Universidad Zaragoza
1542

Trabajo Fin de Máster

Estudio y mejora de sistemas de verificación de locutores bajo condiciones de voz afónica

Study and improvement of speaker verification systems under hoarse voice conditions

Autor

Isabel Querol

Director

Santiago Prieto

Máster en Ingeniería de telecomunicación

ESCUELA DE INGENIERÍA Y ARQUITECTURA

2020

TRABAJOS DE FIN DE GRADO / FIN DE MÁSTER

Escuela de Ingeniería y Arquitectura Universidad Zaragoza

DECLARACIÓN DE AUTORÍA Y ORIGINALIDAD

(Este documento debe entregarse en la Secretaría de la EINA, dentro del plazo de depósito del TFG/TFM para su evaluación).

D./Dª. Isabel Querol Cisneros ,en aplicación de lo dispuesto en el art. 14 (Derechos de autor) del Acuerdo de 11 de septiembre de 2014, del Consejo de Gobierno, por el que se aprueba el Reglamento de los TFG y TFM de la Universidad de Zaragoza,

Declaro que el presente Trabajo de Fin de (Grado/Máster) Máster

(Título del Trabajo)

Estudio y mejora de sistemas de verificación de locutores bajo condiciones de voz afónica

es de mi autoría y es original, no habiéndose utilizado fuente sin ser citada debidamente.

Zaragoza, 25 de Junio de 2020

Fdo: Isabel Querol Cisneros

Agradecimientos

A das-Nano, por darme la oportunidad de trabajar en un lugar donde empresa e investigación se unen para dar resultados brillantes.

Al equipo de voz, en especial a Santiago, director de mi trabajo, por ofrecerme la ayuda para desarrollarlo y la libertad para hacerlo con creatividad.

Resumen

Este proyecto analiza las diferencias entre distintos modos del habla, en especial el susurro, utilizado como medio de comunicación cuando se padece una enfermedad como la afonía, y cómo afectan a los sistemas de verificación automática de locutores. El objetivo del proyecto es el estudio de la pérdida de prestaciones, y la mejora de los sistemas mediante la aplicación de distintas técnicas.

El estudio parte del análisis de las señales en los dominios de voz susurrada y voz neutra, cuyas diferencias explican el detrimento del sistema. Para cuantificarlo, se escogen un sistema de referencia de altas prestaciones y una base de datos que cuenta con audios en condiciones de habla normal y susurrada.

Las técnicas de mejora estudiadas abordan el problema en distintos puntos del sistema completo. Estas técnicas se introducen de forma teórica en el segundo bloque del trabajo, y en el tercer bloque se muestran los resultados obtenidos para cada una de ellas. Para evaluarlas y compararlas se utilizan herramientas de software libre, herramientas de visualización y entrenamiento de modelos estadísticos utilizando *Python* como lenguaje de programación principal. El trabajo muestra el rendimiento de herramientas alternativas a los algoritmos populares de aprendizaje automático, necesarias cuando no se dispone de una cantidad significativa de datos que permitan buenos resultados.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

Las tecnologías de verificación de locutores se encuentran en auge en la actualidad y se utilizan en multitud de aplicaciones como: control de acceso, comercio, autenticación en transacciones de correo electrónico, aplicaciones forenses, aplicaciones de domótica... La gran mayoría de los sistemas desarrollados funcionan de forma precisa con voz neutra. Sin embargo, las enfermedades de la voz como la afonía impiden al locutor utilizar la voz neutra, forzándolo a utilizar el susurro. En estas condiciones, los sistemas sufren una degradación importante. Este capítulo cubre una breve introducción al problema de estudio, la motivación, objetivos y desarrollo del mismo.

1.1. Motivación y contexto del proyecto

Las tecnologías de reconocimiento de locutores pueden proporcionar una forma segura de acceso o gestión de datos multimedia. La evolución de estas tecnologías ha llevado al desarrollo de sistemas automáticos de identificación y verificación de locutores[1]. La verificación de locutores es el proceso por el cual se confirma o rechaza la identidad de un locutor comparando dos muestras de voz; una referencia de la identidad, recogida durante el registro del locutor en el sistema y otra de test cuyo locutor pretende verificarse[2]. En los sistemas de verificación de locutores independientes del texto, pueden llegar a

compararse muestras muy diferentes entre sí (en duración, en condiciones de ruido, contenido léxico...)[3]. Es por ello, que en estos sistemas se hace necesario el análisis y caracterización del locutor a partir de un conjunto de parámetros o características de su voz. La mayoría de los sistemas desarrollados para la verificación de locutores, utilizan muestras de locutores cooperativos, es decir, locutores que utilizan la voz neutra permitiendo que el sistema reconozca su identidad. Diversas características de la voz neutra como la fonación, donde se estudia la información de resonancia presente en las señales periódicas que la componen, permiten caracterizar al locutor de manera fiable[4].

Algunas enfermedades de la voz, como la afonía, impiden que los locutores utilicen su voz neutra forzándoles a utilizar el susurro para comunicarse. El rendimiento de los sistemas de verificación de locutores entrenados utilizando la voz neutra se ve altamente perjudicado cuando trata de verificarse a un locutor a través del susurro. Las características de la voz neutra difieren mucho de las de la voz susurrada. La condición de fonación, por ejemplo, se ve alterada ya que, al susurrar se produce una turbulencia de aire que no hace vibrar las cuerdas vocales[5].

El mayor desafío al que se enfrentan las técnicas que tratan de mejorar estos sistemas en condiciones de susurro, es la falta de datos de voz susurrada en comparación a la gran cantidad de datos que sí existen de voz neutra[6, 7]. Esto dificulta el entrenamiento de modelos complejos donde se requieren gran cantidad de datos.

1.2. Objetivos

Este proyecto consiste en el análisis de las características de la voz susurrada, el impacto de esta condición en sistemas de verificación de locutores y el estudio de las técnicas de mejora del estado del arte con el objetivo de:

1. Encontrar las similitudes y diferencias entre la voz neutra y la voz susurrada a través de técnicas de procesado de voz y herramientas estadísticas que puedan ser explotadas

para la mejora del rendimiento de los sistemas.

2. Cuantificar el deterioro de los sistemas de verificación de locutores en condiciones de voz susurrada.

3. Comparar el rendimiento de las técnicas propuestas en la mejora de un sistema de referencia entrenado en condiciones de voz neutra cuando se enfrenta a condiciones de voz susurrada. La principal diferencia entre las herramientas estudiadas son los datos que utilizan y procesan que pueden ser: datos de la voz previos al sistema de verificación (datos de *front-end*), vectores de salida de la red neuronal utilizada para la verificación (datos de *back-end*) o las puntuaciones utilizadas para determinar la verificación del locutor.

4. Estudiar el deterioro de los sistemas basados en redes neuronales cuando no se dispone de una cantidad de datos representativa de un conjunto y sus alternativas de mejora.

1.3. Metodología del trabajo

A continuación se describen las fases en las que se ha desarrollado el trabajo por orden cronológico.

1.3.1. Búsqueda y selección de la base de datos para el estudio.

La base de datos escogida debe disponer de muestras etiquetadas tanto de voz neutra como de voz susurrada. Se selecciona la base de datos CHAINS (CHaracterizing Individual Speakers)[8] descrita el apartado 2.3 de la memoria.

1.3.2. Estudio de las características de la voz susurrada y la voz neutra.

Para establecer las diferencias que existen entre el dominio de la voz susurrada y el de la voz neutra se recurre a diferentes técnicas de procesado de señal que permiten visualizar los datos en el dominio de la frecuencia y del tiempo.

1.3.3. Selección, estudio y evaluación del sistema de referencia

Con el objetivo de evaluar el deterioro de los sistemas en condiciones de susurro se elige un sistema del estado del arte con una tasa de error muy baja en condiciones de voz neutra. El sistema escogido es el propuesto por Kaldi[9], entrenado con las muestras del corpus *Voxceleb* y descrito en el apartado 1.4 de la memoria. El sistema se evalúa para los casos de: registro en el sistema con voz neutra y verificación con voz neutra (*neutra-neutra*), registro con voz susurrada y verificación con voz susurrada (*susurro-susurro*) y registro con voz neutra y verificación con voz susurrada (*neutra-susurro*).

1.3.4. Estudio de las técnicas propuestas para la mejora de los sistemas.

- Técnicas aplicadas a los datos de voz. Estas técnicas se fundamentan en el procesado de la voz y pretenden explotar las similitudes entre la voz susurrada y la voz neutra. Entre ellas encontramos también aquellas cuyo objetivo es el mapeo de datos de un dominio al otro. Algunas de las técnicas estudiadas no pueden ser utilizadas como mejora del sistema de referencia porque utilizan datos distintos a los datos utilizados para el entrenamiento del sistema.

- Técnicas aplicadas a los vectores de salida de la red neuronal. Las herramientas estudiadas tratan de acercar los datos de ambos dominios a la salida de la red. Para ello se utilizan técnicas de mapeo de datos. La herramienta TSNE facilita la

visualización de los vectores en dos dimensiones permitiendo un análisis acertado de los resultados.

- Técnicas aplicadas a las puntuaciones que determinan la verificación. Las puntuaciones calculadas para cada comparación entre vectores son clasificadas y calibradas de acuerdo con esta clasificación. Encontramos tres tipos de comparaciones: voz neutra – voz neutra, voz susurrada – voz susurrada y voz neutra – voz susurrada.

1.3.5. Implementación y evaluación del sistema completo escogido

Tras comparar los resultados de las distintas técnicas se evalúa el sistema completo utilizando aquellas que han dado los mejores resultados. Para que se trate de un sistema que pueda implementarse en un entorno real, no se dispone de información a priori de las muestras en la evaluación.

1.4. Estructura de la memoria

Este trabajo queda estructurado de la siguiente forma:

- En el capítulo 2, se exponen las técnicas estudiadas, así como la base de datos utilizada y el sistema de referencia seleccionado. El capítulo comienza por un análisis de las diferencias entre los dominios de voz susurrada y voz neutra a través de su representación en el tiempo y la frecuencia.

- En el capítulo 3, se muestran los resultados obtenidos para cada una de las herramientas de mejora propuestas y se plantean las ventajas e inconvenientes de su aplicación en la mejora del sistema de referencia. El capítulo termina con un diagrama de bloques que resume el sistema final con el que se han obtenido los mejores resultados.

- En el capítulo 4, se enumeran las conclusiones del trabajo listando de forma resumida las ventajas y desventajas de cada una de las técnicas y su eficiencia en la mejora del sistema.

Capítulo 2

Estudio del estado del arte

2.1. Estudio de las características de la voz susurrada

El proceso de producción del habla humana puede verse como una operación de filtrado, en el que tres cavidades (el tracto vocal, formado por las cavidades bucal y faríngea, y la cavidad nasal) componen el principal filtro acústico[10]. Este filtro es excitado por los órganos que tiene debajo (excitación glotal) y son los órganos articulatorios (labios, dientes, lengua, mandíbula y cuerdas vocales) los que se encargan de cambiar las propiedades del sistema, la forma de excitación y la carga de salida a través de los labios a lo largo del tiempo. Cuando hablamos, el aire proveniente de los pulmones viaja a través de la tráquea y en la glotis es interrumpido periódicamente por el movimiento de las cuerdas vocales. La figura 2.1 ilustra el sistema de producción del habla humana[11].

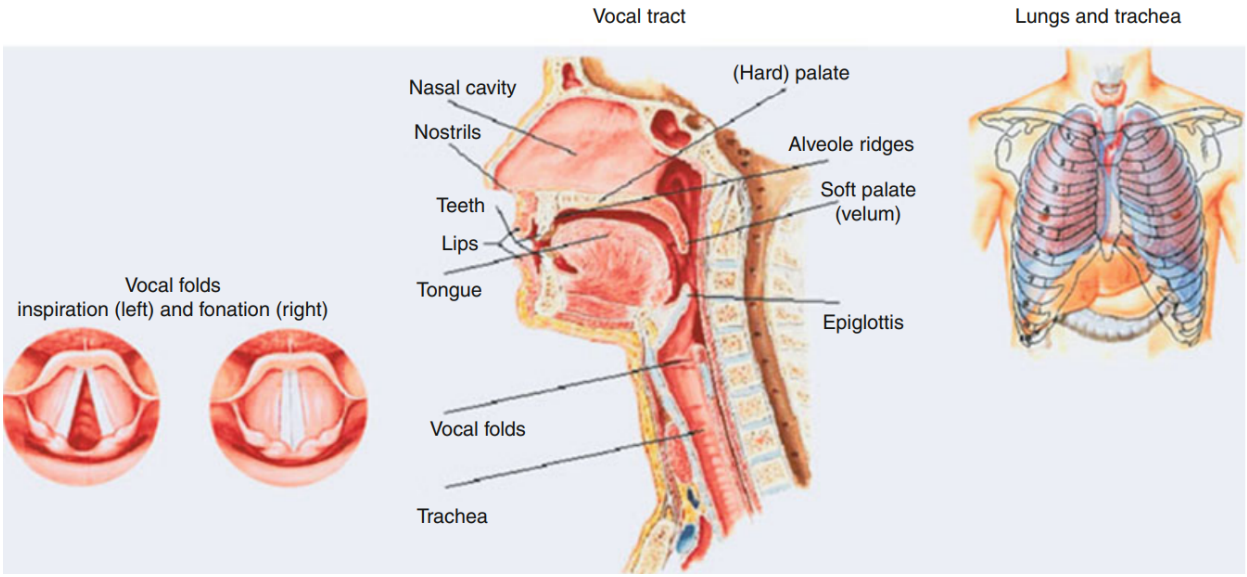
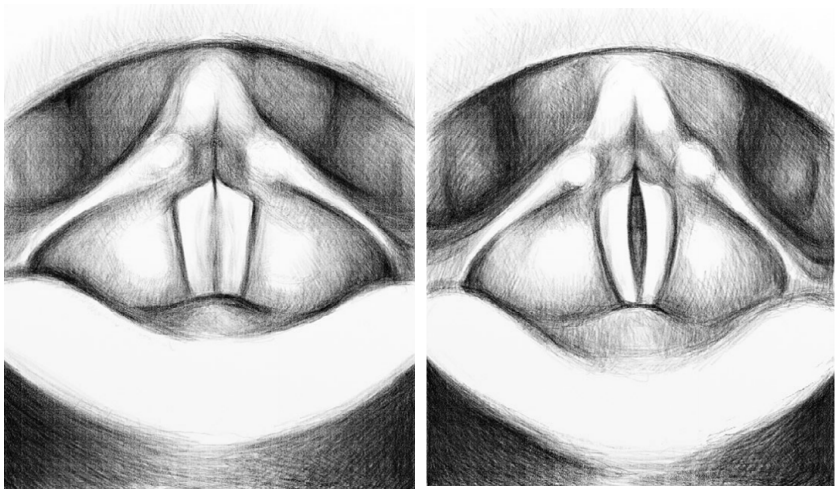


Figura 2.1: Esquemático del mecanismo de producción del habla humana

Los fonemas del habla neutra se producen por una vibración periódica de las cuerdas vocales que genera un flujo de aire hacia la faringe, cavidades orales y nasales. Sin embargo, cuando se susurra, las cuerdas vocales permanecen abiertas e inmóviles, dejando que se produzca una corriente de aire continua sin excitación periódica. El flujo de aire de los pulmones se utiliza como fuente de excitación sonora y la posición de la faringe se ajusta para que las cuerdas vocales no vibren[12]. Las figuras 2.2 y 2.3 muestran los estados de la laringe en voz neutra y susurrada respectivamente[13].



(a) Voz neutra (b) Prefonación

Figura 2.2: Estados de la laringe en la voz neutra

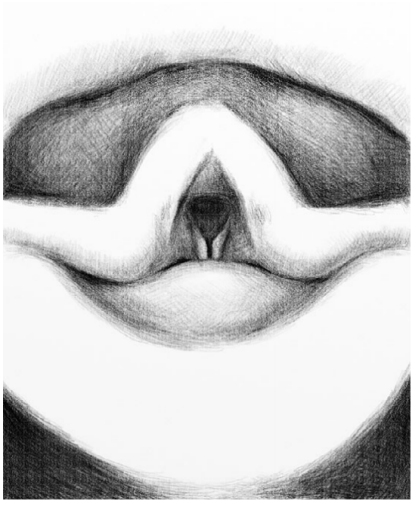
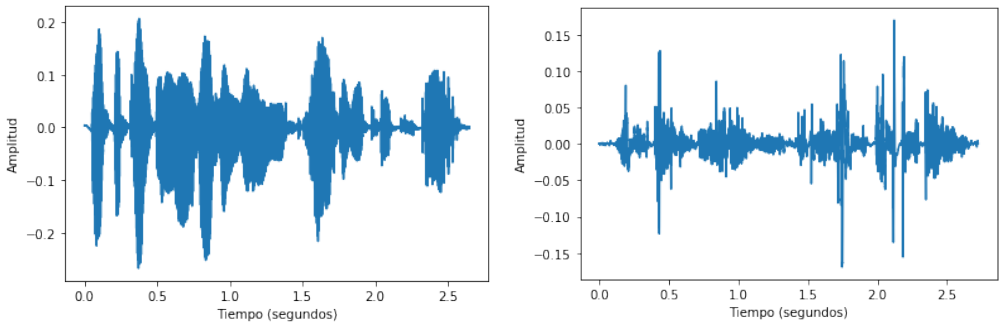


Figura 2.3: Estado de la laringe en la voz susurrada

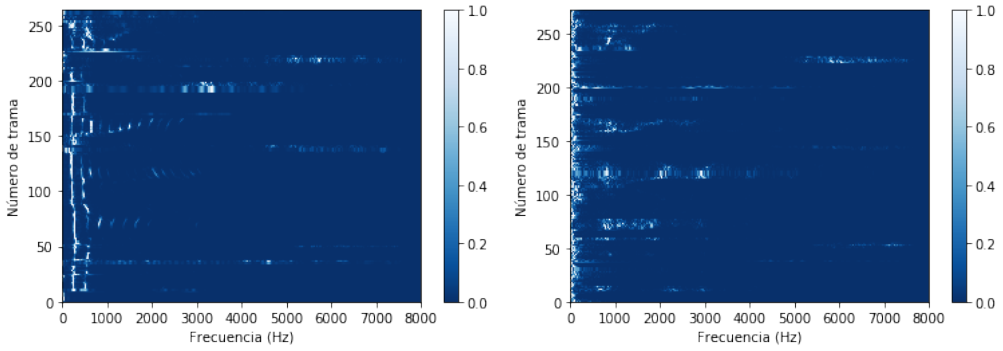
La diferencia en la forma de producción de la voz susurrada y la voz neutra se refleja en la forma de onda de la señal de voz. Para ilustrarlo, la figura 2.4 muestra la señal de voz del corpus correspondiente a la frase “If it doesn’t matter who wins, why do we keep score?”del mismo locutor en voz neutra y susurrada a una frecuencia de muestreo de 16KHz.



