UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO INGENIERÍA ELÉCTRICA

ESTUDIO COMPARATIVO DE TÉCNICAS PARA ROBUSTEZ DE SISTEMAS DE VERIFICACIÓN DE LOCUTOR TEXTO INDEPENDIENTE

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL ELECTRICISTA

JOSUÉ ABRAHAM FREDES SANDOVAL

PROFESOR GUÍA:
NÉSTOR BECERRA YOMA

MIEMBROS DE LA COMISIÓN:
VÍCTOR POBLETE RAMÍREZ
FERNANDO HUENUPÁN QUINÁN

SANTIAGO DE CHILE
2015

RESUMEN DE LA MEMORIA
PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL ELECTRICISTA
POR: Josué Fredes Sandoval.
FECHA: 31 / 03 / 2015
PROF. GUÍA: Dr. Néstor Becerra Yoma

Estudio comparativo de técnicas para robustez de sistemas de verificación de locutor texto independiente

Las técnicas de biometría son métodos automáticos de verificación o reconocimiento de la identidad de una persona basándose en una característica fisiológica o de comportamiento. En este marco se encuentra la tarea de verificación de locutor, que es el proceso de verificar la identidad de una persona basada en su señal de voz.

Un sistema de verificación de locutor usualmente es entrenado bajo ciertas condiciones de grabación o de canal de comunicación, y utilizar el sistema bajo otras condiciones de canal puede ser problemático. Debido a esto se han desarrollado diversas técnicas para cancelar o compensar el efecto del ruido y del canal, y así hacer la tecnología más robusta. En los últimos años se han propuesto nuevas técnicas basadas en análisis factorial que intentan modelar el efecto del canal de comunicación sobre la señal de voz.

En este trabajo se analizan en detalle dos sistemas de análisis factorial: *Joint Factor Analysis* o JFA, y *Total Variability Front-End Factor Analysis* más conocido como *i-Vectors*. Se implementaron ambos sistemas para ser integrados al conjunto de softwares para verificación de locutor del Laboratorio de Procesamiento y Transmisión de Voz, LPTV, de la Universidad de Chile.

Los sistemas implementados se validaron usando un software de referencia que tiene rendimiento *state-of-the-art*. Al utilizar las mismas condiciones iniciales de entrenamiento, los sistemas JFA e *i-Vectors* desarrollados para el LPTV igualaron el rendimiento del software de referencia, validando así su implementación.

Dedicado mis padres y a mi esposa Priscila

Agradecimientos

Este trabajo concluye una larga etapa de mi vida.

Agradezco al profesor Néstor Becerra y a los integrantes del LPTV. Estar en este laboratorio me ayudó no sólo académicamente, si no anímicamente también. En especial quiero mencionar a Víctor Poblete y Felipe Espic con quienes colaboré activamente en lo que sería finalmente mi tema de memoria.

Agradezco a mis padres por su paciencia y preocupación, esto tomó mucho más de lo que debería. Hay lecciones que se aprenden de la manera difícil.

Finalmente agradezco a mi esposa por su apoyo incondicional y por creer en mí siempre.

Tabla de Contenido

Índice de tablas

Índice de figuras

Capítulo 1

Introducción

1.1. Motivación

La robustez es una característica deseable para todo sistema biométrico en general, y para un sistema de verificación de locutor en particular. Es deseable que el sistema sea capaz de reconocer la identidad de una persona no sólo en condiciones ideales, o de laboratorio, si no que también en condiciones reales, en donde existen distintas fuentes de ruido que degradan la señal de voz de los usuarios del sistema.

Se han desarrollado diferentes técnicas para mejorar el rendimiento de los motores de verificación de locutor, que tratan de compensar o cancelar el efecto del canal de comunicación en la señal de voz. En los últimos años también se han propuesto nuevas técnicas de robustez que intentan modelar el efecto del canal, que son catalogadas como técnicas de análisis factorial . En este trabajo se implementarán y compararán dos métodos de análisis factorial: *Joint Factor Analysis* o JFA, y *Total Variability Front-End Factor Analysis* más conocido como *i-Vectors*.

En este trabajo se implementarán y compararán las técnicas de análisis factorial mencionadas, en contexto con el quehacer del Laboratorio de Procesamiento y Transmisión de Voz (LPTV) de la Universidad de Chile, en donde se desarrolló este trabajo.

1.2. Objetivos

El objetivo principal de este trabajo es implementar técnicas de análisis factorial para verificación de locutor como parte del software de verificación texto-independiente del LPTV, y comparar estas técnicas. De acuerdo a esto se fijaron los siguientes objetivos específicos:

- Analizar las técnicas *i-Vectors* y JFA, que son técnicas de análisis factorial para robustez de sistemas de verificación de locutor texto-independiente.
- Describir en detalle los aspectos teóricos de las técnicas seleccionadas.
- Implementar las técnicas seleccionadas como parte del software de verificación texto-independiente del LPTV.

- Realizar pruebas de verificación de locutor utilizando las técnicas seleccionadas, y realizar un análisis crítico de los resultados.

1.3. Estructura de la Memoria

En este trabajo se presenta la implementación y estudio de dos técnicas de análisis factorial para robustez de un sistema de verificación de locutor texto independiente.

En el siguiente capítulo, que es la revisión biblográfica, se revisan los conceptos que dan un marco conceptual a este trabajo. Se iniciará describiendo en forma general la tarea de verificación de locutor texto-independiente. Luego se describirá el modelo GMM-UBM que es el sistema más utilizado como verificador, y que consituye el sistema *baseline* de este trabajo. Al final del capítulo se presentan algunas ténicas para hacer más robusto el sistema, entre ellas los métodos de análisis factorial.

En el capítulo tres se describen en detalle los sistemas *Joint Factor Analysis* e *i-Vectors*. Se describe su fundamentación y los elementos que los componen. Además se detalla de qué manera se entrena cada uno de sus componentes y cómo se realiza la verificación de un locutor en particular.

En el capítulo cuatro se realizan experimentos de verificación de locutor utilizando la base de datos YOHO. Los experimentos son realizados en condiciones libres de distorsión o *clean*, y en condiciones de ruido, que se logra aplicando distorsiones de canal sobre la base de datos, obteniendo así nuevos escenarios de comparación. Los resultados de los experimentos se muestran en tablas con los *Equal Error Rate* (EER), que es una medida porcentual del error cometido en la verificación.

En el capítulo cinco se presentan las conclusiones de este trabajo a la luz de los resultados obtenidos.

Capítulo 2

Revisión Bibliográfica

2.1. Biometría y Verificación de Locutor

Las técnicas de biometría son métodos automáticos de verificación o reconocimiento de la identidad de una persona basándose en una característica fisiológica o de comportamiento [1].

En este marco se encuentra la tarea de verificación de locutor, que es el proceso de verificar la identidad de una persona basada en su señal de voz (huella de voz o *voiceprint*) [2].

Existen dos factores principales por los que este sistema biométrico es tan convincente, comparado con otros sistemas de verificación populares como biometría de huella dactilar o de rostro [3]. El primero es que la voz es una señal natural emitida por los seres humanos, por lo tanto no se considera riesgoso proveer esta información. Por ejemplo, en una comunicación telefónica, proveer una muestra de la voz para realizar la verificación del locutor no se considera (desde el punto de vista del usuario) una etapa separada o invasiva. El segundo es que la red telefónica y la gran disponibilidad de computadores con micrófonos integrados proveen una red ubicua y familiar de sensores para obtener y transmitir la señal de voz.

2.1.1. Producción y Adquisición de Voz

El tracto vocal es el responsable de la articulación de la voz en los humanos. Cuando la señal acústica pasa a través del tracto vocal, el contenido de frecuencia de la señal es alterado por las resonancias en este. Estas resonancias son llamadas formantes. Luego, la forma del tracto vocal de una persona puede ser estimada a partir de la forma del espectro de la señal de voz [4]. Los sistemas de verificación de voz generalmente parametrizan la voz de una persona (o *speaker*) sólo con características o *features* derivadas del tracto vocal.

