sistema de verificación del hablante GMM-UBM, para su utilización basta con llamarlo desde la terminal.

```
#!/usr/bin/python3
loc_audios = "/home/dago/Proyecto_de_grado_v2/con_UIS/corpus/";
loc_params = "/home/dago/Proyecto_de_grado_v2/con_UIS/
   Prueba_to_share/prm/";
loc_config = "/home/dago/Proyecto_de_grado_v2/con_UIS/
   Prueba_to_share/cfg/";
loc_lst    = "/home/dago/Proyecto_de_grado_v2/con_UIS/
   Prueba_to_share/lst/";

#1.:::::: EXTRACCION DE PARAMETROS
   ::::::::::::::::::::::::::::
# Variables para la configuracion en la extraccion de parametros

audio_format = '-F wave '; # Para indicar el formato de los
   archivos de audio de la base de datos
num_ceps = '-p19 '; # numero de coeficientes ceptrales en este
   caso son 19
form_ceps = '-m '; #para la extraccion de MFCC
energy = '-e '; # para agregar log-energy al vector de
   parametros
first_d = '-D '; # Agrega la primera derivada al vector de
   parametros
```

```
second_d = '-A'; # Agrega la segunda derivada al vector de
   parametros



##VOCES MASCULINAS
## Se generan todos los vectores de parametros para las senales
   de voz Masculinas
f = open("/home/dago/Proyecto_de_grado_v2/con_UIS/
   Prueba_to_share/lst/all.lst","r")
while 1:
                line = f.readline()
                line = line.rstrip('\n')
                if not line:break
                #the process
                name = line;
                loc ='';
                for y in xrange(1, 31):
                        if  loc == 'H30':
                                pass
                        else:
                                y_act=' %d'  % (y);
                                loc='H' + y_act;
                                input_file = loc_audios+loc+'_'+
                                   name+".wav";
                                print(input_file)
                                output_file = loc_params+loc+'_'
                                   +name+".prm";
                                print(output_file)
                                config_params = audio_format+
                                   num_ceps+form_ceps+energy+
                                   first_d+second_d;
                                print(config_params)
                                routine = "./sfbcep";
                                command = routine + " " +
                                   config_params + " " +
                                   input_file + " " +
                                   output_file;
                                import os
```

```
                                    os.chdir('/home/dago/
                                       Proyecto_de_grado_v2/spro-4.0
                                       ')
                                    os.system(command)

f.close()

#VOZ MASCULINA
## Normaliza los parametros con los siguientes comandos
import os
os.chdir("/home/dago/Proyecto_de_grado_v2/con_UIS/
   Prueba_to_share/")
con_file='NormeFeatenergy.cfg';
part_config = loc_config+con_file;
part_lst    = loc_lst+"all_male_list.lst";
command = "./NormFeat --config "+part_config+" "+"--
   featureFilesPath ./prm/ --labelFilesPath ./lbl/ --
   inputFeatureFilename"+" "+part_lst+" --debug false --verbose
   true";
print(command)
os.system(command)

# Los siguientes comandos detectan la energia de todas las
   senales
import os
os.chdir("/home/dago/Proyecto_de_grado_v2/con_UIS/
   Prueba_to_share/")
con_file='EnergyDetector.cfg';
part_config = loc_config+con_file;
part_lst    = loc_lst+"all_male_list.lst";
command = "./EnergyDetector --config "+part_config+" "+"--
   featureFilesPath ./prm/ --labelFilesPath ./lbl/ --