(a) Voz neutra (b) Voz susurrada

Figura 2.4: Señal de voz en el dominio del tiempo

La forma de onda de la señal de voz susurrada presenta una menor amplitud y la señal es mucho más ruidosa. Por otro lado, la duración de la señal de voz susurrada es ligeramente superior a la de la señal de voz neutra[1, 14, 15]. Para analizar las diferencias entre la voz neutra y la voz susurrada en el dominio de la frecuencia, se estima la densidad espectral de la señal mediante el periodograma de las señales. La señal se procesa en ventanas de Hamming de 25 milisegundos con un solapamiento de 10 milisegundos y se calcula la transformada FFT de 512 puntos sobre las ventanas resultantes. El periodograma de ambas señales se muestra en la figura 2.5.



(a) Voz neutra (b) Voz susurrada

Figura 2.5: Periodograma de la señal de voz

La ausencia de líneas paralelas a bajas frecuencias en el periodograma de voz susurrada que sí se observan en el de voz neutra, constata que no existe excitación harmónica en la señal de voz susurrada. Además los formantes principales de la señal de voz susurrada

se encuentran en frecuencias superiores a aquellas correspondientes a los de la voz neutra [14, 16, 7]. Como las cuerdas vocales no vibran cuando se susurra, la glotis se comporta de forma similar a una fuente de ruido y cambia la calidad espectral de la mayoría de los fonemas. La pendiente espectral de la señal de voz susurrada es más plana, decae menos bruscamente que la de la voz neutra[1, 16, 15]. En la figura 2.6 se representa la media del periodograma a lo largo del eje de la frecuencia, permitiendo una comparación gráfica de ambas pendientes espectrales.

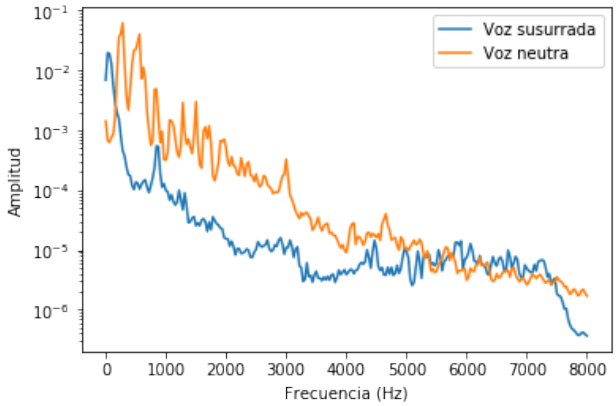
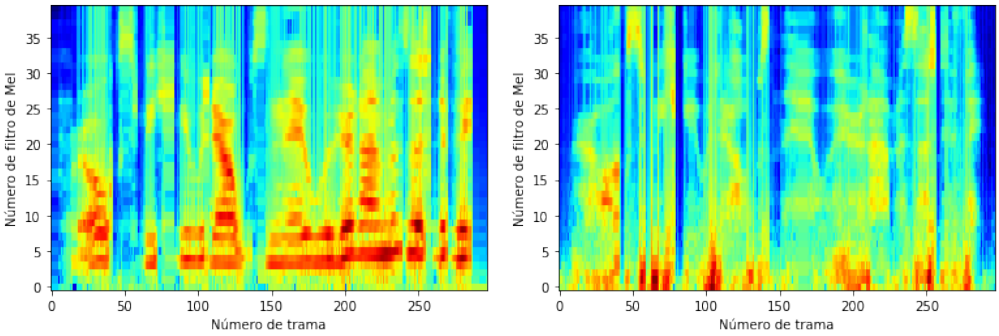


Figura 2.6: Media del periodograma de la señal de voz susurrada y voz neutra

En el análisis de la voz, son de gran utilidad los filtros de Mel ya que se basan en la percepción humana para aplicar un filtrado no lineal sobre la señal de voz y permiten modelar el pitch y otras características frecuenciales de la señal[17, 7]. La figura 2.7 muestra el espectrograma de Mel calculado como el filtrado del periodograma con 40 filtros en la escala de Mel.

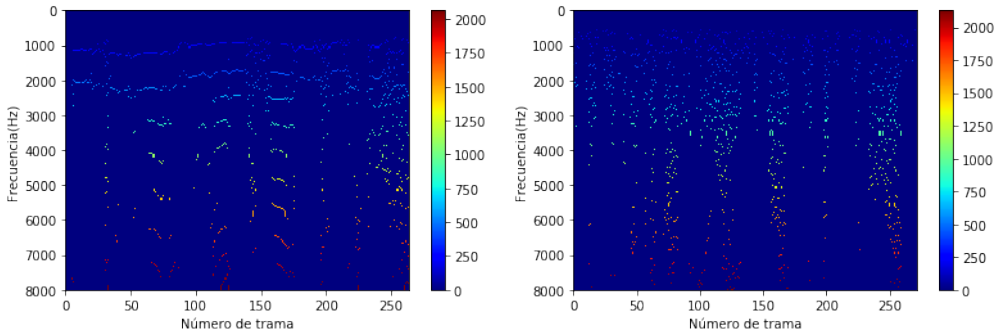


(a) Voz neutra (b) Voz susurrada

Figura 2.7: Espectrograma de la señal de voz

El espectrograma corrobora la ausencia de excitación harmónica en la señal de voz susurrada que se apreciaba en el periodograma, así como la localización de los formantes en frecuencias más altas. Concretamente algunos estudios demuestran que el formante F1 se modifica de forma más acusada que los formantes F2 y F3[16, 18]. Esto se atribuye al cambio en la vibración glotal, ya que se consigue reducir de forma efectiva la longitud total del tracto vocal[19].

Por último, la figura 2.8 muestra la información del pitch conseguida por medio de la umbralización aplicada a la transformada STFT. En esta imagen se observa de forma más clara la falta de contenido armónico en la señal de voz susurrada (no se aprecian las claras líneas horizontales correspondientes a los formantes) y el desplazamiento de los formantes a frecuencias superiores.



(a) Voz neutra (b) Voz susurrada

Figura 2.8: Seguimiento del Pitch de la señal de voz

2.2. Descripción del sistema de referencia

La evaluación de las técnicas estudiadas se basa en la mejora de resultados de un sistema de referencia. El sistema elegido en este proyecto es uno de los ejemplos que ofrece Kaldi, una herramienta dirigida a los investigadores en reconocimiento de locutores[20]. El sistema que se describe en [9] tiene como objetivo discriminar entre distintos locutores para lograr la verificación. La red neuronal (*feed-forward* DNN, Figura 2.9) que compone el núcleo del sistema consigue mapear datos (grabaciones de audio) de longitud variable a vectores de longitud fija a los que se denomina *x-vectors*.

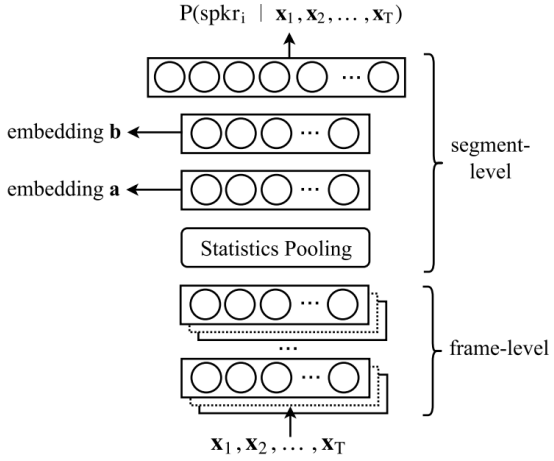


Figura 2.9: Diagrama de red neuronal del sistema de referencia.

2.2.1. Datos de entrada a la red

Como datos de entrada, la red recibe 24 coeficientes MFCC calculados para cada trama de 25ms de audio con un solapamiento de 10ms. Las tramas se normalizan en media y varianza sobre una ventana deslizante de 3s. Las tramas de silencio son eliminadas mediante un sistema VAD (*Voice Activity Detector*) basado en la energía de la señal.

2.2.2. Arquitectura de la red

La arquitectura de la red corresponde al de una red TDNN (Time-Delay Neural Network) y, por tanto, es capaz de analizar las distintas tramas teniendo en cuenta su contexto temporal. Las primeras 5 capas de la red operan trama a trama y cada una de ellas recibe como entrada los patrones de activación de la capa anterior a lo largo de un periodo determinado de tiempo[21].

En la tabla 2.1 se resume la arquitectura completa de la red.

Nombre de la capa	**Contexto de capa**	**Contexto total**	**Entrada x salida**
frame1	{t-2, t+2}	5	120x512
frame2	{t-2, t, t+2}	9	1536x512
frame3	{t-3, t, t+3}	15	1536x512
frame4	{t}	15	512x512
frame5	{t}	15	512x512
stats pooling	[0, T)	T	1500Tx3000
segment6	{0}	T	3000x512
segment7	{0}	T	512x512
softmax	{0}	T	512xN

Tabla 2.1: Arquitectura de la red TDNN (dimensiones)

El contexto temporal de cada una de las 5 capas se suma al de la capa anterior. Por ejemplo, la entrada a la capa *frame3* es la unión de las tramas *t-3*, *t*, *t+3* de la capa *frame2* que han sido calculadas utilizando el contexto temporal de capas anteriores. Por consiguiente, *frame3* dispondrá de un contexto completo de 15 tramas. La capa *stats pooling* une las T tramas que componen el segmento completo. Se tiene un vector de 1500 muestras por cada trama a la entrada de esta capa. A continuación, se calcula la media y la desviación típica a lo largo del tiempo (es decir de las T tramas) de manera que las siguientes capas

operaran sobre el segmento completo. En la tabla 2.1 esto se expresa como un contexto de capa de $\{0\}$ y un contexto completo de T. Tras las dos capas que actúan sobre el segmento completo se encuentra una última capa *softmax* que da lugar a un vector de salida de N muestras, correspondiente al número de locutores (*One Hot Encoding*). Todas las no linealidades presentes entre capas en la red son unidades lineales rectificadas (ReLUs). La red se entrena para clasificar N locutores, con muestras de audio de unos 3 segundos y una etiqueta que identifica al locutor.

El objetivo de la red es producir vectores que generalicen bien para locutores que no hayan formado parte del conjunto de entrenamiento. Para ello, es necesario capturar las características de los locutores a lo largo del audio completo y no de una sola trama. Esto implica que la salida de cualquiera de las capas situadas tras *stats pooling* podría utilizarse para extraer el vector final. La capa *segment6* es inmediata al cálculo de las estadísticas y la capa *segment7* se aplica a la salida de una capa ReLU y por tanto su salida es el resultado de una función no lineal sobre las estadísticas calculadas. Se escoge la salida de *segment6* de forma que no es necesario calcular *segment7* ni la función softmax una vez se ha entrenado la red. Excluyendo estas dos últimas capas, la red cuenta con un total de 4.2 millones de parámetros.

2.2.3. Clasificador PLDA

Finalmente, para clasificar los audios, se utiliza un clasificador PLDA, *Probabilistic Linear discriminant Analysis*. En primer lugar, se centran los vectores y se reduce su dimensión utilizando LDA, *Linear Discriminant Analysis*. A continuación, se normaliza la longitud de las representaciones y se modelan utilizando PLDA.

PLDA proporciona un modelo probabílistico capaz, no sólo de extraer las características de los vectores, si no de combinarlas para lograr el reconocimiento. Las variables que conforman el modelo de PLDA representan tanto la clase del objeto como al objeto dentro de esa clase. Además, PLDA puede utilizarse para reconocer clases que no

han formado parte del entrenamiento[22]. Esta herramienta proporciona un valor relacionado con la probabilidad de que dos muestras pertenezcan a la misma clase.

2.2.4. Normalización de las puntuaciones

Por último, se normalizan las puntuaciones con las que se determinará la verificación utilizando la normalización *s-norm*[23]. La normalización de puntuaciones en la verificación de locutores consiste en transformar las puntuaciones de verificación para aumentar la efectividad del umbral de detección. Para ello, se alinean las distribuciones de las puntuaciones de los distintos locutores. Esta normalización se puede utilizar para reducir los efectos tanto dependientes como independientes del locutor en la señal de voz.

2.2.5. Evaluación del sistema

El sistema se evalúa en función del EER, *Equal Error Rate*, que es una medida de rendimiento muy utilizada en reconocimiento automático de locutores[24]. Para calcularlo, en primer lugar se utilizan las distribuciones de las puntuaciones que permiten computar la tasa de rechazo erróneo (FRR, *False Rejection Rate*) y la tasa de falso positivo (FAR, *False Acceptance Rate*). El EER es el valor para el cual el valor de FRR es igual al valor de FAR.

2.3. CHAIN (Characterizing Individual Speakers) corpus

El corpus CHAIN, está desarrollado específicamente para facilitar las tareas de reconocimiento de locutores. Esta base de datos cuenta con grabaciones de 36 locutores, que utilizan registros del habla distintos y bien definidos: habla neutra, sincronizada con un co-locutor, sincronizada y de forma repetitiva, susurrada y a un ritmo rápido. En este trabajo se utilizan los audios de habla susurrada y neutra. Los locutores utilizan distintos

dialectos del inglés, 28 de los locutores (14 mujeres y 14 hombres) provienen de el Este de Irlanda; los 8 restantes (4 mujeres y 4 hombres) son de Estados Unidos o Reino Unido. Por cada locutor, disponemos de 37 grabaciones de habla susurrada y otras 37 de habla neutra. Cuatro de las 37 grabaciones son fragmentos de fábulas y por tanto tienen una duración larga (entre 30 segundos y 1 minuto), las restantes son frases cortas de menor duración (enre 2 y 5 segundos)[8]. En total se utilizan en este trabajo 1333 muestras de habla susurrada y otras 1333 muestras de habla neutra.

2.4. Técnicas estudiadas

A continuación se enumeran una serie de técnicas propuestas por distintos autores para mejorar los resultados de los sistemas de verificación de locutores en condiciones de voz susurrada. El mayor desafío al que se enfrentan estas técnicas es la falta de datos de voz susurrada en comparación con la gran cantidad de datos de voz neutra[6, 7]. Es por ello de gran importancia explotar las similitudes entre ambos dominios manteniendo a su vez la diferenciación entre locutores. Las técnicas están clasificadas según los datos a los que son aplicadas: datos de entrada a la red neuronal descrita en 2.2, datos de salida de la red y puntuaciones finales.

2.4.1. Técnicas aplicadas a los datos de entrada a la red (Front-end)

El procesado y la extracción de características de la señal de voz puede adaptarse para que los dominios de voz susurrada y voz neutra se acerquen lo máximo posible permitiendo reconocer a un locutor que ha utilizado ambos registros[4, 1]. Para ello es necesario tener en cuenta tanto las peculiaridades que definen a un locutor y que han de compartir ambos dominios como las diferencias que existen entre ellos. Si lo que se pretende es la mejora del sistema base propuesto en 2.2, las técnicas basadas en la recogida de datos diferentes a los

descritos en 2.2.1 no serán compatibles o deberá desarrollarse un sistema paralelo como en [16].

2.4.1.1. Caracterización mediante los parámetros de timbre

Muchas son las características de la voz que permiten modelar a un locutor, sin embargo, sólo algunas de ellas son realmente apropiadas para identificarlo en un conjunto. Algunos autores[25, 21] proponen utilizar LDA para disminuir el número de características, pero se demostró que sus resultados empeoraban en audios ruidosos. El susurro se asemeja mucho a una fuente de ruido como se mencionó en 2.1 por lo que utilizar LDA no es apropiado.

[25] propone un sistema en el que se extraen y concatenan las características más relevantes para identificar a un locutor en un vector, y posteriormente se utiliza un clasificador KNN*(K-Nearest Neighbors)* para catalogarlas. Las funciones de distancia más comúnmente utilizadas en este tipo de clasificador son la distancia Euclídea y la distancia Manhattan[26]. En la figura 2.10 se muestra el diagrama de bloques del sistema propuesto[25].