Para realizar un análisis en frecuencia de la señal de voz en un sistema computacional, es necesario realizar una conversión de la señal acústica producida por el tracto vocal a una señal digital. Se puede utilizar un micrófono convencional o el auricular de un teléfono para convertir la onda acústica en una señal análoga. Luego se debe realizar un filtrado antialiasing para limitar el ancho de banda de la señal a aproximadamente la de Nyquist,

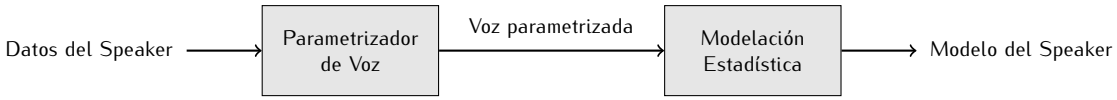

Figura 2.1 Representación de la fase de entrenamiento en un sistema VL

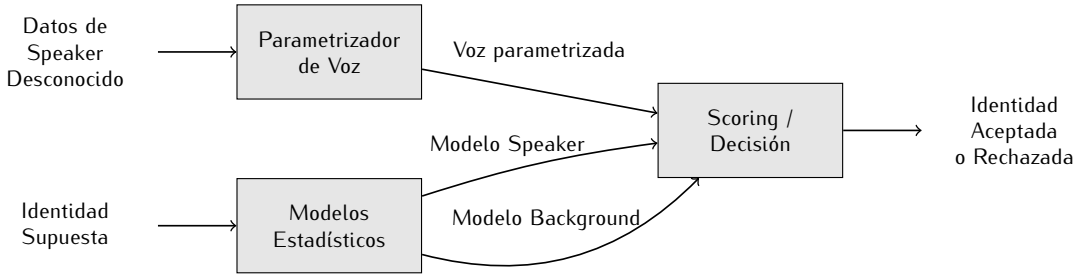

Figura 2.2 Representación de la fase de test en un sistema VL

antes de realizar la etapa de muestreo. Finalmente se muestrea la señal con un conversor análogo-digital para obtener la señal digital.

2.1.2. Métodos de Verificación de Locutor

Los métodos de verificación de locutor se dividen en dos grandes grupos: verificadores texto-dependientes (VL-TD) y verificadores texto-independientes (VL-TI). En VL-TD se requiere que el locutor diga exactamente una secuencia de texto predeterminada. Este tipo de sistemas está basado en modelos estadísticos utilizando Modelos Ocultos de Markov o HMM (por *Hidden Markov Models*). En el caso de VL-TI se verifica la identidad de un usuario sin restringirse al mensaje hablado en la señal de audio. Este tipo de sistemas está tradicionalmente basado en el modelo probabilístico *Gaussian Mixture Model* o GMM, pero también existen otros enfoques como el uso de redes neuronales y *Support Vectors Machines* [3], [5].

2.2. Etapas en la tarea de VL-TI

Cualquier sistema de verificación de locutor está compuesto por dos fases distintas: una fase de entrenamiento o enrolamiento, llamada en la literatura como *training* o *enrolling*, y una etapa de *test* [3], [6]. Cada una de estas fases puede ser descrita como una sucesión de módulos de procesamiento.

La etapa de *training* se observa en la figura 2.1 En el primer bloque se realiza una extracción de características de la señal de voz de una persona o *speaker* cualquiera. El objetivo es obtener una representación apropiada para realizar una modelación estadística de la señal. Este proceso se llama parametrización o *feature extraction* y se detalla en la sección 2.3. El segundo bloque obtiene el modelo estadístico a partir de los parámetros extraídos. La salida de este proceso es un modelo estadístico para cada *speaker* enrolado en el sistema.

La figura 2.2 muestra una representación similar para la fase de test. Las entradas al sistema son la señal de voz de un *speaker* desconocido y la identidad supuesta de ese *speaker*.

Se obtienen los parámetros de la señal de voz con el mismo módulo de parametrización utilizado en la fase de *training*. Por otro lado, se carga el modelo estadístico del *speaker* correspondiente a la identidad supuesta, y un modelo estadístico general llamado *background model*. Finalmente usando estos dos modelos estadísticos más los parámetros de la voz se calculan puntajes o *scores* para decidir aceptar o rechazar la identidad supuesta del *speaker*.

2.3. Parametrización de la voz

El objetivo de parametrizar una señal de voz es caracterizarla en una secuencia temporal de vectores de parámetros. De esta manera se reduce la cantidad de datos a procesar, y al mismo tiempo se retiene información importante contenida en la señal. En esta sección de describe el proceso de una parametrización exitosamente aplicada en verificación de locutor, con la que se obtienen los coeficientes cepstrales Mel o MFCCs.

2.3.1. Frames

La señal de voz cambia continuamente debido a los movimientos articulatorios del tracto vocal, luego es necesario dividir la señal en *frames* de entre 20 y 30 [ms] de duración. Dentro de este intervalo se asume que la señal permanece estacionaria [6], y se calcula un vector de características o *features* para cada *frame*.

El proceso de obtención de *frames* se divide en las siguientes etapas:

1. **Pre-énfasis:** Se aplica un filtro pasa altos para amplificar las altas frecuencias y así compensar el proceso de producción de voz humana que tiende a atenuar las altas frecuencias [7].

2. ***Framing*:** La señal de voz es dividida en segmentos traslapados de duración fija llamados *frames*. En general, la duración de un *frame* es de entre 20 a 30 [ms], y cada *frame* es generado cada 10 [ms]. Luego, dos *frames* consecutivos de 25 [ms] de duración generados cada 10 [ms] estarán traslapados por 15 [ms].

3. **Enventanado:** Finalmente, cada *frame* es multiplicado por una función ventana (generalmente de tipo Hamming). Esta acción es necesaria debido a los efectos de la transformada de Fourier que se aplicará en etapas posteriores. La multiplicación del *frame* por la ventana en el dominio del tiempo es una convolución en el dominio de la frecuencia, y como resultado se obtiene un espectro de frecuencia más suave y menos distorsionado que el espectro del *frame* por sí solo.

2.3.2. Coeficientes Cepstrales

A cada *frame* obtenido en la etapa anterior se realiza una extracción de *features* que caractericen a cada *speaker* de manera particular.

Los coeficientes cepstrales en frecuencia Mel o MFCC (de *Mel-Frequency spaced Cepstral Coefficients*) es una manera de caracterizar la voz basados en la percepción auditiva humana. Este método ha sido ampliamente utilizado las últimas dos décadas para reconocimiento de voz y verificación de locutor [7], [8].

Para obtener los MFCC se realizan los siguientes pasos:

1. **Transformada de Fourier:** Se aplica a cada *frame* la transformada de Fourier rápida o FFT (por *Fast Fourier Transform*), obteniéndose un espectro de magnitud y un espectro de fase del *frame*. La envolvente del espectro de magnitud contiene información sobre las propiedades de resonancia del tracto vocal y se ha establecido que esta es la parte más informativa del espectro para la tarea de verificación de locutor [6]. Por lo anterior, se ignora el espectro de fase y sólo se utiliza el espectro de magnitud.

2. **Banco de Filtros:** En esta etapa se aplica un banco de filtros triangular realizando una estimación de la densidad espectral. El número de filtros depende de la implementación, pero debe ser menor a la cantidad de elementos de la FFT. En esta memoria, la cantidad de elementos del espectro de magnitud de la FFT es de $N = 256$ coeficientes, y la cantidad de filtros es $M = 14$.

   Los centros de los filtros triangulares están espaciados de acuerdo a la escala de frecuencias Mel, definida como:

   $$f_{MEL} = 2595 \log\left(1 + \frac{f_{LIN}}{700}\right) \tag{2.1}$$

   donde f_{LIN} es la frecuencia en escala lineal. Las salidas del banco de M filtros se denota como S_m con $m = 1, ..., M$.

3. **Transformada de coseno:** Como último paso se transforman los S_m con la transformada de coseno discreta o DCT (por *Discrete Cosine Transform*), obteniéndose que los coeficientes cepstrales están dados por:

   $$c_n = \sum_{m=1}^{M} \log(S_m) \cos\left[\frac{\pi n}{M}\left(m - \frac{1}{2}\right)\right] \tag{2.2}$$

   En la ecuación (2.2), n es el índice del coeficiente cepstral. El vector final de coeficientes cepstrales se obtiene manteniendo sólo los primeros 10 a 15 coeficientes de la DCT.

2.3.3. Derivadas temporales

Después de calcular un vector de coeficientes cepstrales que representa a cada *frame*, se agrega a cada vector información de la variación temporal de los coeficientes. Para ello se calculan los parámetros Δ y $\Delta\Delta$ (delta y delta-delta), que son aproximaciones polinomiales de la derivada temporal de primer y segundo orden de cada coeficiente:

$$\Delta c_n = \frac{\sum_{k=1}^{K} k \cdot (c_{n+k} - c_{n-k})}{2\sum_{k=1}^{K} k^2} \tag{2.3}$$

$$\Delta\Delta c_n = \frac{\sum_{k=1}^{K} k \cdot (\Delta c_{n+k} - \Delta c_{n-k})}{2\sum_{k=1}^{K} k^2} \tag{2.4}$$

En esta memoria se utiliza $K = 1$.