   inputFeatureFilename"+" "+part_lst+" --verbose true --debug
   false"
print(command)
os.system(command)

# Re-normaliza los parametros con el siguiente comandos
import os
```

```
os.chdir("/home/dago/Proyecto_de_grado_v2/con_UIS/
   Prueba_to_share/")
con_file='NormeFeat.cfg';
part_config = loc_config+con_file;
part_lst    = loc_lst+"all_male_list.lst";
command = "./NormFeat --config "+part_config+" "+"--
   featureFilesPath ./prm/ --labelFilesPath ./lbl/ --
   inputFeatureFilename"+" "+part_lst;
print(command)
os.system(command)

#2.:::::::::::::: MODELO UNIVERSAL (UBM)
   ::::::::::::::::::::::::

# Se crea un modelo universal (UBM) en dos partes
import os
command = "./TrainWorld --config ./cfg/TrainWorldinit.cfg --
   inputStreamList ./lst/Train_UBM.lst --weightStreamList ./lst/
   Train_UBM.weight --outputWorldFilename world_init --debug
   false --verbose true";
os.system(command)
command = "./TrainWorld --config ./cfg/TrainWorldFinal.cfg --
   inputStreamList ./lst/Train_UBM.lst --weightStreamList ./lst/
   Train_UBM.weight  --outputWorldFilename world --
   inputWorldFilename world_init --debug false --verbose true"
os.system(command)


#3.:::::: ENTRENAMIENTO DE MODELOS GMM PARA CADA HABLANTE
   ::::::::::::::::::::
# Hombres
import os
command = './TrainTarget --config ./cfg/trainTarget.cfg --
   targetIdList ./ndx/trainmodel.ndx --inputWorldFilename world
   --debug false --verbose true'
os.system(command)

#Modelos de los impostores
command = './TrainTarget --config ./cfg/trainTarget.cfg --
   targetIdList ./ndx/trainImpostor.ndx --inputWorldFilename
```

```
   world --debug false --verbose true'
os.system(command)


#5.::::::: EXPERIMENTOS ::::::::::::
# Se realiza las pruebas con los modelos hallados anteriormente
   (las pruebas se describen en el archivo ndx)
import os
command = "./ComputeTest --config ./cfg/ComputeTest.cfg  --
   ndxFilename ./ndx/test_H.ndx --targetIdList ./ndx/trainmodel.
   ndx --outputFilename ./res/H3.res --gender M --debug false --
   verbose true"
os.system(command)

command = "./ComputeTest --config ./cfg/ComputeTest.cfg  --
   ndxFilename ./ndx/test_Imp.ndx --targetIdList ./ndx/
   trainImpostor.ndx --outputFilename ./res/H3_Imp.res --gender
   M --debug false --verbose true"
os.system(command)

#6. ::::::: Post-Procesamiento ::::::::::::::::::

# Compute test for score normalization (t-norm)
import os
command = "./ComputeTest --config ./cfg/ComputeTest.cfg   --
   ndxFilename ./ndx/computetest_gmm_imp-seg.ndx --targetIdList
   ./ndx/trainImpostor.ndx --outputFilename ./res/H_imp-seg_gmm.
   res --gender M --debug false --verbose true"
os.system(command)

command = "./ComputeTest --config ./cfg/ComputeTest.cfg   --
   ndxFilename ./ndx/computetest_IMP.ndx --targetIdList ./ndx/
   trainImpostor.ndx --outputFilename ./res/imp-seg_gmm.res --
   gender M --debug false --verbose true"
os.system(command)

#6.::::::::::::::NORMALIZACION DE LOS RESULTADOS
   ::::::::::::::::::::::
# Compute T-Norm
import os
```

```
command = "./ComputeNorm --config ./cfg/ComputeNorm_tnorm.cfg --
   testNistFile ./res/H3.res --tnormNistFile ./res/H_imp-seg_gmm
   .res --outputFileBaseName ./res/Hnorm.res"
os.system(command)

command = "./ComputeNorm --config ./cfg/ComputeNorm_tnorm.cfg --
   testNistFile ./res/H3_Imp.res --tnormNistFile ./res/imp-
   seg_gmm.res --outputFileBaseName ./res/Hnorm_IMP.res"
os.system(command)

#7.Decision making
#import os
#command = './Scoring --mode NIST --threshold 2 --segTypeTest 1
   side --trainTypeTest 1side --adaptationMode n --inputFile
   target_imp-seg_gmmES2.res.tnorm --outputFile male_ES2D'
#os.system(command)
```