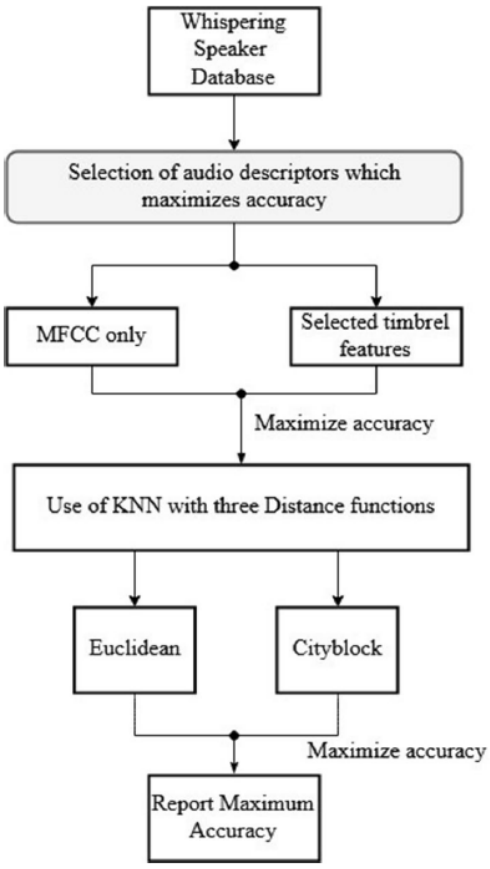


Figura 2.10: Diagrama de bloques del sistema de identificación de locutores.

En primer lugar, es necesario seleccionar las características que serán realmente relevantes en la identificación ya que se ha demostrado que añadir características irrelevantes en la identificación puede incluso empeorar los resultados[27]. Para seleccionarlas se utiliza un algoritmo de selección híbrido[28] que escoge de forma iterativa entre 20 descriptores de audio aquellos que maximizan el rendimiento del sistema. Los descriptores que demuestran mejorar la precisión del sistema de acuerdo al algoritmo son: factor de Roll-off, ratio de cruce por cero (ZCR), *brightness* (referido a la cantidad de alta frecuencia), *roughness* (relacionado con el batido de dos sinusoides próximas en frecuencia) e irregularidad (variación entre los picos del espectro). Estas características son propiedades del timbre, un atributo complejo del sonido que trata de reflejar una realidad subjetiva del mismo. Estos cinco parámetros junto con el primer coeficiente MFCC, conforman el vector de características con el que se

determina mediante el clasificador KNN qué muestras pertenecen al mismo locutor.

En el estudio se compara el rendimiento del sistema utilizando solo los coeficientes MFCC (extraidos con 13 filtros de Mel) y el que consigue utilizando las características de timbre, consiguiendo con estas últimas los mejores resultados. El clasificador KNN logra una mayor precisión utilizando la distancia conocida como *Cityblock*, en vez de la distancia euclídea.

2.4.1.2. Detector de susurro y filtrado independiente de los sonidos consonánticos con lfcc y efcc

Varios estudios[14, 6, 29] sugieren que las diferencias entre la voz neutra y la voz susurrada son menos acusadas para los sonidos consonánticos, que no tienen contenido harmónico pero sí mantienen información relevante del locutor. Modelar la información del locutor mediante un modelo GMM (*Gaussian Mixture Model*) entrenado exclusivamente con las muestras consonánticas de los audios, permite mejorar los resultados en un sistema entrenado con voz neutra y testeado con voz susurrada[16]. Sin embargo, los resultados empeoran cuando en el entrenamiento y el test se utilizan audios pertenecientes al mismo dominio, ya que se pierde mucha información harmónica.

Por otro lado, el procesado de la voz mediante MFCC es conveniente al analizar voz neutra ya que estos coeficientes se basan en la escala de Mel que pretende modelar la percepción humana del sonido[17]. Estos coeficientes enfatizan las bajas frecuencias sobre las altas frecuencias[16]. Cuando se trata de modelar la voz susurrada, donde los formantes se desplazan hacia las altas frecuencias[18], es más apropiado utilizar coeficientes extraídos mediante bancos de filtros exponenciales o lineales (EFCC y LFCC) que dan mayor importancia a las altas frecuencias[30, 31].

La pérdida de prestaciones del sistema en condiciones de susurro no sucede de la misma forma en la identificación de todos los locutores. Algunos de los locutores obtienen mejores resultados que otros. Las similitudes en la información del locutor en los dominios de voz susurrada y voz neutra dependen de como el locutor produce el susurro[16]. Identificar las

tramas que contienen información más relevante para la identificación del locutor y separarlas de aquellas donde la información aparece más distorsionada, puede mejorar el rendimiento del sistema[16], ya que utilizar técnicas de compensación sobre tramas donde la identificación es inmediata empeora los resultados.

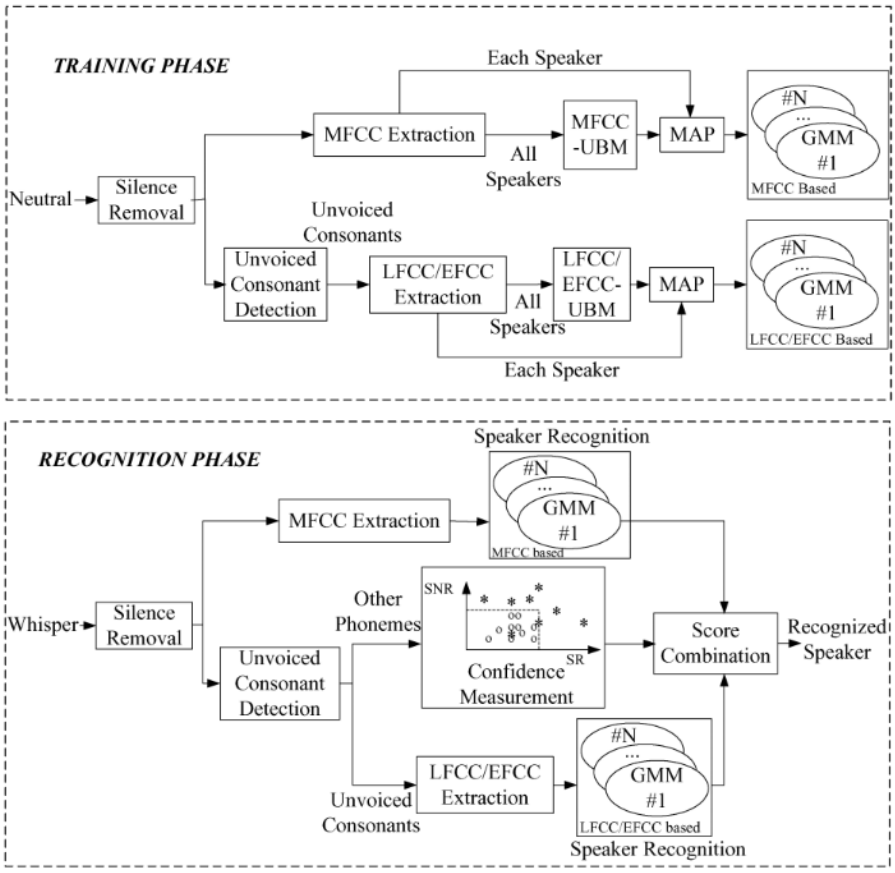


Figura 2.11: Diagrama de flujo del sistema de identificación basado en GMM entrenadas con voz neutra.

El sistema desarrollado por [16] (Figura 2.11), diferencia las tramas que necesitan compensación de aquellas que no la necesitan y establece una medida de confianza que permite decidir si aplicar el procesado habitual de las tramas de voz neutra (MFCC) o el procesado de los sonidos consonánticos mediante LFCC o EFCC. Para diferenciar estas tramas, se basa en la relación señal a ruido (SNR), la pendiente espectral y el ratio espectral $r_{f1-2vs1-8}$. Este ratio trata de representar la distribución frecuencial de la señal mediante la relación entre el porcentaje espectral de la banda de 1000Hz-2000Hz y la

banda de 1000Hz-8000Hz. Este sistema propone por tanto la existencia de dos modelos distintos cuyo resultado tiene mayor o menor peso en la decisión final según el criterio de 'calidad' de la trama.

2.4.1.3. Conversión de voz susurrada a voz neutra con el sistema seq2seq

Es posible caracterizar la relación no lineal que existe entre las tramas de voz susurrada y las tramas de voz neutra[7]. Para ello, [7] utiliza un sistema conocido como *sequence-to-sequence*, que codifica un conjunto de datos fuente a otro de datos objetivo. La estructura de este sistema es como la de un codificador que mapea los datos fuente a un espacio Eigen de un gran número de dimensiones para ser posteriormente decodificado. El codificador, lee las tramas de voz susurrada y acumula esta información en un vector de contexto, que caracteriza toda la información de la secuencia. El decodificador toma el vector de contexto y genera las tramas de voz neutra[7]. Una de las ventajas de este tipo de sistemas es que no necesitan que los datos de entrada y salida tengan la misma longitud, lo que implica que no es necesario alinear las tramas a la entrada del codificador[32, 7].

En las señales de voz, las tramas sucesivas se encuentran muy relacionadas entre ellas. Para caracterizar esta relación se utiliza como codificador una red LSTM, *Long-Short Time Memory*[33]. Por la misma razón, se utiliza una red LSTM para decodificar los datos de contexto a las tramas de voz neutra objetivo[34]. Este tipo de arquitectura permite modelar secuencias temporales y sus dependencias a lo largo del tiempo de forma precisa[35]. La red LSTM propuesta cuenta con una puerta de entrada, *input gate*, una puerta de salida, *output gate*, y una puerta intermedia, *forget gate*. Dadas las tramas de voz susurrada $\{x_1, x_2...x_{T_X}\}$ y las tramas de voz neutra $\{y_1, y_2...y_{T_y}\}$, la red se estructura de la siguiente manera:

$$i_t = \delta(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{2.1}$$

$$p_t = f_t p_{t-1} + i_t * \tanh(W_p \cdot [h_{t-1}, x_t] + b_p) \tag{2.2}$$

$$f_t = \delta(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{2.3}$$

$$o_t = \delta(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{2.4}$$

$$h_t = o_t \tanh(p_t) \tag{2.5}$$

Donde i,f, o y p hacen referencia a *input gate*, *forget gate*, *output gate* y el estado de la célula de la red en t, respectivamente. La letra δ representa la función sigmoide y h_t el estado oculto de la red en t.

Como entrada, cada célula de la red recibe unos datos de entrada x_t y unos datos de contexto compuestos por el vector de estado p_{t-1} y el vector oculto h_{t-1} de la célula anterior. A la salida de cada célula de la red se tiene tanto unos datos de salida o_t, como los datos de contexto, el llamado estado oculto h_t y el vector de estado p_t. Un sistema *sequence-to-sequence convencional*, adopta como vector de contexto inicial del decodificador, el último vector oculto calculado en el codificador y un estado inicial S_o. El problema de este tipo de sistemas es que es muy difícil resumir toda la información de la secuencia en un sólo vector de contexto para que el decodificador sea capaz de sintetizar las tramas objetivo. Además, las secuencias de voz susurrada son más largas por lo general que las de voz neutra y las tramas no se encuentran alineadas temporalmente[7]. Por ello, para modelar de forma más fiable las relaciones no lineales entre las tramas de voz susurrada y las tramas de voz neutra, se utilizan mecanismos de atención. Con esto se consigue obtener un vector de contexto adaptativo con el que se estima en cada instante de tiempo el estado y la salida del decodificador. En el sistema propuesto se asume que el estado S_t del decodificador esta relacionado con todos los vectores ocultos del codificador con una mayor o menor relevancia. Para calcular el contexto c_t se tienen en cuenta los estados ocultos anteriores y posteriores a t del codificador.

Si $\{h_1, h_2 ... h_{T_X}\}$ son los estados ocultos en cada instante de tiempo del codificador, se

obtiene c_i, el vector de contexto en el instante de tiempo i, como:

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j \tag{2.6}$$

$$\alpha_{ij} = \frac{exp(e_{ij})}{\sum_{j=1}^{T_x} exp(e_{ij})} \tag{2.7}$$

$$e_{ij} = a(S_{i-1}, h_j) \tag{2.8}$$

$a = v^T \tanh(WS_{i-1} + Uh_j)$ describe la relación que existe entre el estado oculto del codificador h_j y el estado de la célula anterior del decodificador S_{i-1}. El nuevo estado del decodificador se calcula entonces como:

$$(cell\_out, S_i) = lstm(y_{i-1}, S_{i-1}, c_i) \tag{2.9}$$

y_{i-1} es el vector de características de la trama anterior, S_{i-1} es el estado de la célula anterior LSTM y c_i es el contexto para la trama actual del codificador. Finalmente, la trama decodificada es el resultado de:

$$y_i = liner(cell\_out, c_{i+1}) \tag{2.10}$$

La figura 2.12 muestra de forma gráfica como se decodifica cada trama y_t que compone la secuencia completa, mediante el vector de contexto adaptativo.

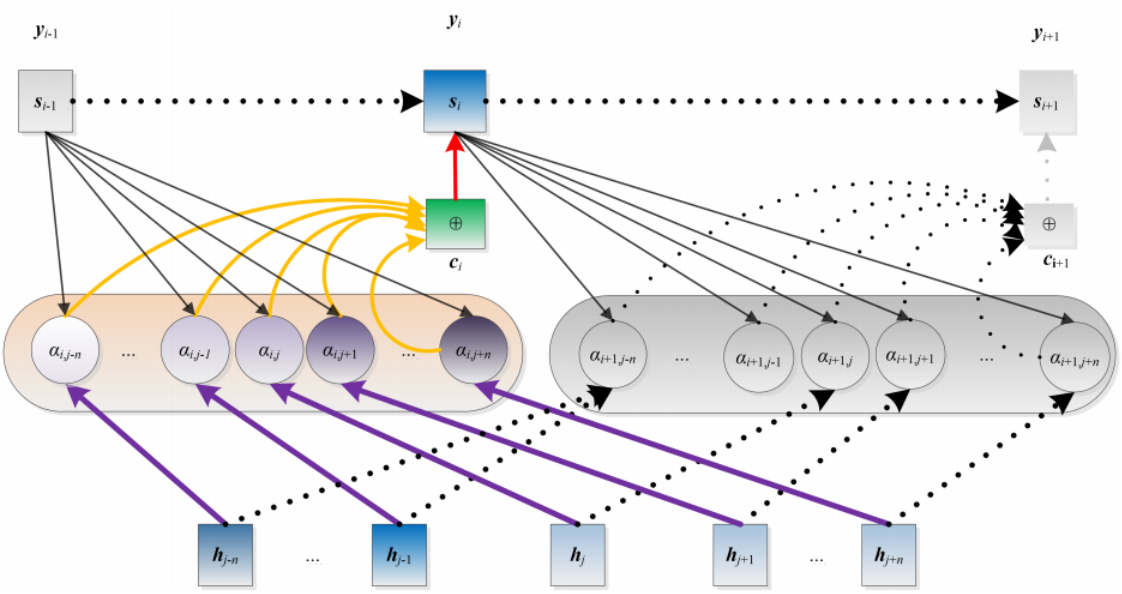


Figura 2.12: Generación del vector de contexto adaptativo del modelo seq2seq

2.4.2. Técnicas aplicadas a los vectores de salida de la red (Back-end)

El bloque principal del sistema de reconocimiento es la red neuronal que se describe en 2.2 capaz de convertir las muestras de los audios de longitud variable en vectores de longitud fija. Las características de los audios tanto de voz neutra como de voz susurrada quedan representadas por esos vectores de 512 muestras. Para facilitar la tarea del clasificador final, se proponen técnicas que establezcan una relación entre los vectores de voz susurrada y los de voz neutra para lograr acercar los dominios. Este tipo de técnicas se conocen como técnicas de compensación o normalización. La ventaja de este tipo de técnicas es que, aunque pueden llegar a ser menos precisas o específicas que las adaptaciones del modelo vistas en 3.3, necesitan menos datos para funcionar correctamente[36]. Ambas técnicas propuestas en esta sección pueden aplicarse a los datos de entrada de la red, como se verá en la sección 3.3.4.

2.4.2.1. Multi-Environment Model-based LInear Normalization (MEMLIN) aplicado a los vectores de salida

Dentro de las técnicas de compensación, destacan los algoritmos basados en el cálculo del MMSE, *Minimum Mean Square Error*. Entre ellos encontramos la técnica conocida como MEMLIN, *Multi-Environment Models based LInear Normalization*, que introduce un factor de corrección que depende de dos modelos GMM, limpio y ruidoso, y de la probabilidad condicional del modelo limpio en base al modelo ruidoso y a una muestra ruidosa[37].

Dado un vector limpio x y un vector ruidoso y, el estimador MMSE obtiene el vector limpio estimado $\hat{x}$:

$$\hat{x} = E\left[x|y\right] = \int_x xp(x|y)dx \tag{2.11}$$

Las distintas técnicas basadas en este estimador, tratan de aproximar la función de densidad de probabilidad $p(x|y)$ dados x e y[37]. El modelo de MEMLIN asume que el vector ruidoso obedece a un modelo diferente de mezcla de Gaussianas para cada uno de los entornos ruidosos que van a modelarse:

$$p_e(y) = \sum_{s_y^e} p(y|s_y^e)p(s_y^e) \tag{2.12}$$

$$p(y|s_y^e) = N(y; \mu_{s_y^e}, \Sigma_{s_y^e}) \tag{2.13}$$

Donde e representa el índice del entorno que se esta modelando y s_y^e la Gaussiana correspondiente al entorno e y a la que se asocian los parámetros de vector de media $\mu_{s_y^e}$, matriz de covarianza $\Sigma_{s_y^e}$ y peso $p(s_y^e)$.

El modelo de MEMLIN asume que el vector limpio sigue un modelo de mezcla de Gaussianas:

$$p(x) = \sum_{s_x} p(x|s_x)p(s_x) \tag{2.14}$$

$$p(x|s_x) = N(x; \mu_{s_x}, \Sigma_{s_x}) \tag{2.15}$$

Donde s_x representa la Gaussiana del modelo limpio y a la que se asocian los parámetros de vector de media μ_{s_x}, matriz de covarianza Σ_{s_x} y peso $p(s_x)$.