El resultado final es que para cada *frame* t se obtiene un vector de características x_t formado por los coeficientes cepstrales y las derivadas temporales, es decir:

$$x_t = [c_1, \ldots, c_N, \Delta c_1, \ldots, \Delta c_N, \Delta\Delta c_1, \ldots, \Delta\Delta c_N] \tag{2.5}$$

Luego, la señal de voz Y de un *speaker* se representa por un conjunto de vectores de características $X = \{x_1, ..., x_T\}$ donde T es la cantidad de *frames*.

2.4. Verificación de Locutor con GMM-UBM

En esta sección de describe el sistema de verificación de locutor llamado GMM-UBM, que es el sistema *baseline* de este trabajo.

2.4.1. Detector de Razón de Verosimilitud

Se ha definido que la verificación de locutor es el proceso de comprobar la identidad de una persona usando su voz. En otras palabras, dada una señal de voz Y, y un locutor o *speaker* S, se debe determinar si Y fue hablado por S.

Si definimos las hipótesis:

$$H_0 : Y \text{ fue hablado por } S$$
$$H_1 : Y \text{ no fue hablado por } S$$

El test óptimo para decidir entre estas dos hipótesis es el test de razón de verosimilitud o *likelihood ratio* [9] definido como:

$$\frac{p(Y|H_0)}{p(Y|H_1)} = \begin{cases} \geq \theta & aceptar\, H_0 \\ < \theta & rechazar\, H_0 \end{cases} \tag{2.6}$$

En la ecuación anterior, θ es el umbral de decisión, y $p(Y|H_i)_{i=0,1}$ es la función de densidad de probabilidad de H_i evaluada para la señal observada Y , llamada también la verosimilitud o *likelihood* de H_i dada la señal Y.

Matemáticamente, la señal de voz Y se representa por el conjunto de vectores de características X, y la hipótesis H_0 se representa por un modelo estadístico λ_{hip} que caracteriza al *speaker* S. La hipótesis alternativa, H_1, se representa por el modelo $\lambda_{\overline{hip}}$. La razón de verosimilitud o *likelihood ratio* es entonces $p(X|\lambda_{hip})/p(X|\lambda_{\overline{hip}})$. Es más común utilizar el logaritmo de esta razón, llamado *log-likelihood ratio* o *LLK ratio*, que es:

$$\Lambda(X) = \log p(X|\lambda_{hip}) - \log p(X|\lambda_{\overline{hip}}) \tag{2.7}$$

El modelo λ_{hip} se entrena a partir de datos de S, mientras que el modelo $\lambda_{\overline{hip}}$ debe representar el espacio de posibles alternativas a S, es decir, representar al resto de posibles *speakers*. Una aproximación a representar ese espacio es elegir un conjunto de modelos de *speakers* que se denominan *background speakers* o *cohorts*. Así, si tenemos un conjunto de N modelos de *cohorts* $\{\lambda_1, ..., \lambda_N\}$, se definirá:

$$p(X|\lambda_{\overline{hip}}) = \mathcal{F}(p(X|\lambda_1), ..., p(X|\lambda_N)) \tag{2.8}$$

donde $\mathscr{F}$ es una función (como máximo o media) de *likelihood* sobre los modelos de *cohorts*.

Otra aproximación es entrenar un sólo modelo que represente el espacio de alternativas, que se entrena a partir de señales de un conjunto de *background speakers*. Este modelo se denomina *Universal Background Model* o UBM. Así, se tendrá que el *LLK ratio* viene dado por:

$$\Lambda(X) = \log p(X|\lambda_{hip}) - \log p(X|\lambda_{UBM}) \tag{2.9}$$

2.4.2. Gaussian Mixture Model

Un modelo de mezclas gaussianas o GMM (por *Gaussian Mixture Model*) es un modelo estadístico ampliamente utilizado en verificación de locutor. Para un vector de características x de dimensión D, la densidad de probabilidad de una mezcla de C gaussianas es:

$$p(x|\lambda) = \sum_{i=1}^{C} w_i p_i(x) \tag{2.10}$$

Como se aprecia en la ecuación anterior, $p(x|\lambda)$ es una combinación lineal ponderada de C densidades gaussianas multivariantes $p_i(x)$, cada una parametrizada por un vector de medias μ_i de dimensión D, y una matriz de varianza Σ_i de dimensión DXD :

$$p_i(x) = \frac{1}{(2\pi)^{D/2}|\Sigma_i|^{1/2}} \exp\left\{-\frac{1}{2}(x-\mu_i)^t \Sigma_i^{-1}(x-\mu_i)\right\} \tag{2.11}$$

Los pesos w_i que ponderan a las gaussianas satisfacen la condición $\sum_{i=1}^{C} w_i = 1$, y en general la matriz de varianza Σ_i es diagonal [3] . En conjunto, los parámetros del GMM se denotan como $\lambda = (w_1,...,w_C,\mu_1,...,\mu_C,...,\Sigma_1,...,\Sigma_C)$.

2.4.3. Entrenamiento de UBM con algoritmo EM

En este trabajo se realiza el entrenamiento del UBM con el algoritmo iterativo *Expectation-maximization*, o EM, a partir de datos *background* definidos como $X_{Bkg} = \{x_1^{Bkg},...,x_N^{Bkg}\}$, consistentes en N vectores de características.

EM parte de un estimado de los parámetros del GMM, λ_0, llamado también modelo inicial. Después, en cada iteración del algoritmo se actualizan estos parámetros usando los datos de entrenamiento, de manera que el *likelihood* del modelo con los datos aumenta, es decir, para las iteraciones k y $k+1$, $p(X_{Bkg}|\lambda_k) \geq p(X_{Bkg}|\lambda_{k+1})$. En esta tesis, el modelo inicial se calcula asignando a cada gaussiana el mismo peso y la misma varianza correspondiente a la varianza global de los datos X_{Bkg}. La media de cada gaussiana se calcula como la media de una muestra aleatoria de los datos X_{Bkg}.

En cada iteración del algoritmo, se calculan los nuevos parámetros de cada gaussiana i según las ecuaciones de EM derivadas para GMM:

$$\hat{w_i} = \frac{1}{N}\sum_{j=1}^{N}\gamma_i(x_j) \tag{2.12}$$

$$\hat{\mu_i} = \frac{\sum_{j=1}^{N}\gamma_i(x_j)\cdot x_j}{\sum_{j=1}^{N}\gamma_i(x_j)} \tag{2.13}$$

$$\hat{\Sigma_i} = \frac{\sum_{j=1}^{N}\gamma_i(x_j)(x_j-\hat{\mu_i})(x_j-\hat{\mu_i})^t}{\sum_{j=1}^{N}\gamma_i(x_j)} \tag{2.14}$$

En las ecuaciones anteriores $\gamma_i(x_j)$ corresponde a la probabilidad posterior o *a posteriori* de que el vector x_j siga la distribución de la gaussiana i, y por el teorema de Bayes, se calcula como la probabilidad *a priori* multiplicada por la función de *likelihood* de la gaussiana i (ver ecuación (2.11)), y normalizada:

$$\gamma_i(x_j) = \frac{w_i \cdot p_i(x_j)}{\sum_{c=1}^{C} w_c \cdot p_c(x_j)} \tag{2.15}$$

El algoritmo EM y la derivación de sus ecuaciones aplicadas a GMM pueden revisarse en detalle en [10].

2.4.4. Adaptación de Modelo de Speaker

Para generar el modelo estadístico de cada *speaker* se utilizan modelos estimados MAP (*Maximum a posteriori*) del modelo UBM. En general, se podría generar cada modelo de *speaker* como se generó el modelo UBM, pero se ha mostrado que tomar como base el modelo UBM, que fue entrenado con una gran cantidad de datos, produce mejores resultados y permite implementar una técnica de scoring rápida [7], [9].

El procedimiento de adaptación es un proceso iterativo. Dados el modelo UBM y los datos de entrenamiento del *speaker* $X_S = \{x_1, ..., x_{N_S}\}$ consistentes en N_S vectores de características, se calcula en cada iteración la probabilidad *a posteriori* $\gamma_i(x_j)$ definida en la ecuación (2.15) para cada gaussiana i y vector x_j. Luego, se calculan estadísticos suficientes definidos por:

$$n_i = \sum_{j=1}^{N_S}\gamma_i(x_j) \tag{2.16}$$

$$E_i(X_S) = \frac{1}{n_i}\sum_{j=1}^{N_S}\gamma_i(x_j)\cdot x_j \tag{2.17}$$

$$E_i(X_S^2) = \frac{1}{n_i}\sum_{j=1}^{N_S}\gamma_i(x_j)\cdot x_j {x_j}^t \tag{2.18}$$

El cálculo de estos estadísticos es equivalente al paso de *expectation* o *E step* del algoritmo EM.

Luego los estadísticos son usados para actualizar los parámetros del UBM en cada iteración, creando parámetros adaptados al *speaker*, utilizando las siguientes ecuaciones:

$$\hat{w_i} = [\alpha_i(n_i/N_S) + (1-\alpha_i)w_i]\delta \tag{2.19}$$

$$\hat{\mu_i} = \alpha_i E_i(X_S) + (1-\alpha_i)\mu_i \tag{2.20}$$

$$\hat{\Sigma_i} = \alpha_i E_i(X_S^2) + (1-\alpha_i)(\Sigma_i - \mu_i {\mu_i}^t) - \hat{\mu_i}{\hat{\mu_i}}^t \tag{2.21}$$

El coeficiente α_i es un factor de adaptación que controla los pesos que tienen los estimados viejos y nuevos. En la ecuación (2.19), $delta$ es un factor de escala que se calcula para cada peso adaptado, para que la suma de los pesos sobre todas las gaussianas sea igual a la unidad. El factor de adaptación α_i se define para cada gaussiana como:

$$\alpha_i = \frac{n_i}{n_i + r} \tag{2.22}$$

donde r es un escalar fijo llamado factor de relevancia de la adaptación MAP.

2.5. Evaluación de un sistema de verificación

En un sistema de verificación de locutor, dada la identidad de un locutor objetivo y una señal de audio, se intenta determinar si esa señal fue pronunciada por el locutor objetivo. En un sistema como el descrito, pueden ocurrir dos tipos de error: Falsa aceptación y falso rechazo. Una falsa aceptación sucede cuando se acepta la identidad del locutor objetivo pero la señal de audio no corresponde a esa identidad. Un falso rechazo ocurre cuando el locutor es quien dice ser pero el sistema no lo acepta. Ambos tipos de error dependen del umbral θ utilizado para tomar la decisión de aceptar o rechazar (ecuación (2.6)).

Si el umbral es muy pequeño, el sistema tenderá a siempre aceptar la identidad presentada, realizando muy pocos falsos rechazos, pero muchas falsas aceptaciones. Por el contrario, si el umbral es alto, el sistema rechazará la identidad la mayoría de las veces, cometiendo pocas faltas aceptaciones pero muchos falsos rechazos. Luego, al definir el umbral θ existe un *trade-off* entre dos tipos de errores.

2.5.1. Curvas DET y EER

A partir de lo que se ha mencionado se pueden definir las tasas de errores de falso aceptación (T_{FA}) y falso rechazo (T_{FR}) en función del umbral θ usado para tomar la decisión en el verificador. Además, es posible representar el rendimiento de un sistema de verificación graficando T_{FA} como función T_{FR}. Esta curva es conocida como característica de operación del sistema, y si se realiza en una escala normalizada, se conoce como curva de *trade-offs* de errores de detección, o curva DET (*detection error trade-offs curve*). La curva DET se ha convertido en una representación estándar del rendimiento de un sistema de verificación y de cualquier sistema que deba tomar una desición binaria [3].

El *Equal Error Rate* o EER es otra medida de rendimiento basado en las tasas de errores T_{FA} y T_{FR}, y corresponde al punto de operación en donde T_{FA} = T_{FR}. El EER es una medida muy utilizada para medir la habilidad que tiene un sistema de verificación para separar

impostores de *speakers* verdaderos [3].

En este trabajo se utiliza como medida de rendimiento el EER.

2.5.2. Normalización de Verosimilitud

La verosimilitud calculada para un locutor puede ser normalizada. Así, las verosimilitudes calculadas para diferentes locutores quedan en un rango similar, y es posible fijar de mejor manera el umbral de decisión del sistema. La normalización se realiza relativa a un conjunto de modelos de locutores que no participan de la verificación, conocidos como impostores o *cohorts*. La normalización de verosimilitud puede corregir algunos desfases que no son compensados por los métodos de extracción de características y de adaptación de modelos [6].

En las ecuaciones (2.7) y (2.8) se definió la razón de verosimilitud o *log-likelihood ratio* (LLR) usando el concepto de *cohorts*. Una manera de normalización que se utiliza en la mayoría de los verificadores de locutor es normalizar con el promedio de las M mejores verosimilitudes de *cohorts* [5]. Luego, dado el conjunto de vectores de características X de una elocución de test y un locutor s, el LLR se calcula como:

$$LLR_{normalizado}(X|\lambda_s) = \log\big(P(X|\lambda_s)\big) - \log\big(\frac{1}{M}\sum_{k=1}^{M} P(X|\lambda_{cohort_k})\big) \tag{2.23}$$

donde λ_s es el modelo del locutor objetivo s y λ_{cohort_k} es el modelo del *cohort* k.

2.6. Técnicas para robustez en sistemas verificación de locutor

En el modelo GMM-UBM que se describió en la sección anterior, se utiliza el algoritmo MAP para adaptar el modelo UBM a un *speaker* específico utilizando datos de entrenamiento. En otras palabras, la adaptación MAP es una forma de modelar la diferencia o variabilidad que existe entre las voces de *speakers* distintos, obteniéndose modelos diferentes para diferentes *speakers*. Esta variabilidad se llama en la literatura variabilidad entre *speakers* o *inter-speaker variability*.

Existe otra variabilidad llamada variabilidad de sesión o *session variability* que tiene que ver con la diferencia entre las condiciones de grabación de los datos de entrenamiento y los datos de test. Si se adapta el modelo para un *speaker* usando datos de entrenamiento grabados en ciertas condiciones, utilizar ese modelo para reconocer al *speaker* bajo condiciones de grabación diferentes puede ser problemático [11].

2.6.1. Cepstral Mean Normalization

Cepstral Mean Normalization o CMN es una técnica que permite compensar el ruido convolucional en las elocuciones que se utilizan en el verificador. El ruido convolucional produce un componente aditivo en el espectro logarítmico de la señal. Dada la forma en que se

calculan los coeficientes cepstrales MFCC, este tipo de ruido aparecerá como una constante aditiva en cada coeficiente del vector de características. Luego, si se resta el promedio de los vectores a cada vector, se está cancelando el canal [12].

Si se tiene una señal representada por un conjunto de $\{x_1,...,x_T\}$ vectores de características, el vector compensado $\tilde{x}_i$ con CMN se calcula como:

$$\tilde{x}_i = x_i - \frac{1}{T}\sum_{j=1}^{T} x_j \tag{2.24}$$

2.6.2. RASTA

RASTA es una técnica que, al igual que CMN, se aplica en el nivel de extracción de características de la señal. La variación temporal de los coeficientes cepstrales también puede ser vista como una señal, que puede ser filtrada. El filtro RASTA es un filtro IIR pasa banda que actúa en el dominio cepstral:

$$H(z) = 0{,}1z^4 \cdot \frac{2 + z^{-1} - z^3 - 2z^{-4}}{1 - 0{,}98z - 1} \tag{2.25}$$

Con este filtro se logra atenuar el efecto de variaciones lentas en las características de frecuencia en la señal acústica que son atribuibles al canal y no a la voz del locutor [13]. Desde este punto de vista, CMN realiza un filtrado pasa altos en el dominio cepstral.

2.6.3. Análisis Factorial

El los últimos años se han propuesto nuevos métodos basados en GMM-UBM que intentan modelar por separado la variabilidad de speaker y sesión con buenos resultados. Estos métodos son catalogados como sistemas de Análisis Factorial o *Factor Analysis*. En este trabajo se implementan dos sistemas *Joint Factor Analysis* o JFA, y *Total Variability Front-End Factor Analysis* más conocido como *i-Vectors* [14] - [15].

En este trabajo se implementarán y estudiarán ambos métodos.

Capítulo 3

Implementación de Métodos de Análisis Factorial

Este capítulo describe en detalle la implementación de los métodos de análisis factorial para verificación de locutor mencionados en el capítulo anterior: *Joint Factor Analysis* y *Total Variability Front-End Factor Analysis*. Se muestra como es posible derivar estos modelos desde el sistema GMM-UBM, y la manera en que se realiza el entrenamiento de los modelos y el test de verificación.