MEMLIN aproxima entonces la función de densidad de probabilidad de x dado y, s_y^e, s_x, como una mezcla de Gaussianas cuya matriz de covarianzas Σ_{s_x,s_y^e} depende de s_x y s_y, y cuyo vector de media depende de s_y^e, s_x y α_e que es el peso asociado al entorno ruidoso e. El vector de transformación r_{s_x,s_y^e} representa la diferencia entre los datos limpios y ruidosos dados los el modelo limpio s_x y el ruidoso s_y^e del entorno e.

$$p(x|y, s_y^e, s_x) = N(x; y - \sum_e \alpha_e r_{s_x,s_y^e}, \Sigma_{s_x,s_y^e}) \tag{2.16}$$

Si se estima x como la media de (2.16), se aproxima (2.11) por:

$$\hat{x}_t \simeq y_t - \sum_{s_x} \sum_e \sum_{s_y^e} \alpha_{e,t} r_{s_x,s_y^e} p(s_y^e|y_t) p(s_x|s_y^e, y_t) \tag{2.17}$$

Donde t es el índice temporal, $p(s_y^e|y_t)$ es la probabilidad del modelo ruidoso s_y^e dado la trama ruidosa y_t, y $p(s_x|s_y^e, y_t)$ es la probabilidad del modelo limpio s_x dados el modelo ruidoso s_y^e y la trama ruidosa y_t. Para poder calcular $\hat{x}_t$ es necesario estimar los parámetros del modelo:

1. El peso del entorno ruidoso e, $\alpha_{e,t}$, y la probabilidad del modelo s_y^e, $p(s_y^e|y_t)$, deben calcularse para cada instante de tiempo t dada la trama ruidosa y_t que se desea normalizar.

2. Los vectores de transformación r_{s_x,s_y^e} y la probabilidad $p(s_x|s_y^e, y_t)$ para cada entorno ruidoso e deben calcularse mediante una fase de entrenamiento y con un conjunto de datos de entrenamiento.

El cálculo de $\alpha_{e,t}$ sigue un proceso iterativo que utiliza la trama ruidosa de cada instante t:

$$\alpha_{e,t} = \beta \cdot \alpha_{e,t-1} + (\beta - 1) \cdot \frac{p_e(y_t)}{\sum_e p_e(y_t)} \tag{2.18}$$

donde β es una constante de memoria y $\alpha_{e,0}$ se considera uniforme para todos los entornos. Utilizando la ecuación (2.13) y el teorema de Bayes, se calcula $p(s_y^e|y_t)$ como:

$$p(s_y^e|y_t) = \frac{p(y_t|s_y^e)p(s_y^e)}{\sum_{s_y^e} p(y_t|s_y^e)p(s_y^e)} \tag{2.19}$$

Dado un conjunto de entrenamiento compuesto por $X_e = \{x_1^e, x_2^e ... x_{T_e}\}$ vectores limpios de características y $Y_e = \{y_1^e, y_2^e ... y_{T_e}\}$ vectores ruidosos, pueden obtenerse los vectores de transformación r_{s_x,s_y^e} por medio del algoritmo de máxima verosimilitud, *Maximum Likelihood (ML)*. La función de maximización será:

$$L(Y_e) = \sum_{t_e} \log(\sum_{s_y^e} p(s_y^e) N(y; \mu_{s_y^e} + r_{s_x,s_y^e}, \Sigma_{s_x,s_y^e})) \tag{2.20}$$

El algoritmo de estimación de máxmima verosimilitud, *Expectation Maximization (EM)* permite resolver la ecuación (2.20):

$$r_{s_x,s_y^e} = \frac{\sum_{t_e} p(s_x|x_{t_e}^e)p(s_y^e|y_{t_e}^e)(y_{t_e}^e - x_{t_e}^e)}{\sum_{t_e} p(s_x|x_{t_e}^e)p(s_y^e|y_{t_e}^e)} \tag{2.21}$$

Finalmente, la probabilidad condicional $p(s_x|s_y^e, y_t)$ puede estimarse con la frecuencia relativa (solución *hard*) de los datos de entrenamiento. Para cada par de vectores (limpio y ruidoso) del conjunto de entrenamiento, se obtiene el par de Gaussianas más probable. Tras ello, se obtiene la probabilidad del modelo entre Gaussianas utilizando la solución *hard* como:

$$p(s_x|s_y, y_t) = \frac{C_N(s_x|s_y^e)}{N} \tag{2.22}$$

donde $C_N(s_x|s_y^e)$ es el número de veces que el par de Gaussianas s_x y s_y^e es el más probable. N es el número de veces que la Gaussiana más probable para el vector ruidoso es s_y^e. La

probabilidad del modelo entre Gaussianas utilizando la solución *soft* se calcula:

$$p(s_x|e, s_y^e) = \frac{\sum_{t_e} p(x_{t_e}^{T_{r,e}}|s_x)p(y_{t_e}^{T_{r,e}}|s_y)p(s_x)p(s_y^e)}{\sum_{t_e}\sum_{s_x} p(x_{t_e}^{T_{r,e}}|s_x)p(y_{t_e}^{T_{r,e}}|s_y)p(s_x)p(s_y^e)} \tag{2.23}$$

Cuando no se dispone de una gran cantidad de datos, la solución *soft* proporciona una solución más estable.

2.4.2.2. Linear Multivariate Regression (LMR) aplicado a los vectores de salida

Una de las estrategias propuestas por [38] es realizar una transformación espectral utilizando una herramienta de análisis estadístico conocida como LMR, *Linear Multivariate Regresion*. Con ello consigue modelar la voz de un locutor para que sea reconocido como otro distinto. De nuevo, tratan de acercarse los dominios de susurro y voz neutra mediante un mapeado estadístico de los datos.

En primer lugar, es necesario alinear los datos de susurro y voz neutra utilizando el algoritmo de DTW, *Dynamic Time Warping*. A continuación, el algoritmo utiliza una fase de entrenamiento en la que se divide el espacio acústico de los locutores de referencia mediante un algoritmo no supervisado de *clustering* para reducir la complejidad del mapeo[39]. De esta forma, el mapeo queda aproximado por un conjunto de transformadas elementales, cada una asociada a una de las clases en las que se ha dividido el espacio. La base de esta estrategia es que las transformaciones dependerán de la naturaleza fonética del sonido. El número de clases tiene que ser lo suficientemente alto como para que puedan diferenciarse los distintos contextos fonéticos de la base de datos. La figura 2.13 muestra un diagrama de bloques del sistema propuesto[38].

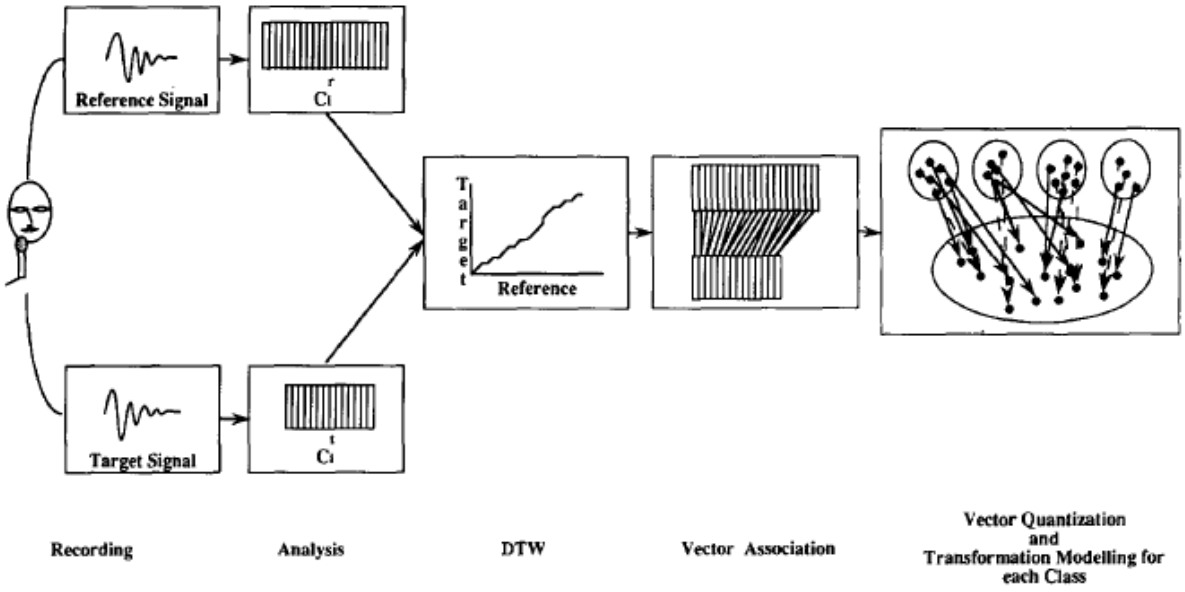


Figura 2.13: Diagrama de bloques de la fase de entrenamiento de la transformación LMR

Siendo $C^{r,q} = \{C_j^{r,q}\}_{j=1}^{M_q}$ el conjunto de vectores espectrales de referencia que pertenecen a la *q-ésima* clase y Q, M_q el número total de vectores de esa clase, asociados a ellos mediante DTW existirá otro conjunto de vectores objetivo representado por $C^{t,q} = \{C_j^{t,q}\}_{j=1}^{M_q}$. El algoritmo de LMR trata de encontrar una transformación lineal, a través de la matriz P_q que minimice el error cuadrático medio E entre los vectores de referencia y objetivo.

Dadas las medias empíricas del componente *j-ésimo* de los vectores espectrales de referencia y objetivo, $m_j^{r,q}$ y $m_j^{t,q}$ respectivamente:

$$m_j^{t,q} = \frac{1}{M_q} \sum_{k=1}^{M_q} C_k^{r,q}(j) \tag{2.24}$$

Los vectores normalizados se obtendrán a través de la transformación lineal:

$$\tilde{C}_k^{r,q} - m_k^{r,q} \tag{2.25}$$

La matriz P_q minimiza el error cuadrático medio entre ambos conjuntos de vectores

normalizados y por tanto pretende dar solución al problema de minimización:

$$\sum_{k=1}^{M_q} ||\tilde{C}_k^{t,q} - P_q \tilde{C}_k^{r,q}||^2 \tag{2.26}$$

La solución al problema de minimización (2.26) es:

$$P_q = \tilde{C}^{t,q} \tilde{C}^{r,q^\dagger} \tag{2.27}$$

donde † representa la pseudo-inversa de $\tilde{C}^{r,q}$.

Para obtener el vector objetivo tras la fase de entrenamiento donde se calcula P_q, es necesario normalizar primero el vector que pretende transformarse substrayendo la media $m_j^{r,q}$ que se ha calculado en la fase de entrenamiento. El vector transformado se obtiene multiplicando el normalizado por la matriz P_q y este ha de ser desnormalizado añadiendo la media $m_j^{t,q}$ calculada.

Esta técnica muestra buenos resultados en la conversión de voz hablada de un registro a otro. El estudio puede ser interesante en la transformación de la voz susurrada a la voz neutra pese a que el objetivo de este autor es la transformación de la voz de un locutor a la de otro distinto. Aunque en [38] la técnica se aplica a los datos de voz, en el presente trabajo se aplica a los datos de salida de la red, como se verá en la sección 3.4.3.

2.4.3. Técnicas aplicadas a las puntuaciones

Tal y como se describe en la sección 2.2, tras obtener los vectores de salida de la red, se calculan las puntuaciones entre pares de muestras (la de registro en el sistema y la que quiere verificarse) con las que se determina la aceptación o rechazo del usuario en el sistema.

2.4.3.1. Calibración de puntuaciones

Dada la degradación que sufre el sistema cuando se presentan muestras de voz susurrada, [40] propone informar al sistema cuando esto ocurre y actuar en consecuencia. Los valores de EER obtenidos para las comparaciones de muestras pertenecientes al mismo dominio (*neutra-neutra* y *susurro-susurro*) son notablemente más altos que los obtenidos cuando los dominios difieren (*neutra-susurro*).

El sistema propuesto por [40] parte por tanto de un detector de susurro capaz de clasificar las muestras en *habla normal* y *habla neutra*. Esta información se utiliza posteriormente para calibrar las puntuaciones obtenidas a la salida del sistema. La calibración de un sistema de reconocimiento de locutores puede verse como la transformación de los valores de las puntuaciones de salida en ratios *LLR (Log Likelihood Ratio)* calibrados[41]. Un sistema calibrado puede utilizarse en escenarios muy diversos sin que sea necesario cambiar el umbral de decisión[42]. Habitualmente los sistemas se calibran mediante el escalado y desplazamiento de las puntuaciones utilizando para ello una serie de parámetros obtenidos por el algoritmo de optimización de regresión logística LR, *Logistic Regression*[43]. La distribución de las puntuaciones de cada condición de comparación de voz neutra y voz susurrada es muy diferente. Por ello los parámetros ideales para cada una de las condiciones difieren y es necesario modelar y calibrar cada condición por separado. Se tendrán por tanto tres modelos LR diferentes según la condición de comparación: *neutra-neutra*, *susurro-susurro* y *neutra-susurro*.

La calibración de las puntuaciones permite grandes mejoras en el rendimiento del sistema, para que funcione correctamente es necesario disponer de un detector de voz susurrada que sea fiable[40].

Capítulo 3

Resultados

3.1. Preprocesado de los datos

El preprocesado de los datos permite mejorar el rendimiento del sistema y facilita el análisis de los mismos y la aplicación de las técnicas estudiadas.

En primer lugar, todos los audios deben ser muestreados con la misma frecuencia. Los audios que conforman el corpus descrito en 2.3 presentan una frecuencia de muestreo de $44,1Khz$. La frecuencia de muestreo de los audios con los que fue entrenado el sistema de referencia es de $16Khz$. Los audios que van a ser clasificados por este sistema deben presentar la misma frecuencia de muestreo que aquellos con los que fue entrenado para que se correspondan las características de las muestras "aprendidas" por la red neuronal. La herramienta *sox* se utiliza para cambiar con facilidad la frecuencia de muestreo del dataset. Por otro lado, las tramas de silencio que pueden presentar los audios de muestra introducen ruido en el sistema ya que no aportan información acerca del locutor. Para eliminarlas, se utiliza un detector de actividad de voz VAD *(Voice Activity Detector)* desarrollado en Python. Uno de los parámetros de configuración de este detector es el umbral de potencia a partir del cual se considera que la trama es una trama de silencio. Es importante ajustar correctamente este parámetro, el ajuste se realiza de forma independiente para los audios

de voz neutra y de susurro, ya que la potencia de la voz susurrada es menor y por tanto un umbral alto podría eliminar tramas de información relevante. El nivel de potencia elegido para voz neutra es $-25db$ mientras que para voz susurrada es de $-35db$.

En este trabajo se ha utilizado el lenguaje de programación *Python* para extraer los descriptores y características de audio como los coeficientes MFCC y para desarrollar los modelos básicos de datos como GMM o KNN. Además, se han utilizado algunas funciones de librerías como *librosa* (análisis de audios) y *scikit learn* (librería de aprendizaje automático) desarrolladas en *Python*. Por otro lado, la herramienta Kaldi proporciona algunas funciones de utilidad como son el cálculo de los coeficientes MFCC, el cálculo y substracción de la media y el cómputo de las puntuaciones y el EER, desarrolladas en C++.

3.2. Resultados iniciales

Para evaluar el sistema de referencia, se utilizan muestras del corpus presentado en 2.3. El valor de EER se calcula en función de un conjunto de comparaciones entre pares de muestras de audio. En un escenario real, una de ellas sería la muestra registrada en la base de datos y la otra, la muestra tomada para la verificación. Para poder cuantificar la pérdida de prestaciones del sistema cuando se enfrenta a distintas condiciones de esfuerzo vocal, se utilizan distintos grupos de comparaciones:

- En primer lugar se comparan pares de muestras de voz neutra (*neutra - neutra*). Este conjunto de comparaciones permite establecer como funciona el sistema de referencia en condiciones de locutor cooperativo, cuando se clasifican muestras de una base de datos distinta a la de entrenamiento. El valor de EER obtenido para la comparación *neutra-neutra* es muy bajo y mejorarlo no forma parte del objetivo del proyecto. El corpus cuenta con un total de 1332 muestras de voz neutra, se comparan todas las muestras distintas entre sí por lo que se tienen 1.772.892 comparaciones.

- En segundo lugar, se comparan pares de muestras de voz susurrada (*susurro - susurro*).

En este caso, el sistema se enfrenta a un esfuerzo vocal distinto al que caracteriza al conjunto de datos con el que fue entrenado. El sistema se degrada notablemente, sin embargo, cuenta con la ventaja de que tanto la muestra de registro como la de verificación, pertenecen al mismo dominio. De nuevo se tienen 1.772.892 comparaciones entre las 1332 muestras distintas de audio de voz susurrada.

- El tercer grupo de comparaciones reúne aquellas en las que las muestras difieren en la condición de esfuerzo vocal (*neutra* - *susurro*). La degradación del sistema en este caso es mayor debido a las diferencias entre los registros analizadas en el trabajo. El procesado de los datos a la entrada de la red tendrá como objetivo paliar esta pérdida de prestaciones. Cuando los dominios difieren el sistema sufre un aumento del EER del 15 %. La combinación de las 1332 muestras de voz neutra y las 1332 muestras de voz susurrada da lugar a un total de 1.774.224 comparaciones.

- Finalmente se agrupan los tres tipos de comparaciones mencionados previamente y se calcula el EER conjunto (*todos* - *todos*). Este grupo de comparaciones refleja el resultado del sistema completo, que en producción se enfrentará a los tres tipos de comparaciones. Este grupo de comparaciones cobrará especial interés en la calibración de puntuaciones como se verá en 3.5.1. La suma de los tres grupos anteriores de comparaciones da lugar a un total de 5.320.008 comparaciones.

En la tabla 3.1 se presentan los resultados obtenidos para los datos del corpus utilizando el sistema de referencia (Sección 2.2). Estos son los resultados de partida y que tratarán de mejorarse con las técnicas propuestas.

Sistema	Comparación	EER %
Baseline	neutra - neutra	1.981 %
	susurro - susurro	7.533 %
	neutra - susurro	17.79 %
	todos - todos	22.47 %

Tabla 3.1: Valor de EER obtenido con los datos del corpus utilizando el sistema de referencia

3.3. Técnicas aplicadas a los datos de entrada a la red (Front-End)

3.3.1. Caracterización mediante los parámetros de timbre

El primero de los sistemas estudiados es el propuesto por [4]. El sistema consiste en la extracción de una serie de descriptores de audio que pretenden caracterizar a un locutor sin error en ambos dominios (susurro y voz neutra). Los descriptores de audio que se han utilizado son los enumerados en la sección 3.3.1:

- *Zero Cross Rate*: mide la tasa de cruce por cero de una señal en el dominio del tiempo. Deacuerdo a [4], es alta para la voz susurrada y baja para la voz neutra.

- *Roll off*: es una medida espectral de audio definida como la frecuencia bajo la cual se encuentra el 85 % de la energía total de la señal. Para un espectro de señal S_t con N muestras, el factor de *Roll off* se calcula como:

$$\sum_{i=1}^{R} S_t\,[n] = 0{,}85 \times \sum_{i=1}^{N} S_t\,[n] \tag{3.1}$$

- *Brightness*: Es una medida de la energía de la señal para frecuencias superiores a una frecuencia de corte f_c. En este trabajo se utiliza una frecuencia de corte de $1000hz$.

- *MFCC*: Para extraer los coeficientes, se utilizan ventanas de Hamming de $25ms$ con un 50 % de solapamiento, se aplica la transformada de Fourier de 512 puntos y se filtra el resultado con 13 filtros de Mel.

- *Roughness*: esta característica pretende estimar la disconformidad sensitiva producida por dos sinusoides batiendo próximas en frecuencia. Para estimarla se localizan los picos del espectro y se calculan las disconformidades (utilizando el algoritmo de Plomp [44]) entre todos los posibles pares de picos.

- *Irregularity*: grado de variación entre los picos del espectro de la señal. En este proyecto se calcula utilizando picos sucesivos del espectro.

$$\frac{\sum_{k=1}^{n} (a_k - a_{k+1})^2}{\sum_{k=1}^{n} a_k^2} \tag{3.2}$$

Todos los descriptores se han calculado utilizando *Python* como lenguaje de programación. La librería *librosa* permite funciones de cálculo de algunos parámetros como son los MFCC y el ZCR. En este caso, como los datos recogidos son distintos a los que emplea el sistema de referencia, la técnica se presenta como un sistema completo y no como una mejora del sistema existente. Para tener una comparación realista de lo que supone utilizar los parámetros de timbre respecto a los coeficientes MFCC, se comparan los resultados de verificación para vectores de 13 coeficientes MFCC (media de los coeficientes para todas las tramas del audio) y para vectores formados por los parámetros de timbre. Cada audio está representado por un vector de 13 o 6 muestras respectivamente.

Una de las principales dificultades en la extracción de las características del timbre para formar el vector, es la normalización de los valores. Para que una característica no cobre mayor importancia que el resto, su rango de valores ha de ser el mismo, es por ello que se necesita normalizar los valores de timbre. Esta tarea no es trivial, puesto que normalizar los valores puede significar una pérdida de información relevante si no se hace de la manera correcta. Para ello, se calcula la media y desviación típica de los valores timbre para todas

las muestras de la base de datos y se normalizan los datos mediante la fórmula:

$$\hat{x_i} = \frac{x_i - \mu_x}{\sigma_x} \tag{3.3}$$

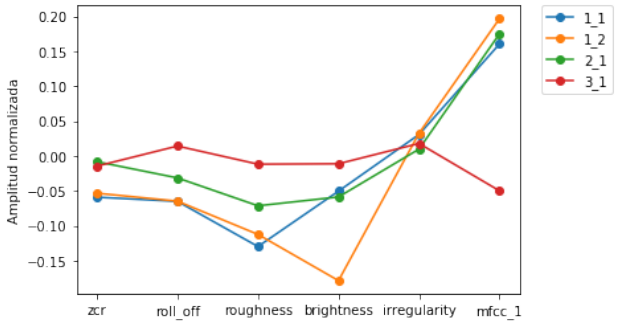
En un escenario real, la media y la desviación típica se calculan con muestras de un conjunto de entrenamiento.

Este experimento se basa en el estudio de [4] que pretende la identificación de locutores mediante un clasificador KNN. El problema que se aborda en este trabajo es la verificación de locutores, por lo que en vez de utilizar un clasificador KNN, se calcula directamente la distancia euclídea entre cada par de muestras y el umbral a partir del cual se confirma o rechaza la verificación. La tabla 3.2 recoge los resultados obtenidos de EER y porcentaje de error en el sistema de extracción de características de timbre y de MFCC.

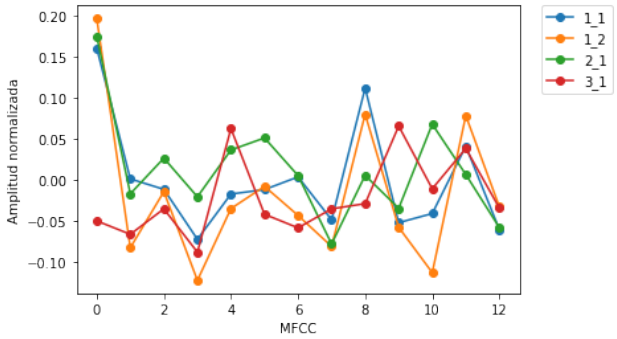
Sistema	**EER %**	**Error %**
Características de timbre	47.03 %	51.6 %
MFCC	48.37 %	50.24 %

Tabla 3.2: Valor de EER y porcentaje de error obtenidos con el sistema de extracción de características de timbre y de MFCC

Ninguno de los sistemas logra clasificar los datos correctamente. El número de características del vector no es suficiente para captar las variabilidades temporales de las muestras.



(a) Características del timbre en tres locutores disintos



(b) Coeficientes MFCC en tres locutores distintos

Figura 3.1: Vectores de características de los sistemas

La figura 3.1 representa los vectores de características correspondientes a la primera muestra de tres locutores distintos (1_1, 2_1 y 3_1) y la segunda de uno de ellos (1_2). Esta figura refleja como incluso muestras de audios del mismo locutor presentan vectores de características muy distintos (1_1 y 1_2). El problema que trata de resolverse es la verificación de locutores sin que las muestras dependan del texto (*text-independent*), por lo que se trata de un problema complejo que necesita un gran número de parámetros para obtener una caracterización precisa del locutor como es el sistema de Kaldi.

3.3.2. Detector de susurro y filtrado independiente de los sonidos consonánticos con LFCC y EFCC

El sistema desarrollado por [16] propone tratar de forma diferente las tramas de voz susurrada, y que estas sean clasificadas por un sistema entrenado con muestras de los sonidos

no vocálicos procesados mediante filtros lineales capaces de dar mayor importancia a las altas frecuencias que los filtros de voz habituales (MFCC). Para poder tratar las tramas de voz susurrada con un sistema paralelo de verificación, es necesario identificar en primer lugar estas tramas mediante un detector de voz susurrada. El detector consiste en dos modelos GMM, uno de ellos entrenado con muestras de voz neutra y otro, entrenado con muestras de voz susurrada. El código del detector se desarrolla en *Python* utilizando la librería *scikit learn* para definir los modelos GMM. Como la cantidad de muestras de las que se dispone es limitada, se utiliza en el entrenamiento la técnica *leave-one-out*, que consiste en utilizar las muestras de todos los locutores menos de uno para entrenar los modelos y evaluar con las muestras del locutor restante. Una vez entrenados los modelos, se calcula la puntuación de la muestra en cada modelo y se clasifica la muestra en función de la puntuación más alta. Cada entrenamiento cuenta con 1295 muestras de voz susurrada y 1295 muestras de voz neutra, y el modelo se testea con 37 muestras de cada uno de los registros.

Sistema	**Número de componentes**	**Error %**
Detector de susurro	16	0.26 %
	32	0.67 %
	64	0.037 %
	128	0.67 %

Tabla 3.3: Porcentaje de error obtenido con el detector de susurro con distinto número de componentes

La tabla 3.3 muestra los resultados del detector con modelos GMM con distinto número de componentes. Los mejores resultados se obtienen utilizando 64 componentes con un error de 0,037 %, el detector es muy fiable.

A continuación, es necesario un sistema capaz de extraer los sonidos consonánticos de las muestras de voz neutra y voz susurrada. Estas muestras se filtran utilizando filtros lineales (LFCC o EFCC) y con ellas se entrena un modelo GMM para cada locutor. Para separar

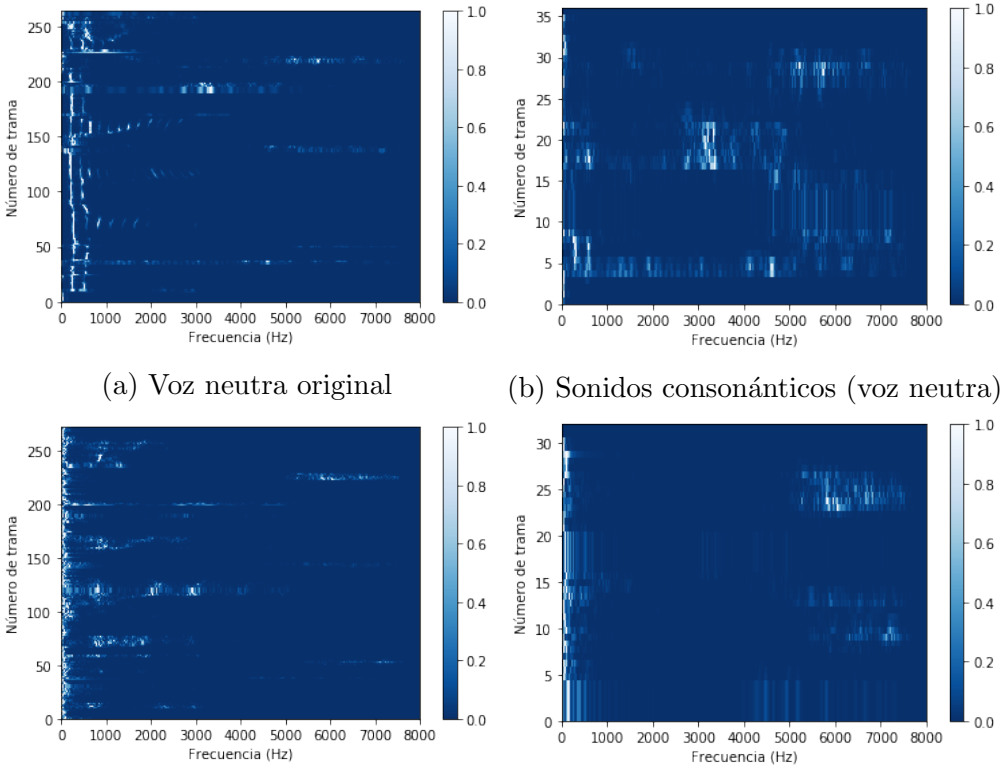
los sonidos consonánticos de los vocálicos se toman tres medidas para cada trama de voz i, la energía total en las bandas de frecuencia: $100-4000Hz\,(f_{i,l})$, $4000-8000Hz\,(f_{i,h})$ y $100-8000Hz\,(f_{i,e})$. Con estas medidas se calculan dos parámetros cuyo valor se utiliza para clasificar la trama[16]:

$$Rn(i) = \frac{f_{i,l}}{f_{i,e}} \tag{3.4}$$

$$En(i) = -P_{h,i-1}log(P_{h,i}) - P_{h,i}log(P_{h,i-1}) \tag{3.5}$$

donde $P_{h,i} = \frac{f_{i,h}}{f_{i,e}}$.

Estas medidas se basan en el hecho de que gran parte de la energía espectral de la señal de los sonidos consonánticos se encuentra en las altas frecuencias, mientras que en el caso de los sonidos vocálicos se centra en frecuencias más bajas. De forma empírica se calcula un umbral máximo de entropía En (0,05) y mínimo de ratio Rn (0,93), que permita extraer las tramas consonánticas.



(a) Voz neutra original

(b) Sonidos consonánticos (voz neutra)

(c) Voz susurrada original

(d) Sonidos consonánticos (susurro)

Figura 3.2: Periodograma de la señal de voz

La figura 3.2 muestra el periodograma de la señal de voz original y el de los sonidos consonánticos extraídos. Esta figura muestra como, en el caso de la voz neutra, desaparecen las líneas horizontales de los formantes al eliminar los sonidos vocálicos y como la energía se desplaza a frecuencias más altas.

El sistema consiste en modelar una mezcla de Gaussianas (GMM) de 64 componentes para cada uno de los locutores. Estos modelos de Gaussianas se entrenan con las muestras de voz neutra y posteriormente, se calcula el modelo más probable para cada una de las muestras de voz susurrada. De la misma forma que el detector de susurro, el sistema se desarrolla en *Python* utilizando las librerías de *scikit learn* para los modelos GMM y *librosa* para la extracción de los coeficientes MFCC.

Sistema	**EER %**
Baseline	17.79 %
MFCC	42.99 %
MFCC (sonidos consonánticos)	40.73 %
LFCC (sonidos consonánticos)	38.06 %

Tabla 3.4: Comparativa del valor de EER resultado de los modelos GMM utilizando los coeficientes LFCC y MFCC (escenario *neutra - susurro*)

La tabla 3.4 muestra los resultados obtenidos de EER para los distintos sistemas basados en GMM entrenados con voz neutra y testeados con voz susurrada. El primero de ellos utiliza 19 coeficientes MFCC de las señales de voz original (siguiendo el baseline propuesto por [16]), el segundo utiliza los coeficientes MFCC extraídos de los sonidos consonánticos y el último, se basa en el empleo de 26 coeficientes lineales LFCC.

Las técnicas basadas en GMM fueron muy populares hace unos años, sin embargo, en la actualidad se utilizan modelos más avanzados basados en redes neuronales. La red neuronal propuesta por Kaldi y entrenada con miles de muestras de voz neutra, es capaz de captar con mayor precisión las características de la señal de voz. Por ello, aunque el uso de los

coeficientes LFCC y la extracción de los sonidos consonánticos pueda suponer una mejora respecto a los coeficientes MFCC convencionales en el escenario *neutra - susurro*, este no puede competir contra una red neuronal con un gran número de parámetros y un mayor conjunto de entrenamiento.

El detector de susurro desarrollado en esta sección permite tratar de forma diferenciada las tramas de voz neutra y las tramas de voz susurrada. Este detector se utiliza en el sistema final desarrollado para distinguir las tramas de voz susurrada y no degradar el sistema para el escenario habitual *neutra-meutra*.

3.3.3. Conversión de voz susurrada a voz neutra con el sistema seq2seq

El sistema *sequence-to-sequence* tiene como objetivo la conversión de los datos de entrada de un dominio a otro. Se trata de un mapeo de los datos para los que se requieren algoritmos de aprendizaje automático complejos y que requieren mucha capacidad computacional.

3.3.3.1. Preprocesado de los datos

Para poder entrenar el sistema, es necesario normalizar la duración de los audios fuente y objetivo, aunque la duración máxima de los audios del dominio fuente (audios de voz susurrada) no tiene por que ser la misma que la de los audios del dominio objetivo (audios de voz neutra). Los audios se dividen en segmentos cuya duración máxima es tres segundos, utilizando para ello el algoritmo DTW que permite mantener la alineación entre los segmentos de audios de voz susurrada y los segmentos de audios de voz neutra. Aunque el algoritmo *seq2seq* no necesita alineación trama a trama, sí es necesario que la frase dicha por el locutor en el audio del dominio origen sea la misma que la dicha por el locutor en el dominio objetivo, de forma que la red pueda captar la relación entre dominios y el mapeo pueda realizarse correctamente. Tras obtener los coeficientes MFCC de los segmentos resultantes y normalizar las muestras mediante la ecuación (3.3), se iguala el número de

tramas de los segmentos con los que se va a entrenar la red mediante *zero-padding*. Finalmente, se añade al principio de cada segmento una trama de comienzo y al final una trama de cierre. La trama de comienzo es una trama de 0s y la trama de cierre una trama de 1s, siendo poco probable que alguna de las tramas de audio presente estos valores para sus coeficientes. Estas tramas permiten a la red captar el comienzo y final de cada una de las muestras.