3.1. Joint Factor Analysis

Joint Factor Analysis intenta modelar por separado el efecto de la variabilidad entre *speakers* y la variabilidad de canal. En esta sección se explica la definición del modelo y los algoritmos de entrenamiento y test.

3.1.1. Factores de *Speaker* y Sesión

Si nombramos el vector de medias del modelo del *speaker* s como $M(s)$ y el vector de medias del modelo *background* como m_{UBM}, se podría representar la adaptación MAP que realiza el sistema GMM-UBM como una operación matricial:

$$M(s) = m_{UBM} + Dz(s) \tag{3.1}$$

donde D es una matriz diagonal, y $z(s)$ es un vector oculto que sigue una distribución normal $\mathcal{N}(\mathbf{0}, I)$.

Joint Factor Analysis añade dos elementos más a este modelo. Se asume que existe una matriz V de rango bajo que representa la variabilidad entre *speakers* y una matriz U de rango bajo que representa la variabilidad de sesión. Luego, el vector de medias del *speaker* s se puede escribir como:

$$M(s) = m_{UBM} + U \cdot x(s,r) + V \cdot y(s) + D \cdot z(s) \tag{3.2}$$

donde $x(s,r)$ es un vector oculto llamado factor de sesión, factor de canal o *channel factor* y $y(s)$ es un vector oculto llamado factor de locutor, factor de *speaker* o *speaker factor*. Ambos factores siguen distribuciones normales. El factor de *speaker* y depende del *speaker* s,

mientras que el factor de canal x depende tanto s como de las condiciones de la sesión de grabación, representadas en la ecuación por r .

Las columnas de las matrices V y U son conocidas como *eigenvoices* y *eigenchannels*, respectivamente, mientras que la cantidad de filas de ambas matrices es igual a la cantidad de gaussianas del UBM multiplicada por el tamaño del vector de *features*. La cantidad de *eigenvoices* y *eigenchannels* son parámetros de diseño del sistema, y determinan no solo el tamaño de las matrices V y U, si no que corresponden al tamaño de los factores de *speaker* y canal, respectivamente.

El modelo de la ecuación (3.2) puede ser reescrito para poner énfasis en la separación entre variabilidad *speaker* y variabilidad de sesión o canal:

$$M(s) = M_s(s) + M_c(s,r) \tag{3.3}$$

$$M_s(s) = m_{UBM} + V \cdot y(s) + D \cdot z(s) \tag{3.4}$$

$$M_c(s,r) = U \cdot x(s,r) \tag{3.5}$$

En otras palabras, el vector $M(s)$ que representa al *speaker* es la suma de dos vectores: $M_s(s)$ que representa la variabilidad de *speaker*, y $M_c(s,r)$ representa la variabilidad de sesión o canal [16].

De las ecuaciones se desprende que para adaptar el modelo de un locutor en particular, es necesario conocer el vector z y los factores x e y que representan a ese *speaker*.

3.1.2. Entrenamiento del Modelo

En el modelo base GMM-UBM, el modelo background es una mezcla de gaussianas multivariantes entrenado a partir de datos de locutores que no participarán en el reconocedor. Supongamos que dicho modelo es entrenado, y que está compuesto por C gaussianas multivariante, y que la dimensión de los vectores de *features* es F.

En base a ese modelo, se define m_{UBM} de la ecuación (3.2) como la concatenación de los vectores de medias de cada gaussiana del UBM:

$$m_{UBM} = [\mu_1; \mu_2; ...; \mu_{C-1}; \mu_C] \tag{3.6}$$

luego m_{UBM} es un vector de tamaño CF.

Los elementos restantes de la ecuación (3.2) que son comunes a todos los *speakers* y que deben ser entrenados con datos de locutores que no participarán en el reconocedor son las matrices V, U y D.

3.1.2.1. Estadísticos de Baum-Welch

Dado un *speaker* s y sus datos de entrenamiento representados por una secuencia de T vectores de características acústicas $x_1, x_2, \ldots, x_T$, se definen los estadísticos de Baum-Welch de orden cero y uno, para cada componente c de la mezcla de gaussianas del modelo UBM, de acuerdo a [15]:

$$N_c(s) = \sum_{t=1}^{T} \gamma_c(x_t) \tag{3.7}$$

$$F_c(s) = \sum_{t=1}^{T} \gamma_c(x_t) \cdot x_t \tag{3.8}$$

donde $\gamma_c(x_t)$ es la probabilidad posterior definida en (2.15) para el vector x_t. Es importante notar que el estadístico de orden cero $N_c(s)$ es un escalar, mientras el estadístico de orden uno $F_c(s)$ es un vector de tamaño F.

Además, se define el estadístico centrado de Baum-Welch de orden uno como:

$$\tilde{F}_c(s) = \sum_{t=1}^{T} \gamma_c(x_t) \cdot (x_t - \mu_C) = F_c(s) - N_c(s) \cdot \mu_c \tag{3.9}$$

donde μ_c es el vector de medias de la gaussiana c del modelo UBM.

Por último, es necesario definir:

$$N(s) = \begin{bmatrix} N_1(s) \cdot I_F & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & N_C(s) \cdot I_F \end{bmatrix} \tag{3.10}$$

$$\tilde{F}(s) = [\tilde{F}_1(s); \cdots ; \tilde{F}_C(s)] \tag{3.11}$$

En la ecuación (3.10), $N(s)$ es una matriz diagonal de tamaño $CF \times CF$ cuya diagonal está formada por las matrices diagonales $N_c(s) \cdot I_F$ con $c = 1, ..., C$, y I_F es la matriz identidad de tamaño F.

En la ecuación (3.11), $\tilde{F}(s)$ es un vector de tamaño CF resultante de la concatenación de los C estadísticos centrados de orden uno para el *speaker* s.

3.1.2.2. Estimación de la matriz V

La estimación de los parámetros del modelo JFA presentado en la ecuación (3.2) es un proceso iterativo de tipo *Expectation-maximization*, es decir, en cada iteración se calcula un estimado de la variable de interés, y luego se modifica maximizando la verosimilitud de esa variable con los datos de entrenamiento.

La primera matriz que se debe estimar es la matriz V, y el entrenamiento se realiza asumiendo que las matrices D y U son iguales a cero. Con esta condición la ecuación (3.2) se reduce a

$$M(s) = m_{UBM} + V \cdot y(s) \tag{3.12}$$

El algoritmo de entrenamiento necesita una inicialización de la matriz V (p. ej. aleatoria) y datos de entrenamiento *background*, es decir, que correspondan a un conjunto de *speakers*

que no serán verificados por el sistema, conocido como *impostores*. Después de la inicialización, el algoritmo entra en un proceso iterativo que permitirá maximizar la verosimilitud de la matriz V con los datos de entrenamiento.

En cada iteración se calcula el valor esperado del factor $y(s)$ para cada impostor s , y luego se usan esos estimados para actualizar el valor de V. Para cada impostor s se define la matriz $l(s)$ como:

$$l_V(s) = I + V^t \cdot \Sigma^{-1} \cdot N(s) \cdot V \tag{3.13}$$

Con la definición anterior, se tiene que la distribución posterior del factor $y(s)$ condicionado a los datos de entrenamiento es una gaussiana multivariante de media $l_V^{-1}(s) \cdot V^t \cdot \Sigma^{-1} \cdot \tilde{F}(s)$ y matriz de covarianza $l_V^{-1}(s)$ [17]. Luego, si se nota $E[\cdot]$ como el valor esperado o *expectation*, se tendrá que $E[y(s)]$ representa la media *a posteriori* del factor $y(s)$, y que $E[y(s) \cdot y^t(s)]$ representa su matriz de covarianza *a posteriori*. Es decir, es posible estimar $y(s)$ utilizando la siguiente expresión:

$$\bar{y}(s) = E[y(s)] = l_V^{-1}(s) \cdot V^t \cdot \Sigma^{-1} \cdot \tilde{F}(s) \tag{3.14}$$

Una vez se ha estimado el valor esperado del factor $y(s)$ para cada impostor s, se procede a actualizar el valor de V. Si $\mathscr{P}$ es el conjunto de impostores, se define la matriz $\mathfrak{C}$ y las matrices $\mathfrak{A}_c$ para cada gaussiana c según las siguiente ecuaciones :

$$\mathfrak{A}_c = \sum_{s \in \mathscr{P}} N_c(s) E[y(s) \cdot y^t(s)], \quad c = 1, \ldots, C \tag{3.15}$$

$$\mathfrak{C} = \sum_{s \in \mathscr{P}} \tilde{F}(s) \cdot E[y^t(s)] \tag{3.16}$$

Con estas definiciones, V se actualiza según la siguiente ecuación :

$$V = \begin{bmatrix} V_1 \\ \vdots \\ V_C \end{bmatrix} = \begin{bmatrix} \mathfrak{A}_1^{-1} \\ \vdots \\ \mathfrak{A}_C^{-1} \end{bmatrix} \cdot \begin{bmatrix} \mathfrak{C}_1 \\ \vdots \\ \mathfrak{C}_C \end{bmatrix} \tag{3.17}$$

donde V_c y $\mathfrak{C}_c$ corresponden a bloques consecutivos dentro de las matrices V y $\mathfrak{C}$ asociados con la gaussiana c, y cada uno compuesto por F filas (F es el tamaño del vector de *features*).

3.1.2.3. Estimación de la matriz U

En esta sección se explica el proceso de estimar la matriz U de las ecuaciones (3.2) y (3.5), que representa la variabilidad de canal.

Combinando las ecuaciones (3.3) y (3.5) , se puede resumir el modelo JFA como:

$$M(s) = M_s(s) + U \cdot x(s, r_s) \tag{3.18}$$

Esta ecuación es muy parecida a la ecuación (3.12), cuando se asumió D y U iguales a $\mathbf{0}$, lo que sugiere que se puede ocupar aquí el mismo método aplicado para el entrenamiento de V, con algunas diferencias. La más importante es que los estadísticos de Baum-Welch deben

calcularse para cada elocución de entrenamiento, y no para cada *speaker*. Es decir, para un *speaker* s y su elocución r_s definida por una secuencia de K vectores de características $x_1,...,x_K$, se redefinen los estadísticos de orden cero y uno como:

$$N_c(s,r_s) = \sum_{k=1}^{K} \gamma_c(x_k) \tag{3.19}$$

$$F_c(s,r_s) = \sum_{k=1}^{K} \gamma_c(x_k) \cdot x_k \tag{3.20}$$

La otra diferencia es que el caso de la matriz V, los estadísticos $\tilde{F}c(s)$ están centrados alrededor de las medias del UBM , pero en el caso de la matriz U, el estadístico de primer orden centrado para un *speaker* s y su elocución Y_s se define como:

$$\tilde{F}_c(s,r_s) = F_c(s,r_s) - N_c(s,r_s) \cdot (\mu_c + V \cdot y(s)) \tag{3.21}$$

Es decir, el estadístico está centrado alrededor del vector $M_s(s)$. Luego, para calcular estos estadísticos es necesario contar con la matriz V ya entrenada, y estimar los factores $y(s)$ para cada impostor utilizado en el entrenamiento de U.

En cada iteración del algoritmo se estiman los factores $x(s)$ que luego son utilizados para actualizar los valores de U. Para cada impostor s se define la matriz $l_U(s)$ como:

$$l_U(s,r_s) = I + U^t \cdot \Sigma^{-1} \cdot N(s,r_s) \cdot U \tag{3.22}$$

Y se tiene que la distribución posterior del factor de canal $x(s)$ condicionado a los datos de entrenamiento es una gaussiana multivariante de media $l_U^{-1}(s,r_s) \cdot U^t \cdot \Sigma^{-1} \cdot \tilde{F}(s,r_s)$, y matriz de covarianza $l_U^{-1}(s,r_s)$. Luego, el valor estimado de $x(s,r_s)$ es:

$$\bar{x}(s,r_s) = E[x(s,r_s)] = l_U^{-1}(s,r_s) \cdot U^t \cdot \Sigma^{-1} \cdot \tilde{F}(s,r_s) \tag{3.23}$$

Al igual que en el entrenamiento de V, si $\mathscr{P}$ es el conjunto de impostores, y si $\mathscr{R}$ es el conjunto de elocuciones de los impostores , se define la matriz $\mathfrak{C}$ y las matrices $\mathfrak{A}_c$ para cada gaussiana c como:

$$\mathfrak{A}_c = \sum_{s \in \mathscr{P}, r_s \in \mathscr{R}} N_c(s,r_s) \cdot l_U^{-1}, \quad c = 1, \cdots, C \tag{3.24}$$

$$\mathfrak{C} = \sum_{s \in \mathscr{P}, r_s \in \mathscr{R}} \tilde{F}(s,r_s) \cdot \bar{x}(s,r_s) \tag{3.25}$$

Finalmente, en cada iteración se actualiza la matriz U resolviendo:

$$U = \begin{bmatrix} U_1 \\ \vdots \\ U_C \end{bmatrix} = \begin{bmatrix} \mathfrak{A}_1^{-1} \\ \vdots \\ \mathfrak{A}_C^{-1} \end{bmatrix} \cdot \begin{bmatrix} \mathfrak{C}_1 \\ \vdots \\ \mathfrak{C}_C \end{bmatrix} \tag{3.26}$$

donde U_c y $\mathfrak{C}_c$ corresponden a bloques consecutivos dentro de las matrices U y $\mathfrak{C}$ asociados con la gaussiana c, y cada uno compuesto por F filas (F es el tamaño del vector de *features*).

3.1.2.4. Estimación de la matriz D

El último paso para entrenar el modelo JFA es determinar la matriz diagonal D. El entrenamiento se realiza con datos *speaker* tomados de un conjunto $\mathscr{P}$ de impostores, tal como en los casos anteriores.

El proceso de entrenamiento consiste en determinar en cada iteración estimados de los factores $z(s)$ para cada *speaker* s en $\mathscr{P}$, y luego utilizar esos estimados para actualizar los valores de D.

Al igual que en el caso del entrenamiento de la matriz U, es necesario redefinir los estadísticos centrados de primer orden. De acuerdo a la ecuación (3.2), los estadísticos deben ser centrados alrededor de $m_{UBM} + U \cdot x(s, r_s) + V \cdot y(s)$, donde r_s es una elocución de s. Para poder calcular estos estadísticos centrados, es necesario determinar los factores $y(s)$ y $x(s, r_s)$ para cada *speaker*. El factor $y(s)$ se estima usando la ecuación (3.14), considerando todos los datos de entrenamiento, es decir, todas las elocuciones del *speaker* s. En cambio, para cada elocución r_s que conforma la base de datos se calcula un $x(s, r_s)$, utilizando la ecuación (3.23).

Luego, si $\mathscr{R}_s$ es el conjunto de elocuciones del *speaker* s se redefinen el estadístico centrado de primer orden para la gaussiana c como:

$$\tilde{F}_c(s) = F_c(s) - N_c(s) \cdot (\mu_c + V \cdot y(s)) - \sum_{r \in \mathscr{R}_s} N_c(s, r) \cdot U \cdot x(s, r) \tag{3.27}$$

donde $N_c(s, r)$ se calcula según la ecuación (3.19).

Con los estadísticos adecuadamente definidos y estimados se procede con el entrenamiento iterativo. Para cada impostor s se define la matriz $l_D(s)$ como:

$$l_D(s) = I + D^2 \cdot \Sigma^{-1} \cdot N(s) \tag{3.28}$$

Se tiene que con la definición anterior, la distribución posterior del factor residual $z(s)$ condicionado a los datos de entrenamiento es una gaussiana multivariante de media $l_D^{-1}(s) \cdot D \cdot \Sigma^{-1} \cdot \tilde{F}(s)$, y matriz de covarianza $l_D^{-1}(s)$. Luego, el valor estimado de $z(s)$ es:

$$\bar{z}(s) = E[z(s)] = l_D^{-1}(s) \cdot D \cdot \Sigma^{-1} \cdot \tilde{F}(s) \tag{3.29}$$

Una vez obtenido un estimado para $z(s)$ se debe actualizar los valores de D. Si $\mathscr{P}$ es el conjunto de impostores, se definen las matrices $\mathfrak{a}$ y $\mathfrak{b}$ como:

$$\mathfrak{a} = \sum_{s \in \mathscr{P}} \operatorname{diag}\big(N(s) \cdot E[z(s) \cdot z^t(s)]\big) \tag{3.30}$$

$$\mathfrak{b} = \sum_{s \in \mathscr{P}} \operatorname{diag}\big(\tilde{F}(s) \cdot E[z^t(s)]\big) \tag{3.31}$$

Finalmente, definiendo $i = 1, \ldots, CF$, con C la cantidad de gaussianas del UBM y F el tamaño del vector de features, entonces cada elemento d_i de la matriz diagonal D se actualiza resolviendo:

$$d_i \cdot \mathfrak{a}_i = \mathfrak{b}_i \tag{3.32}$$

3.1.3. Adaptación de Modelos de *Speakers*

El modelo JFA para un *speaker* se ha definido en la ecuación (3.2), en donde las matrices V y D definen el subespacio del *speaker* y la matriz U representa el subespacio de la sesión [14]. En otras palabras, V representa el espacio de variabilidad de *speakers*, D es una matriz diagonal residual y U representa el espacio de variabilidad de diferentes condiciones de sesión de grabación de las elocuciones utilizadas en el sistema.