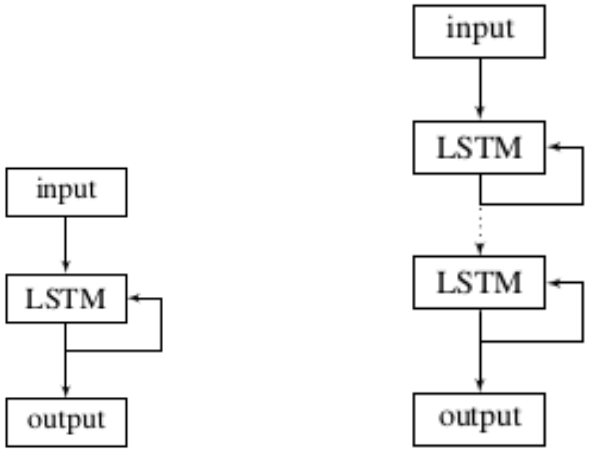
Tras el procesado, los datos se dividen en un conjunto de entrenamiento de 2587 muestras de voz susurrada (más el mismo número de muestras paralelas de voz neutra), y un conjunto de test de 73 muestras de voz susurrada y otras 73 de voz neutra correspondientes a un solo locutor. De esta forma, se evita que la red tenga información acerca del locutor que va a evaluarse durante el entrenamiento, tal y como sucedería en un escenario real. El número máximo de tramas para cada muestra de audio tanto de voz neutra como de voz susurrada es 336 aunque podría haber sido diferente para cada conjunto.

3.3.3.2. Arquitectura del sistema seq2seq

La arquitectura escogida para el sistema es la propuesta por [7]. El sistema se compone de un codificador y un decodificador, ambos formados por una red LSTM de dos capas de 256 unidades. El número total de parámetros a entrenar es 819.200 en codificador y 1.024.030 en decodificador. La red se ha desarrollado utilizando las librerías *Tensorflow* y *Keras* que permiten definir los distintos bloques que componen el sistema, la función de pérdidas, el optimizador..., así como los procesos de entrenamiento y evaluación.

Las redes LSTM formadas por más de una capa se conocen como *Deep LSTM*. Las redes recurrentes LSTM ya se consideran arquitecturas profundas, las entradas del modelo son procesadas por múltiples capas no lineales como sucede en las DNNs. Sin embargo, las características de un instante temporal se procesan por una sola capa no lineal para contribuir a la salida en ese instante. Añadir capas a estas arquitecturas permite que la entrada en un instante de tiempo se procese en cada una de las capas añadidas, además de

a lo largo del tiempo [35]. La figura 3.3 muestra las dos arquitecturas posibles LSTM [35].



(a) Red recurrente LSTM (b) Red *Deep*-LSTM

Figura 3.3: Arquitecturas LSTM RNN

La función de pérdidas utilizada para determinar la actualización de los parámetros de la red es el error cuadrático medio entre vectores.

En este trabajo se presentan dos aproximaciones distintas del sistema *seq2seq* que difieren en el entrenamiento:

- En primer lugar se utiliza la estrategia *teacher forcing*, en la que durante el proceso de entrenamiento el decodificador del sistema recibe como entrada, además de la salida del codificador, la trama del instante anterior de la muestra real objetivo.
- En segundo lugar, se utiliza la predicción del decodificador del instante anterior como entrada del mismo en el instante actual.

3.3.3.3. Resultados del sistema seq2seq

La estrategia *teacher forcing*, es muy utilizada en algoritmos de traducción de textos y facilita la convergencia del entrenamiento del modelo. Durante el entrenamiento, el error se decrementa rápidamente y se llegan a alcanzar errores cercanos al 0.1. La figura 3.4 muestra el error medio para cada una de las 50 épocas del entrenamiento del sistema *seq2seq*.

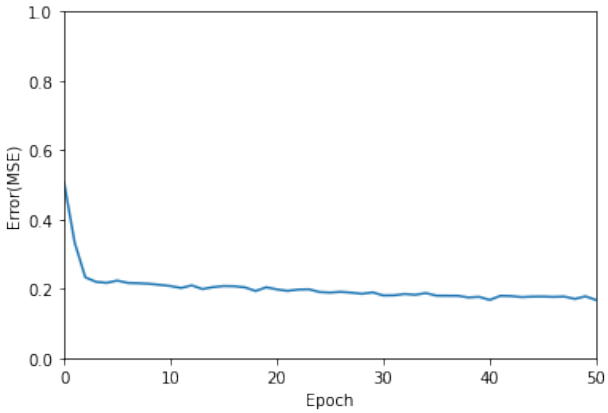
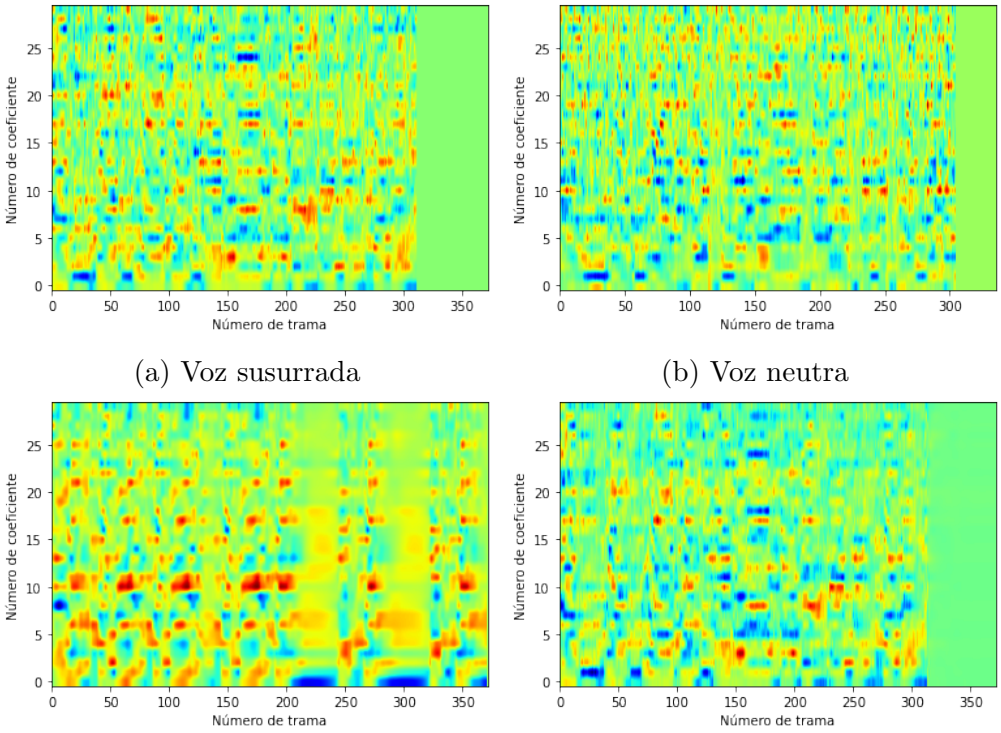


Figura 3.4: Error medio obtenido para cada época en el entrenamiento *seq2seq*

Sin embargo, en el proceso de inferencia, el modelo no consigue resultados aceptables siendo incluso peores a los obtenidos antes del entrenamiento de la red. El error medio obtenido para el conjunto de test es 1,41. Los resultados del entrenamiento, donde el decodificador tiene acceso a la información de la muestra real, no son representativos ya que en el proceso de inferencia el decodificador no puede acceder a esta información.

Esta estrategia aplicada a las muestras de audio, induce al modelo, debida a la estrecha relación temporal entre tramas, a dar como salida la misma trama que ha recibido como entrada. Esta técnica resulta en muchos casos inadecuada cuando las muestras de salida vuelven a introducirse a la entrada, debido a que las muestras que recibe la red durante el entrenamiento difieren mucho de las que recibe durante la evaluación[45].



(a) Voz susurrada

(b) Voz neutra

(c) Voz neutra transformada (utilizando la predicción)

(d) Voz neutra transformada (utilizando la trama objetivo)

Figura 3.5: Resultado de la transformación de los coeficientes MFCC en el proceso de inferencia del sistema *seq2seq*

Las figuras 3.5c y 3.5d muestran los resultados obtenidos durante el proceso de inferencia cuando se introduce como entrada al decodificador la trama predicha y la trama real del instante anterior respectivamente. Cuando se utiliza la predicción, el decodificador no sabe detectar cuando termina la secuencia (el segmento verde de las imágenes representa los valores nulos añadidos por *zero-padding*) y presenta zonas ciegas donde su salida es la predicción del instante anterior.

Algunos autores como [46], proponen estabilizar el entrenamiento mediante arquitecturas más complejas de preservación de contexto, modificando la función de pérdidas y que consiguen que el codificador cobre una mayor relevancia en la predicción.

3.3.3.4. Resultados del sistema seq2seq_pre

Como solución alternativa, [7] propone un entrenamiento en el que el decodificador recibe como entrada la trama predicha en la anterior iteración, de forma que entrenamiento e inferencia tienen el mismo comportamiento, se refiere a este modelo como *seq2seq_pre*. En este tipo de sistemas, la convergencia del entrenamiento es mucho más lenta. El problema resulta de actualizar los estados de las capas ocultas de codificador y decodificador con predicciones erróneas, de forma que los errores se acumulan[45].

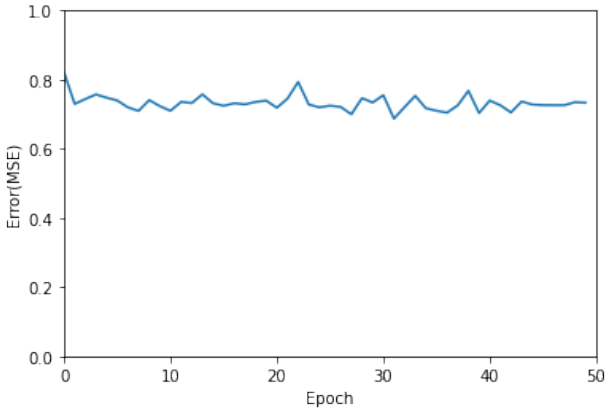


Figura 3.6: Error medio obtenido para cada época en el entrenamiento *seq2seq_pre*

La figura 3.6 muestra el error medio para cada una de las 50 épocas del entrenamiento. El error medio resultado de la evaluación del conjunto de test es 0,83 y aunque es menor que el del sistema *seq2seq*, no se predicen las tramas, como muestra la figura 3.7.

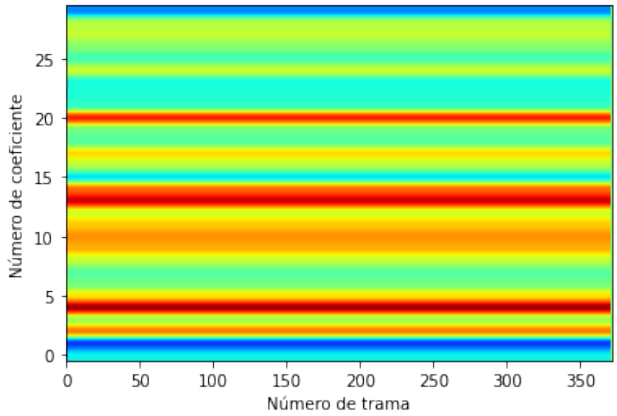


Figura 3.7: Resultado de la transformación de los coeficientes MFCC en el proceso de inferencia del sistema *seq2seq_pre*

El modelo compuesto por redes recurrentes de dos capas requiere una gran capacidad de cómputo. En los experimentos realizados, dados los problemas de este tipo de estrategia y la falta de una máquina potente capaz de entrenar la red, el entrenamiento no llega a converger.

3.3.4. Sistemas de mapeo de datos

Los dominios de voz susurrada y voz neutra pueden acercarse mediante mapeo de datos a la entrada de la red. Una de las desventajas de realizar el mapeo sobre los datos de audio de entrada, es la cantidad de datos a mapear. La red consigue condensar la información de los datos de audio en vectores de longitud fija, permitiendo que el mapeo de un dominio a otro sea más sencillo.

3.3.4.1. Multi-Environment Model-based LInear Normalization (MEMLIN) aplicado a los coeficientes MFCC

La técnica de MEMLIN utilizada por [37] propone un mapeo de datos de distintos entornos ruidosos e, a un entorno limpio. En este trabajo, se tiene un único entorno ruidoso (la voz susurrada) y un entorno limpio (la voz neutra), por lo que en las ecuaciones (2.12) y (2.13) no será necesario el índice e que se refiere al entorno, ni el cálculo del factor

α_e. Debido a la gran cantidad de características que representan cada una de las muestras en relación al número de muestras de las que se disponen, se opta por la solución *soft* del algoritmo (2.23). Para poder realizar el mapeo de un dominio a otro de forma correcta, es necesario alinear las tramas de voz susurrada y voz neutra. Para ello se utiliza el algoritmo DTW. De nuevo el código completo se desarrolla en *Pyhon*, utilizando *scikit learn* para los modelos GMM.

Los resultados varían según el número de Gaussianas utilizadas en el modelo GMM que configura los entornos. Los mejores resultados para el sistema completo (comparación *todos-todos*) se obtienen con modelos de 16 componentes.

Sistema	**Número de Gaussianas**	**Comparación**	**EER %**
Baseline	-	neutra - susurro	17.79 %
		todos - todos	22.47 %
MEMLIN	4	neutra - susurro	22.56 %
		todos - todos	22.56 %
	8	neutra - susurro	18.59 %
		todos - todos	21.9 %
	16	neutra - susurro	18.52 %
		todos - todos	21.63 %

Tabla 3.5: Comparativa del valor de EER resultado de la aplicación del algoritmo MEMLIN a los datos de entrada de la red

La tabla 3.5, presenta los resultados obtenidos en la aplicación del algoritmo de MEMLIN sobre los datos de entrada.

El algoritmo no consigue mejorar los resultados del baseline de forma notable. La aplicación del algoritmo de MEMLIN a los datos de entrada es muy costosa computacionalmente debido a la cantidad de datos que representa cada una de las muestras. Además el mapeo de datos no resulta adecuado ya que cada uno de los

componentes que conforman la muestra aporta poca información.

3.3.4.2. Linear Multivariate Regression (LMR) aplicado a los coeficientes MFCC

La aplicación del algoritmo LMR a los coeficientes MFCC de las tramas de audio no es posible debido al coste computacional que supone la inversión de la matriz $\tilde{C}^{r,q^\dagger}$. Si bien podría plantearse el algoritmo con un mapeo trama a trama del audio, este no sería eficiente puesto que como se ha comentado en la sección anterior, cada unas de las muestras del audio no recoge suficiente información para permitir la correcta abstracción. Es por ello que cobra interés la aplicación de estas técnicas a la salida de la red, que consigue representar las muestras de audio en vectores unidimensionales de 512 muestras.

3.4. Técnicas aplicadas a los datos de salida de la red (Back-End)

La red neuronal TDNN descrita en 2.2 recibe como entrada los coeficientes MFCC de la señal de voz y obtiene a su salida un vector de 512 coeficientes con el que un clasificador PLDA es capaz de determinar si dos muestras pertenecen al mismo locutor. Este vector, resume la información de la señal y por tanto mantiene muchas de las diferencias que existen entre el dominio de voz neutra y voz susurrada. La librería TSNE permite visualizar los vectores de características en un espacio de dos dimensiones. Se trata de una herramienta muy útil para ver de forma gráfica como se desplazan los vectores al aplicar distintas técnicas, y lograr reconocer aquellos sistemas que permiten acercar los dominios.

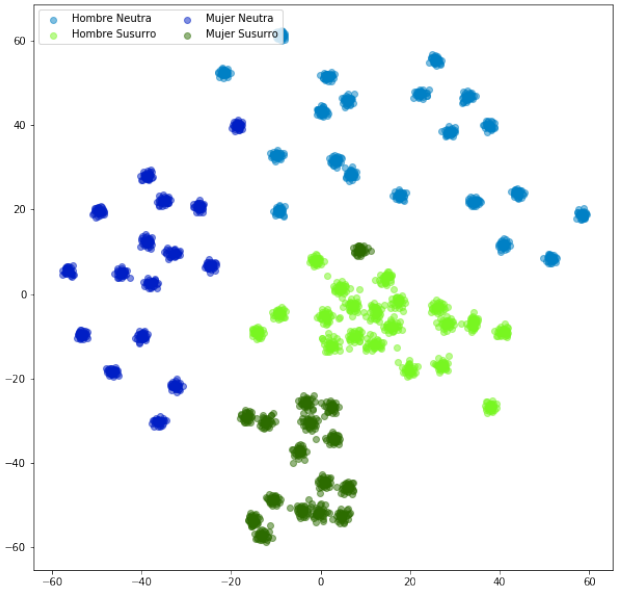


Figura 3.8: Representación en dos dimensiones de los vectores de características a la salida de la red

La representación en dos dimensiones los vectores originales a la salida de la red se muestra en la figura 3.8. En esta figura se distingue un cluster de datos de susurro (azul) y otro de datos de voz neutra (verde). Además, se diferencian las mujeres (oscuro) de los hombres (claro) por tonalidad, para verificar si los dominios se encuentran separados por sexo. Dos puntos pueden encontrarse próximos por pertenecer al mismo dominio de sexo pero a distinto dominio de registro de voz.

3.4.1. Resta de la media

El centro de los grupos que aparecen en la figura 3.8 esta relacionado con la media de los datos representados. Por defecto, en el sistema original desarrollado en Kaldi, a la salida de la red, se resta a los *xvectors* la media calculada para los datos de *VoxCeleb* (datos de entrenamiento de la red). Lo que se propone a continuación, es restar la media del conjunto de datos que se está verificando.