En la sección anterior se describió como se realiza la estimación de estas tres matrices, utilizando datos *background* de impostores, que son elocuciones de locutores que no participarán de la verificación. Una vez que se ha obtenido el valor de estas matrices, es posible realizar la estimación de los modelos de cada *speaker* que participa de la verificación. Esta estimación o adaptación del modelo se reduce a estimar los factores x, y y z de la ecuación (3.2).

A partir de datos de entrenamiento de un *speaker* s, se calculan los estadísticos de orden cero y de orden uno centrados, según las definiciones de las ecuaciones (3.7), (3.8) y (3.9). Se calculan las matrices $l_V(s)$ y $l_U(s)$ según las ecuaciones (3.13) y (3.22), y con ellas se estiman los factores x e y como:

$$y(s) = l_V^{-1}(s) \cdot V^t \cdot \Sigma^{-1} \cdot \tilde{F}(s) \tag{3.33}$$

$$x(s, r_s) = l_U^{-1}(s) \cdot U^t \cdot \Sigma^{-1} \cdot \tilde{F}(s, r_s) \tag{3.34}$$

Estas ecuaciones son similares a las ecuaciones (3.14) y (3.23), respectivamente.

Para calcular el factor z, se calcula $l_D(s)$ según la ecuación (3.28), y se calcula un estadístico de orden uno centrado según la siguiente ecuación:

$$\tilde{F}_c(s) = F_c(s) - N_c(s) \cdot (\mu_c + V \cdot y(s) + U \cdot x(s)) \tag{3.35}$$

Finalmente, se calcula el factor z del *speaker* como:

$$z(s) = l_D^{-1}(s) \cdot D \cdot \Sigma^{-1} \cdot \tilde{F}(s) \tag{3.36}$$

3.1.4. Realización de la verificación

La verificación de locutor consiste decidir si una elocución Y de test fue hablada por un locutor s, que llamaremos locutor objetivo o *speaker* objetivo. Como se explica en la sección 2.4 el sistema *baseline* GMM-UBM utiliza el *log-likelihood ratio* o LLR para tomar esta decisión La estrategia seguida en ese sistema es realizar un análisis *frame* a *frame* de la señal Y contra el modelo de s.

En JFA, el modelo del locutor objetivo se puede escribir como:

$$M(s) = s_o + U \cdot x \tag{3.37}$$

En la ecuación anterior, $s_o = M_s(s)$ según la definición de la ecuación (3.1.1) y representan la variabilidad de *speaker*, mientras $U \cdot x$ representa la variabilidad de canal. Con estas definiciones, dada una elocución Y, la probabilidad que haya sido hablada por el locutor objetivo s según [18] y [19] es:

$$P(Y|s_o) = \int P(Y|s_o,x)\mathcal{N}(x|\mathbf{0},I)dx \tag{3.38}$$

puesto que el factor x sigue una distribución normal con media $\mathbf{0}$ y varianza la matriz identidad. Se debe integrar sobre el factor x porque representa la variabilidad de canal y no es dependiente del *speaker*. En otras palabras, uno espera que los factores y y z para un *speaker* no varíen de una elocución a otra para ese *speaker*, pero el factor x será diferente para cada elocución.

Dada la complejidad de evaluar la expresión (3.38) para cada *frame* de la señal, se realiza una aproximación utilizando los estadísticos de Baum-Welch calculados para la señal de *test*. El logaritmo de la probabilidad estimada viene dada por la siguiente expresión:

$$\log\tilde{P}(Y|s_o) = \sum_{c=1}^{C} N_c \log\frac{1}{(2\pi)^{F/2}|\Sigma_c|^{1/2}} - \frac{1}{2}\mathrm{tr}(\Sigma^{-1}S(s_o)) - \frac{1}{2}\log|L| + \frac{1}{2}\|L^{1/2}U^t\Sigma^{-1}F(s_o)\|^2 \tag{3.39}$$

En el primer término de la ecuación (3.39), C es el número de gaussianas, F es el número de *features* y Σ_c es la matriz de varianza asociada a la gaussiana c. El valor del primer término será igual para el UBM y para el modelo del *speaker* objetivo, por lo tanto se anulará cuando se calcule el LLR.

En el segundo término, Σ es una matriz diagonal por bloques, donde cada bloque es la matriz de varianza de cada gaussiana. $S(s_o)$ es el estadístico Baum-Welch de segundo orden centrado en el *speaker* objetivo definido como:

$$S(s_o) = S - 2\mathrm{diag}(F(s_o)^t) + \mathrm{diag}(N(s_o)^t) \tag{3.40}$$

donde S es el estadístico de Baum-Welch de segundo orden, y $N(s_o)$ y $F(s_o)$ se definen según las ecuaciones (3.10) y (3.11).

Finalmente, en el tercer término de la ecuación (3.39), $L = U^t\Sigma{-}1N(s_o)U$.

A pesar que se obtiene una expresión más manejable para calcular el LLR, es posible realizar algunas aproximaciones extras que permiten obtener una expresión aún más simple. Si se considera que el factor de canal x es conocido, no es necesario realizar la integración. Aún más, si consideramos que el *shift* realizado por el factor de canal x es igual tanto para el modelo objetivo como para el UBM, entonces el factor de canal para la elocución Y se puede estimar con el UBM [20]. Con estas simplificaciones se llega a una buena aproximación para el LLR llamado *linear scoring* o *scoring* lineal, que es ampliamente utilizado en estudios realizados con el sistema JFA. La expresión del LLR lineal para una elocución Y y un *speaker* objetivo s_o es:

$$LLR_{lineal}(Y|s_o) = \big(Vy(s_o) + Dz(s_o)\big)\Sigma^{-1}\big(F_Y - N_Y m_{UBM} + N_Y Ux(Y))\big) \tag{3.41}$$

En la ecuación anterior, $y(s_o)$, $z(s_o)$ son los factores del modelo, $x(Y)$ es el factor de canal calculado para la elocución Y, y N_Y y F_Y son los estadísticos de Baum-Welch de orden cero y uno calculados para la elocución Y.

3.2. i-Vectors

Como se mostró al principio de este capítulo, la idea del sistema JFA es definir dos espacios de variabilidad: locutor y canal. En el sistema *Front-End Factor Analysis* [14], más conocido como i-Vectors, se define un sólo espacio de variabilidad, llamado *espacio de variabilidad total* o *total variability space*. Este nuevo espacio contiene las variabilidades de *speaker* y canal en forma simultánea.

Este nuevo enfoque fue motivado por los experimentos en [21], donde se muestra que los factores de canal calculados para el sistema JFA, que se supone modelan sólo variaciones en el canal, también tenían información de locutor.

En la ecuación (3.2) se define el vector de medias del modelo de un speaker s para JFA. Esta ecuación se puede reescribir para i-Vectors como:

$$M(s) = m_{UBM} + T \cdot w(s) \tag{3.42}$$

donde T es una matriz rectangular de rango bajo y w es un vector aleatorio que sigue una distribución normal $\mathcal{N}(\mathbf{0}, I)$. Los componentes del vector w son los factores *totales*, en contraposición a los factores de *speaker* y canal de JFA. Estos nuevos vectores son llamados *identity vectors* o bien i-vectors, de ahí que el sistema se conozca como i-Vectors.

El proceso de entrenar la matriz de variabilidad total T es exactamente igual a entrenar la matriz V, excepto por una importante diferencia [14]: en JFA, se utilizaban todas las elocuciones del set de entrenamiento de un speaker para obtener sus factores, en cambio, en este sistema se calcula un factor identidad o i-vector para cada elocución.

3.2.1. Entrenamiento de matriz de Variabilidad Total

La matriz de variabilidad total T se entrena con un proceso iterativo de tipo EM. Los datos de entrenamiento consisten en elocuciones de locutores que no participarán de la verificación, llamados *impostores*.