Comparación	Media	EER %
neutra - neutra	Baseline	1.981 %
	neutra	1.468 %
susurro - susurro	Baseline	7.533 %
	susurro	5.13 %
neutra - susurro	Baseline	17.79 %
	susurro	16.1 %
	neutra	18.34 %
	neutra + susurro	17.24 %
	neutra - susurro	14.43 %
todos - todos	Baseline	22.47 %
	susurro	17.44 %
	neutra	24.9 %
	neutra + susurro	20.19 %
	neutra - susurro	10.25 %

Tabla 3.6: Comparativa del valor de EER resultado de la extracción de distintas medias a los datos de salida de la red

La tabla 3.6 muestra los resultados de la extracción de la media en distintos escenarios. La primera columna hace referencia a las muestras comparadas, la segunda a los datos con los que se ha calculado la media que se substrae y la tercera el valor de EER en porcentaje obtenido. La media se calcula según el caso para los vectores de voz susurrada (*susurro*), los vectores de voz neutra (*neutra*), el conjunto de todos los vectores (*neutra + susurro*) y, por último, se calculan dos medias individuales y se resta cada una a su conjunto de datos correspondiente (*neutra - susurro*). Para que los experimentos sean realistas, se excluyen las muestras del locutor que se evalúa para el cálculo y substracción de la media en cada caso. Además, se utiliza el detector de susurro desarrollado y cuyos resultados se presentan en la

sección 3.3.2, para los casos en los que la media se resta según el dominio. La resta de la media correspondiente a cada dominio logra disminuir el error en mas de un 10 %.

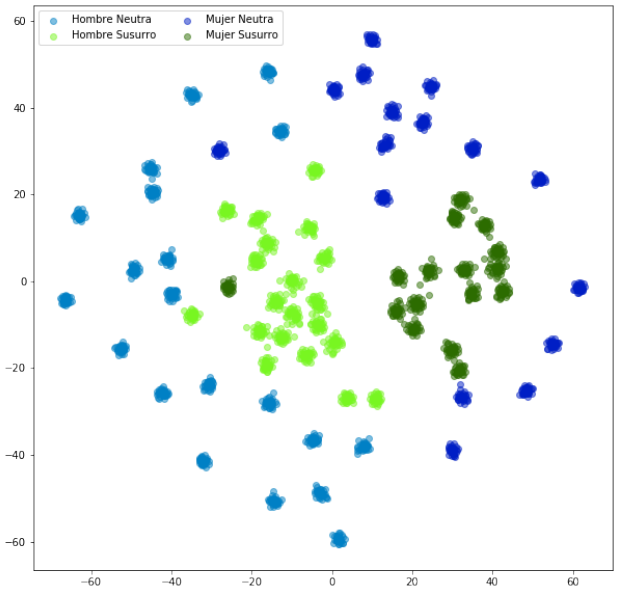


Figura 3.9: Representación en dos dimensiones de los vectores de características tras la extracción de la media según el dominio

La figura 3.9 permite comprobar como se han aproximado los dominios tras la extracción de la media. En este caso, los vectores correspondientes a voces masculinas se sitúan a la izquierda y los de voces femeninas a la derecha. Los vectores de voz neutra se concentran más cerca del centro pero ya no se da la separación tan acusada que se observa en la figura 3.8.

3.4.2. Multi-Environment Model-based LInear Normalization (MEMLIN) aplicado a los vectores de salida

El algoritmo de MEMLIN se basa en el mapeo mediante una transformación lineal de los datos de un dominio origen a un dominio objetivo. Los datos del dominio origen, a los que se aplica el algoritmo, cambian a lo largo del tiempo y, por tanto, se tiene en cuenta el índice temporal t en las ecuaciones (2.17) y (2.21).

En el presente trabajo, a la salida de la red del sistema de referencia, cada muestra de audio queda representada por un vector de longitud fija de 512 componentes, por tanto no existe la componente temporal. En este caso, y a diferencia del explicado en la sección 3.3.4, las muestras se representan con un número limitado de características y la solución *hard* (2.22) proporciona buenos resultados. Para que los resultados sean realistas, se utiliza el detector de susurro que permite reconocer qué muestras precisan de calibración y cuales no.

Sistema	**Número de Gaussianas**	**Comparación**	**EER %**
Baseline	-	susurro - susurro	7.533 %
		neutra - susurro	17.79 %
MEMLIN	4	susurro - susurro	8.171 %
		neutra - susurro	19.04 %
		todos - todos	11.47 %
	8	susurro - susurro	10.52 %
		neutra - susurro	19.56 %
		todos - todos	12.38 %

Tabla 3.7: Comparativa del valor de EER resultado de la aplicación del algoritmo MEMLIN a los datos de salida de la red

Los resultados recogidos en la tabla 3.7, muestran como el algoritmo de MEMLIN no consigue mejorar los resultados en las comparaciones *neutra- susurro*, sin embargo, sí los mejora cuando se comparan todas las muestras. Al acercar los dominios, se produce una calibración implícita entre las distintas comparaciones que permite una puntuación y un umbral de decisión más apropiado para cada comparación. Los mejores resultados de la aplicación del algoritmo de MEMLIN sobre los *xvectors* en el sistema completo (comparación *todos-todos*), se obtienen con modelos GMM de 4 componentes. La figura 3.10 muestra la representación en dos dimensiones de los vectores obtenidos tras la aplicación del algoritmo de MEMLIN a los vectores de salida de la red con modelos GMM de 4 componentes.

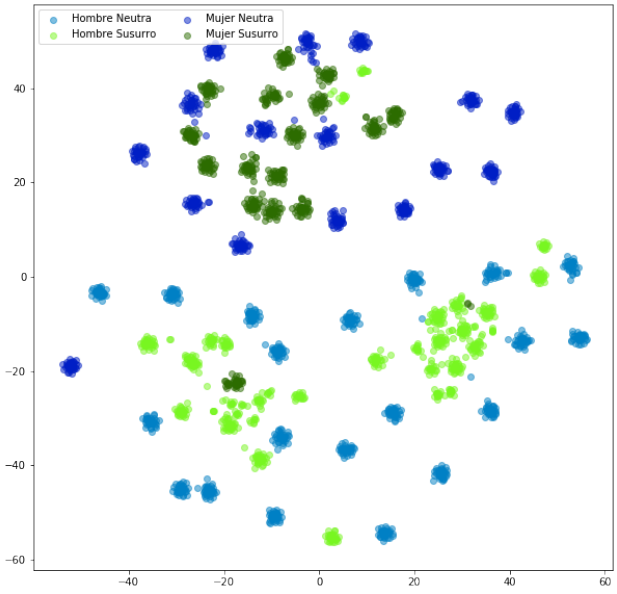


Figura 3.10: Representación en dos dimensiones de los vectores de características tras la aplicación del algoritmo de MEMLIN con GMM de 4 componentes

Esta representación confirma que el modelo de MEMLIN logra acercar los dominios y por tanto facilitar la tarea de reconocimiento del clasificador. A diferencia de la extracción de la media, el algoritmo de MEMLIN consigue que los datos de susurro y voz neutra queden integrados sin formar grupos y predomina la diferenciación por género.

3.4.3. Linear Multivariate Regression (LMR) aplicado a los vectores de salida

La segunda técnica de mapeo propuesta en el trabajo se conoce como LMR. A la entrada de la red, no es posible utilizar esta técnica de forma directa debido al coste computacional que supone la inversión de matrices de grandes dimensiones. A la salida de la red, las muestras quedan representadas por vectores de 512 componentes, por lo que el coste computacional deja de ser un problema. Para facilitar el mapeo de los datos, estos se agrupan por categorías y para ello se utiliza el algoritmo KNN, *K-Nearest Neighbors*, de la librería *scikit learn*. Como en el resto de los experimentos llevados a cabo en este trabajo, para la evaluación del sistema se utiliza la técnica de *leave-one-out*, y ninguna de las muestras del locutor que se evalúa se

incluye en el conjunto de entrenamiento.

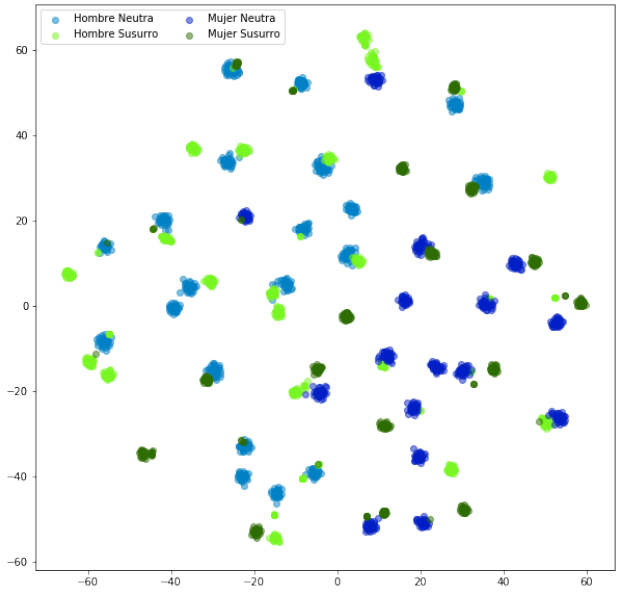
Sistema	**Número de categorías**	**Comparación**	**EER %**
Baseline	-	neutra - susurro	17.79 %
		todos - todos	22.47 %
LMR	4	neutra - susurro	40.01 %
		todos - todos	29.07 %
	8	neutra - susurro	37.44 %
		todos - todos	27.59 %
	16	neutra - susurro	33.21 %
		todos - todos	24.81 %
	32	neutra - susurro	43.42 %
		todos - todos	26.04 %

Tabla 3.8: Comparativa del valor de EER resultado de la aplicación del algoritmo LMR a los datos de salida de la red

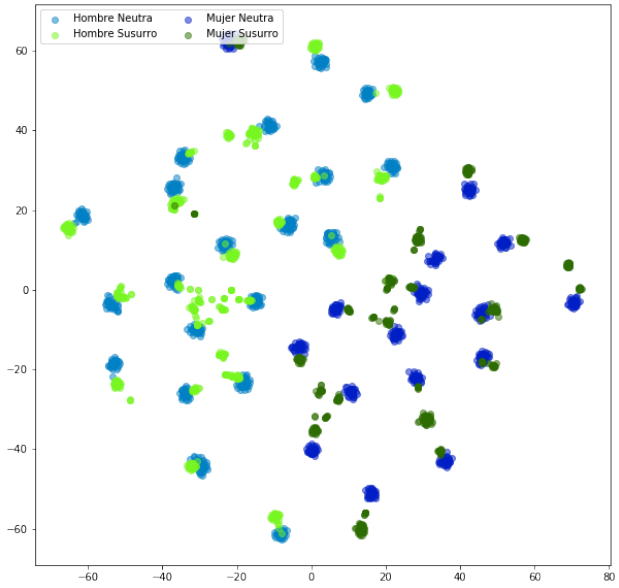
La tabla 3.8 muestra los resultados del sistema LMR entrenado sin incluir muestras del locutor evaluado para distinto número de grupos o categorías. LMR no consigue mejorar los resultados en ninguno de los casos. Sin embargo, cuando se dispone de muestras del locutor evaluado en el conjunto de entrenamiento (Tabla 3.9), se obtienen buenos resultados. Esta técnica de mapeo se encuentra muy ligada al locutor y no consigue generalizar cuando no dispone de muestras del mismo.

Sistema	**Número de categorías**	**Comparación**	**EER %**
Baseline	-	neutra - susurro	17.79 %
		todos - todos	22.47 %
LMR	4	neutra - susurro	36.85 %
		todos - todos	26.91 %
	8	neutra - susurro	26.94 %
		todos - todos	19.03 %
	16	neutra - susurro	14.76 %
		todos - todos	11.1 %
	32	neutra - susurro	6.419 %
		todos - todos	4.958 %

Tabla 3.9: Comparativa del valor de EER resultado de la aplicación del algoritmo LMR a los datos de salida de la red (incluyendo datos del locutor en el entrenamiento)



(a) Sin muestras del locutor en el conjunto de entrenamiento



(b) Con muestras del locutor en el conjunto de entrenamiento

Figura 3.11: Representación en dos dimensiones de los vectores tras la aplicación del algoritmo LMR (32 categorías)

La figura 3.11 muestra la representación en dos dimensiones de los vectores de salida de la red tras la aplicación del algoritmo LMR con 32 categorías, contando y no contando con muestras del locutor que se evalúa. Las muestras de ambos dominios se aproximan

mucho y de forma precisa cuando el conjunto de entrenamiento cuenta con las muestras del locutor que se evalúa. El sistema modela por locutor. Cuando el sistema no dispone de estas muestras el acercamiento de los dominios no es correcto y tal y como refleja la figura 3.11a los locutores del dominio de voz susurrada se acercan a locutores distintos del dominio de voz neutra.

Pese a que los resultados son buenos cuando se dispone de muestras del locutor en el conjunto de entrenamiento, este trabajo está dirigido a un sistema real que no dispondría de esta ventaja. Por tanto, se descarta LMR como solución de mejora del sistema.

3.5. Técnicas aplicadas a las puntuaciones

El mapeo de datos de un dominio a otro facilita la tarea del clasificador, acotando las distancias existentes entre las distintas comparaciones. Otra forma de conseguirlo, es mediante un sistema de calibración de puntuaciones, que permita modificar el umbral de clasificación según la comparación que se esté dando.

3.5.1. Técnicas de calibración

Como se explicó en la sección 2.2.5, el sistema se evalúa en función del *EER* que es el valor para el cual el valor de *FRR* es igual al valor de *FAR*. Este valor, se corresponde con un umbral de puntuación a partir del cual la comparación entre dos muestras resulta positiva (se verifica al locutor) o negativa (se rechaza al locutor). Los distintos grupos de comparaciones *susurro-susurro*, *neutra-neutra* y *susurro-neutra* presentan distinto valor de umbral y por tanto, en la clasificación *todos-todos* se producen errores, se consideran positivas muestras de grupos cuyo umbral es menor (*neutra-neutra*) y negativas muestras de grupos cuyo umbral es mayor (*susurro-neutra*). La calibración de puntuaciones trata de resolver este problema, moviendo los umbrales de cada grupo de comparaciones en la clasificación global.

Para ello, es necesario, en primer lugar, entrenar un modelo de regresión logística LR,

Logistic Regression, para cada dominio de comparaciones. A continuación, cada muestra en el sistema se clasifica como muestra de susurro o muestra de voz neutra, utilizando para ello el clasificador basado en GMM. Esta clasificación permite conocer que tipo de comparación se está dando y, obtener el valor proporcionado por el modelo LR correspondiente para la puntuación calculada. Este valor es la probabilidad de que la verificación sea positiva de acuerdo al modelo LR. Con estos nuevos valores se define el valor de EER y el umbral asociado.

Sistema	**Comparación**	**EER %**
Baseline	neutra-neutra	1.981 %
	susurro-susurro	7.533 %
	neutra-susurro	17.79 %
	todos-todos	22.47 %
Calibración	neutra-neutra	1.98 %
	susurro-susurro	7.533 %
	neutra-susurro	17.79 %
	todos-todos	9.21 %

Tabla 3.10: Comparativa del valor de EER resultado de la calibración de puntuaciones

La tabla 3.10 muestra los resultados del sistema tras la calibración de las puntuaciones.

La calibración de puntuaciones consigue mejorar los resultados del sistema completo (comparación *todos-todos*) en más de un 10 %. Los resultados para cada grupo de comparaciones no se ven afectados puesto que se basa en un desplazamiento del umbral en el sistema completo y no dentro de cada grupo.

3.6. Resultado del sistema mejorado

Por último, se recogen las técnicas con las que se han obtenido los mejores resultados y se combinan para mejorar el sistema final.

Sistema	**EER % (*todos - todos*)**
Baseline	22.47 %
MEMLIN + Calibración	9.47 %
MEMLIN + Extracción de media (global)	9.828 %
MEMLIN + Extracción de media (por dominio)	10.21 %
MEMLIN + Extracción de media (global) + Calibración	8.612 %
MEMLIN + Extracción de media (por dominio) + Calibración	7.913 %

Tabla 3.11: Comparativa del valor de EER resultado de la aplicación conjunta de las técnicas estudiadas

Los mejores resultados se obtienen cuando se unen las tres técnicas y se utiliza la media por dominio. Se consigue disminuir el valor del EER en un 14.55 %. La figura 3.12 muestra los vectores de salida de la red tras la extracción de la media por dominio y la aplicación del algoritmo de MEMLIN.

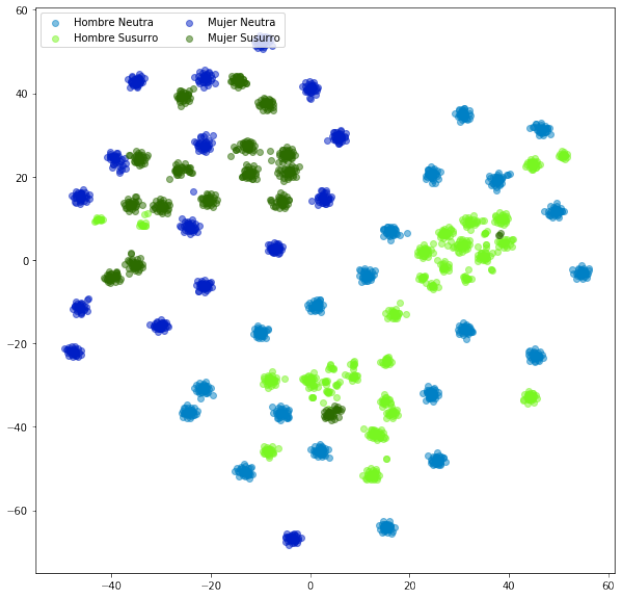
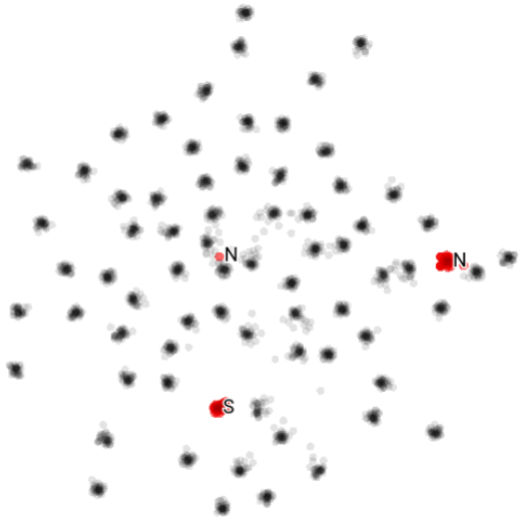
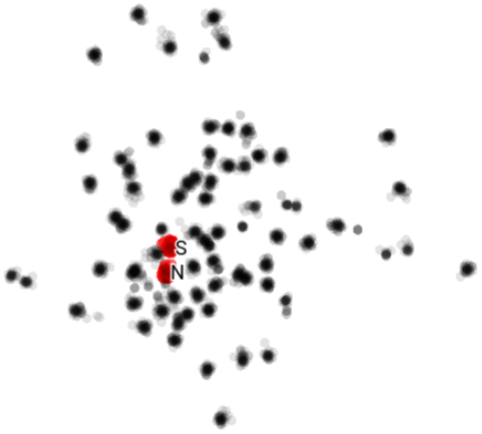


Figura 3.12: Representación en dos dimensiones de los vectores a la salida de la red del sistema final

El resultado es muy similar al mostrado en la figura 3.10, la diferencia del valor de EER calculado sobre cada conjunto de vectores (*MEMLIN* y *MEMLIN + extracción de media*) es tan sólo de un 1.2 %. La herramienta de tensorflow *projector* permite filtrar las muestras y representarlas en dos dimensiones utilizando TSNE. Puede observarse en la figura 3.13 como se logra disminuir las distancias entre muestras de distintos dominios. En esta figura, las muestras iluminadas en rojo corresponden a uno de los locutores. La etiqueta "S" hace referencia a la voz susurrada y la etiqueta "N", a la voz neutra.



(a) Baseline



(b) MEMLIN + Extracción de media global

Figura 3.13: Representación en dos dimensiones de las muestras a la salida de la red de un locutor antes y después de la mejora del sistema.

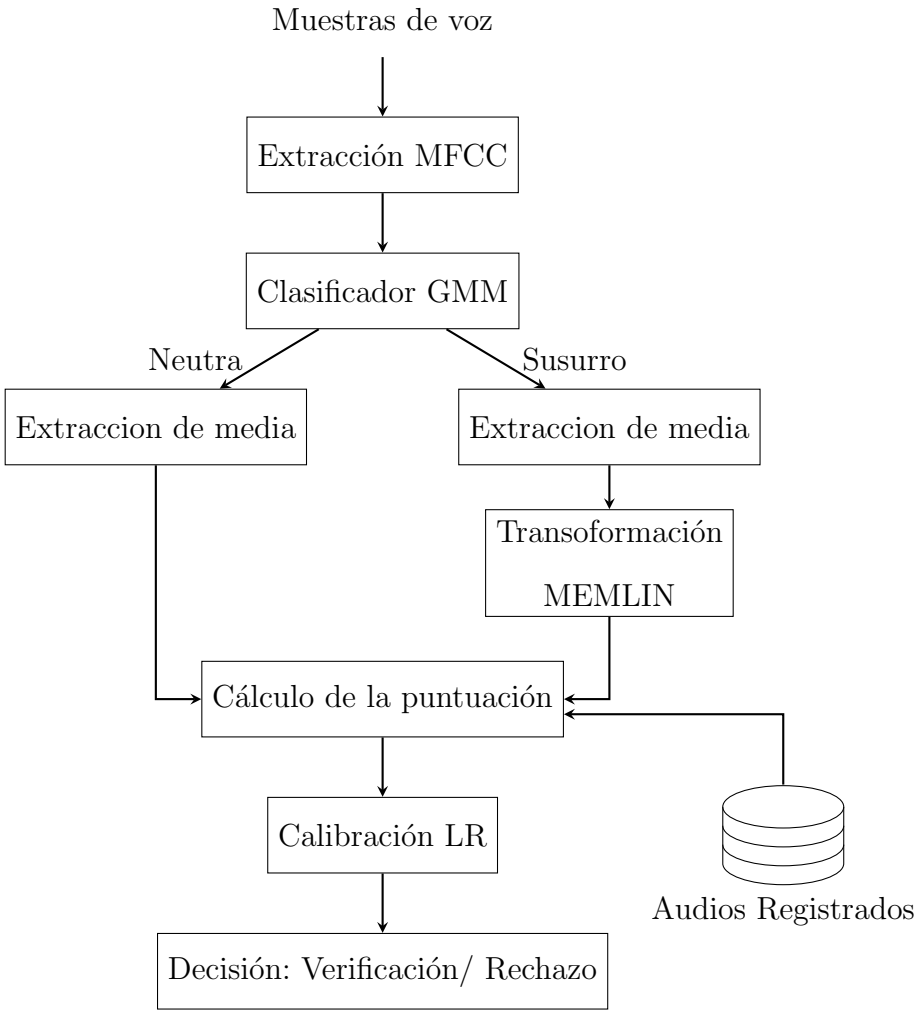


Figura 3.14: Diagrama de bloques del sistema de verificación final.

La figura 3.14 muestra el diagrama de bloques del sistema de verificación final. El clasificador de susurro, los valores de mapeo del algoritmo de MEMLIN y los modelos LR para la calibración se entrenan previamente con datos de entrenamiento. La media se calcula también con ese conjunto de datos. Los audios recogidos en el proceso de registro se consideran clasificados según su registro de voz antes de guardarse en la base de datos, en caso contrario, sería necesario clasificarlos utilizando el clasificador GMM. La calibración LR dependerá del dominio de la muestra registrada en el sistema y la que trata de verificarse.

Capítulo 4

Conclusiones

Los sistemas de verificación de locutores tratan de confirmar la identidad de un usuario comparando para ello dos muestras de voz recogidas en ficheros de audio. Algunas enfermedades como la afonía, impiden que los locutores utilicen la voz neutra para comunicarse, y estos deben recurrir al susurro. En estas situaciones los sistemas de verificación ven altamente perjudicado su rendimiento. En este trabajo se analizan herramientas y técnicas de mejora de los sistemas de verificación en las situaciones en las que se toman muestras de voz susurrada.

El primer conjunto de técnicas estudiadas reúne las aplicadas sobre los datos de entrada a la red, técnicas de *front-end*. En el sistema final, ninguna de estas técnicas se ha incluido debido a su ineficiencia en la mejora del mismo por distintas razones:

- En primer lugar, la recogida de datos distintos a los del sistema de referencia no logra mejorar los resultados. Los coeficientes MFCC utilizados por la red de Kaldi son capaces de reflejar fielmente las características de la señal de voz y, procesados por un sistema complejo, entrenado con un gran número de datos y capaz de capturar las peculiaridades de cada locutor, permite obtener muy buenos resultados en la verificación. Sin embargo, en sistemas más sencillos como los modelos GMM, la recogida de datos que mantengan las características del locutor en ambos dominios

ha demostrado mejorar los resultados.

- El objetivo del trabajo es la mejora de un sistema referencia del estado del arte, ya entrenado, y que obtiene buenos resultados en condiciones de voz neutra. Sin embargo, la recogida de datos diferentes en la verificación de locutores que utilicen el susurro no puede aplicarse directamente al sistema y conllevaría la creación de un modelo paralelo.

- El correcto mapeo de los datos de entrada de un dominio a otro, no puede realizarse mediante un modelo simple, ya que a la entrada de la red cada audio se representa con un gran número de muestras y cada muestra no agrupa información suficiente y condensada sobre el locutor. Para obtener buenos resultados mediante el mapeo de los datos de entrada, es necesario utilizar sistemas complejos para los que se requiere un número elevado de muestras de audio que permita entrenar el modelo y una gran capacidad de cómputo.

A la salida de la red que conforma el bloque principal del sistema de referencia, se tienen vectores de 512 componentes. Las aplicación de distintas técnicas sobre estos vectores resulta eficiente en la mejora del sistema.

- La normalización de los vectores mediante la resta de la media permite acercar los dominios, facilitando la tarea del clasificador.

- Los vectores de longitud fija consiguen reunir la información relevante en la clasificación en un número limitado de muestras.

- El mapeo de estos datos de un dominio a otro permite calibrar el sistema, de forma que el umbral utilizado por el clasificador sea adecuado para ambos dominios.

- El mapeo de los datos consigue mejorar los resultados del sistema completo, pero no mejora su precisión en el escenario de registro con voz neutra y evaluación con voz

susurrada. Para mejorar este escenario, sería necesario aplicar técnicas basadas en el procesado de la señal de voz.

Finalmente, la calibración de puntuaciones consigue un efecto similar al del mapeo de datos. Conocer donde se sitúa el umbral a partir del cual dos muestras se consideran del mismo locutor según el dominio al que pertenecen las muestras, permite que el clasificador obtenga resultados precisos independientemente del número de comparaciones que se tengan de cada tipo (*susurro-susurro*, *neutra-neutra* y *neutra-susurro*).

Los mejores resultados se obtienen mediante el empleo conjunto de la normalización, el mapeo de datos y la calibración de puntuaciones. Las tres técnicas permiten calibrar el sistema de distinta manera y funcionan correctamente cuando se utilizan de forma conjunta.

Bibliografía

[1] C. Zhang and J. Hansen, "Analysis and classification of speech mode: Whispered through shouted," in *Eighth Annual Conference of the International Speech Communication Association*, pp. 2289–2292, 2007.

[2] A. Larcher, K. A. Lee, B. Ma, and H. Li, "Text-dependent speaker verification: Classifiers, databases and rsr2015," *Speech Communication*, vol. 60, pp. 56 – 77, 2014.

[3] H. Boies D. and L. M. Heck, "Study on the effect of lexical mismatch in text-dependent speaker verification," *Odyssey Speaker and Language Recognition Workshop*, p. 1–5, June 2004.

[4] V. M. Sardar and S. D. Shirbahadurkar, "Speaker identification of whispering speech: An investigation on selected timbrel features and knn distance measures," *Int. J. Speech Technol.*, vol. 21, p. 545–553, Sept. 2018.

[5] H. Beigi, "Speaker recognition: Advancements and challenges," in *New Trends and Developments in Biometrics* (J. Yang and S. J. Xie, eds.), ch. 1, Rijeka: IntechOpen, 2012.

[6] X. Fan and J. H. L. Hansen, "Acoustic analysis and feature transformation from neutral to whisper for speaker identification within whispered speech audio streams," *Speech Commun.*, vol. 55, p. 119–134, Jan. 2013.

[7] H. Lian, Y. Hu, W. Yu, J. Zhou, and W. Zheng, "Whisper to normal speech conversion using sequence-to-sequence mapping model with auditory attention," *IEEE Access*, vol. 7, pp. 130495–130504, 2019.

[8] F. Cummins, M. Grimaldi, T. Leonard, and J. Simko, "The chains corpus: Characterizing individual speakers," *Proc. SPECOM*, pp. 431–435, 01 2006.

[9] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust dnn embeddings for speaker recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5329–5333, 2018.

[10] G. Fant, *Acoustic theory of speech production*. No. 2, Walter de Gruyter, 1970.

[11] L. Docio-Fernandez and C. García Mateo, *Speech Production*, pp. 1493–1498. Boston, MA: Springer US, 2015.

[12] I. B. Thomas, "Perceived pitch of whispered vowels," *The Journal of the Acoustical Society of America*, vol. 46, no. 2B, pp. 468–470, 1969.

[13] J. H. Esling, S. R. Moisik, A. Benner, and L. Crevier-Buchman, *Laryngeal Voice Quality Classification*, p. 37–82. Cambridge Studies in Linguistics, Cambridge University Press, 2019.

[14] T. Ito, K. Takeda, and F. Itakura, "Analysis and recognition of whispered speech," *Speech Communication*, vol. 45, pp. 139–152, 02 2005.

[15] D. Grozdic and S. Jovicic, "Whispered speech recognition using deep denoising autoencoder and inverse filtering," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, pp. 2313–2322, 12 2017.

[16] X. Fan and J. Hansen, "Speaker identification within whispered speech audio streams," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 19, pp. 1408 – 1421, 08 2011.

[17] M. Xu, L.-Y. Duan, J. Cai, L.-T. Chia, C. Xu, and Q. Tian, "Hmm-based audio keyword generation," in *Proceedings of the 5th Pacific Rim Conference on Advances in Multimedia Information Processing - Volume Part III*, vol. 3333, pp. 566–574, 11 2004.

[18] N. Obin, "Cries and whispers - classification of vocal effort in expressive speech," *Interspeech*, 09 2012.

[19] K. J. Kallail and F. W. Emanuel, "Formant-frequency differences between isolated whispered and phonated vowel samples produced by adult female subjects," *Journal of Speech, Language, and Hearing Research*, vol. 27, no. 2, pp. 245–251, 1984.

[20] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The kaldi speech recognition toolkit," in *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*, IEEE Signal Processing Society, Dec. 2011. IEEE Catalog No.: CFP11SRW-USB.

[21] V. Peddinti, D. Povey, and S. Khudanpur, "A time delay neural network architecture for efficient modeling of long temporal contexts," in *INTERSPEECH*, 2015.

[22] S. Ioffe, "Probabilistic linear discriminant analysis," in *Computer Vision – ECCV 2006* (A. Leonardis, H. Bischof, and A. Pinz, eds.), (Berlin, Heidelberg), pp. 531–542, Springer Berlin Heidelberg, 2006.

[23] D. E. Sturim and D. A. Reynolds, "Speaker adaptive cohort selection for tnorm in text-independent speaker verification," in *Proceedings. (ICASSP '05). IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005.*, vol. 1, pp. I/741–I/744 Vol. 1, 2005.

[24] Jyh-Min Cheng and Hsiao-Chuan Wang, "A method of estimating the equal error rate for automatic speaker verification," in *2004 International Symposium on Chinese Spoken Language Processing*, pp. 285–288, 2004.

[25] R. D. Zilca and Y. Bistritz, "Feature concatenation for speaker identification," in *2000 10th European Signal Processing Conference*, pp. 1–4, 2000.

[26] S. Prasath, H. Alfeilat, O. Lasassmeh, A. Hassanat, and A. Tarawneh, "Distance and similarity measures effect on the performance of k-nearest neighbor classifier - a review," 08 2017.

[27] S. Kwon and S. Narayanan, "Robust speaker identification based on selective use of feature vectors," *Pattern Recognition Letters*, vol. 28, pp. 85–89, 01 2007.

[28] S. Deshmukh and S. Bhirud, "A hybrid selection method of audio descriptors for singer identification in north indian classical music," in *2012 Fifth International Conference on Emerging Trends in Engineering and Technology*, pp. 224–227, 11 2012.

[29] T. Hueber, G. Chollet, B. Denby, and M. Stone, "Acquisition of ultrasound, video and acoustic speech data for a silent-speech interface application," 01 2008.

[30] S. E. Bou-Ghazale and J. H. L. Hansen, "A comparative study of traditional and newly proposed features for recognition of speech under stress," *IEEE Transactions on Speech and Audio Processing*, vol. 8, no. 4, pp. 429–442, 2000.

[31] X. Fan and J. H. L. Hansen, "Speaker identification for whispered speech based on frequency warping and score competition," in *INTERSPEECH*, 2008.

[32] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[33] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[34] F. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with lstm," *Neural computation*, vol. 12, pp. 2451–71, 10 2000.

[35] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, pp. 338–342, 01 2014.

[36] A. Acero and X. Huang, "Augmented cepstral normalization for robust speech recognition," in *Proc. of IEEE Automatic Speech Recognition Workshop*, pp. 146–147, 1995.

[37] L. Buera, E. Lleida, A. Miguel, and A. Ortega, "Multi-environment models based linear normalization for speech recognition in car conditions," in *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 1, pp. I–1013, 2004.

[38] H. Valbret, E. Moulines, and J. Tubach, "Voice transformation using psola technique," *Speech Communication*, vol. 11, no. 2, pp. 175 – 187, 1992. Eurospeech '91.

[39] Y. Linde, A. Buzo, and R. Gray, "An algorithm for vector quantizer design," *IEEE Transactions on Communications*, vol. 28, no. 1, pp. 84–95, 1980.

[40] F. Kelly and J. Hansen, "Detection and calibration of whisper for speaker recognition," in *2018 IEEE Spoken Language Technology Workshop (SLT)*, 12 2018.

[41] N. Brümmer and J. [du Preez], "Application-independent evaluation of speaker detection," *Computer Speech and Language*, vol. 20, no. 2, pp. 230 – 275, 2006. Odyssey 2004: The speaker and Language Recognition Workshop.

[42] N. van Leeuwen, David A.and Brümmer, *An Introduction to Application-Independent Evaluation of Speaker Recognition Systems*, pp. 330–353. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007.

[43] S. Pigeon, P. Druyts, and P. Verlinde, "Applying logistic regression to the fusion of the nist'99 1-speaker submissions," *Digit. Signal Process.*, vol. 10, pp. 237–248, 2000.

[44] P. Vassilakis, "Sra: A web-based research tool for spectral and roughness analysis of sound signals," *Proceedings of the 4th Sound and Music Computing Conference, SMC 2007*, 01 2007.

[45] A. Lamb, A. Goyal, Y. Zhang, S. Zhang, A. Courville, and Y. Bengio, "Professor forcing: A new algorithm for training recurrent networks," 2016.

[46] K. Tanaka, H. Kameoka, T. Kaneko, and N. Hojo, "Atts2s-vc: Sequence-to-sequence voice conversion with attention and context preservation mechanisms," 2018.