Luego de inicializar la matriz T, se calculan los estadísticos de Baum-Welch de orden cero y uno para cada elocución u de entrenamiento. A diferencia del entrenamiento de la matriz V de JFA, se calcula un estadístico para cada elocución. Visto de otro modo, es como si cada elocución fuera pronunciada por un *speaker* diferente. Luego, para una elocución u definida por una secuencia de K vectores de características $x_1, ..., x_K$, se tiene que los estadísticos de orden cero y uno son:

$$N_c(u) = \sum_{k=1}^{K} \gamma_c(x_k) \tag{3.43}$$

$$F_c(u) = \sum_{k=1}^{K} \gamma_c(x_k) \cdot x_k \tag{3.44}$$

donde $\gamma_c(x_k)$ es la probabilidad posterior de x_k dado la gaussiana c, definida en (2.15). Además, se define el estadístico de orden uno centrado en el UBM como:

$$\tilde{F}_c(u) = \sum_{k=1}^{K} \gamma_c(x_k) \cdot (x_k - \mu_C) = F_c(u) - N_c(u) \cdot \mu_c \tag{3.45}$$

donde μ_c es el vector de medias del UBM correspondiente a la gaussiana c. Además, se necesita definir las siguientes matrices:

$$N(u) = \begin{bmatrix} N_1(u) \cdot I_F & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & N_C(u) \cdot I_F \end{bmatrix} \tag{3.46}$$

$$\tilde{F}(u) = [\tilde{F}_1(u); \cdots ; \tilde{F}_C(u)] \tag{3.47}$$

Con estos estadísticos calculados se entra en el proceso iterativo. En cada iteración, se estima el factor w (o i-vector) para cada elocución u, según la siguiente ecuación:

$$\bar{w}(u) = \left(I + T^t \cdot \Sigma^{-1} \cdot N(u) \cdot T\right)^{-1} \cdot T^t \cdot \Sigma^{-1} \cdot \tilde{F}(u) \tag{3.48}$$

Para realizar la actualización de la matriz T se definen las matriz $\mathfrak{C}$ y las matrices $\mathfrak{A}_c$ para la gaussiana c en forma similar al entrenamiento de la matriz V de JFA. Si $\mathscr{U}$ es el conjunto de elocuciones de entrenamiento, se tiene:

$$\mathfrak{A}_c = \sum_{u \in \mathscr{U}} N_c(u) \cdot \bar{w}(u) \cdot \bar{w}^t(u), \quad c = 1, \ldots, C \tag{3.49}$$

$$\mathfrak{C} = \sum_{u \in \mathscr{U}} \tilde{F}(u) \cdot \bar{w}^t(u) \tag{3.50}$$

Con estas definiciones, T se actualiza según la siguiente ecuación :

$$T = \begin{bmatrix} T_1 \\ \vdots \\ T_C \end{bmatrix} = \begin{bmatrix} \mathfrak{A}_1^{-1} \\ \vdots \\ \mathfrak{A}_C^{-1} \end{bmatrix} \cdot \begin{bmatrix} \mathfrak{C}_1 \\ \vdots \\ \mathfrak{C}_C \end{bmatrix} \tag{3.51}$$

donde T_c y $\mathfrak{C}_c$ corresponden a bloques consecutivos dentro de las matrices T y $\mathfrak{C}$ asociados con la gaussiana c, y cada uno compuesto por F filas (F es el tamaño del vector de *features*).

3.2.2. Extracción de i-vectors

Una vez entrenada la matriz de variabilidad total T se procede a realizar la adaptación de los modelos de locutores que participarán de la verificación. Si observamos la ecuación (3.42), adaptar el modelo del locutor se reduce a estimar el i-vector correspondiente a cada locutor.

Para un *speaker* s y sus datos de entrenamiento, representados por una secuencia de T vectores de características acústicas x_1, x_2, ... , x_T, se calculan los estadísticos Baum-Welch de orden cero y uno, y de orden uno centrados en el UBM, de acuerdo a las ecuaciones (3.7), (3.8) y (3.9). Además, se calculan $N(s)$ y $\tilde{F}(s)$ según las ecuaciones (3.10) y (3.11).

Finalmente el i-vector para s viene dado por:

$$w(s) = \left(I + T^t \cdot \Sigma^{-1} \cdot N(s) \cdot T\right)^{-1} \cdot T^t \cdot \Sigma^{-1} \cdot \tilde{F}(s) \tag{3.52}$$

3.2.3. Realización de la verificación

La verificación de locutor consiste decidir si una elocución u de test fue hablada por un locutor s, que llamaremos locutor objetivo o *speaker* objetivo.

En este sistema, basta con calcular el i-vector de la elocución u de test, y compararla con el i-vector que representa al *speaker* objetivo s. Una medida de la similitud de estos dos vectores es el coseno del ángulo que forman. Si el coseno es 1, significa que los vectores son exactamente iguales; si es -1 , que son exactamente opuestos. Si el coseno es 0, significa que los vectores son independientes. Luego, esta función asigna un puntaje o *score* que reemplaza a la probabilidad de que la elocución u fue hablada por el *speaker* objetivo s, $P(u|s)$.

Luego, dado un umbral de decisión θ, si w_{test} es el i-vector de la elocución u de test, y w_{obj} es el i-vector del *speaker* objetivo, entonces la verificación resulta positiva si:

$$\frac{\langle w_{obj}, w_{test} \rangle}{\|w_{obj}\| \|w_{test}\|} \geq \theta \tag{3.53}$$

Este tipo de *scoring* se conoce como *Cosine Distance Scoring* [22].

Capítulo 4

Comparación de Métodos de Análisis Factorial

En esta sección se realizan pruebas con los sistemas descritos en el capítulo anterior. En la primera parte se detalla la naturaleza de los experimentos a realizar. Luego, se muestra la validación de los sistemas implementados comparándolos con el software de referencia ALIZE. Finalmente, se analiza la incidencia que tiene variar el tamaño de los factores de *speaker*, canal y variabilidad total en el rendimiento del sistema.

4.1. Configuración de Experimentos

4.1.1. Base de datos

La base de datos YOHO [23] es un *corpus* de archivos de audio que es habitualmente usado en tareas de verificación de locutor. Consiste en personas hablando en inglés diciendo tres números de dos dígitos (p. ej. 35-72-41 es pronunciado *thirty-five seventy-two forty-one*). Las grabaciones son archivos PCM de un canal a una tasa de 8000 muestras por segundo, grabados en estudio. Hay 138 locutores: 108 hombres y 30 mujeres. Para cada locutor, existen 4 sesiones para entrenamiento con 24 elocuciones cada una, y 10 sesiones para test de 4 elocuciones cada una; es decir, un total de 136 elocuciones en 14 sesiones por locutor.

En este trabajo se utilizó una base de datos reducida, debido al largo tiempo de procesamiento de la base de datos completa. La base de datos reducida consta de 30 locutores que se usan como clientes, es decir, que participarán en la verificación de identidad, y 40 locutores que se utilizan como impostores, como se resume en la tabla 4.1. Para cada cliente se seleccionan 24 elocuciones de entrenamiento y 16 elocuciones de test. Para los impostores solo se utilizan 24 elocuciones de entrenamiento. Estos datos de impostores son utilizados para calcular el UBM, las matrices V, U y D de JFA, la matriz T de *i-Vectors*, y aplicar la normalización de verosimilitud (*cohorts*).

Con esta configuración de la base de datos, las condiciones de canal de las elocuciones de entrenamiento y de test son prácticamente las mismas. Este escenario es el ideal, pero en la práctica no se cumple, por tanto se introducen dos distorsiones a la base de test para reproducir otro tipo de escenarios con *mismatch* de canal, obteniéndose dos bases de test adicionales.

Tabla 4.1. Configuración Base de Datos

	Número de Locutores	Señales de Entrenamiento	Señales de Test
Clientes	30	24	16
Impostores	40	24	-

- *Tilt 6dB*. Consiste en aplicar un *tilt* espectral de -6 [dB] por octava. Esta distorsión simula una condición de grabación donde se utilizó un micrófono distante, fuera del eje u ocluido [24].
- *Tilt Dyn 6dB*. Consiste en aplicar un *tilt* espectral que varía en el tiempo, comenzando en 0 [dB] por octava al inicio de la elocución y terminando con -6 [dB] por octava al final de la elocución. Este distorsión simula un canal que no es estático, si no que tiene variaciones lentas en el tiempo [24].

Las bases de entrenamiento y test sin distorsión se denominan *clean*.

4.1.2. Verificaciones de locutor

En cada experimento se realizan verificaciones de dos tipos:

- *Cliente consigo mismo*. Se realiza la verificación de cada elocución de la base de test de un cliente contra la identidad de ese cliente. El error que se puede cometer en este tipo de verificación es un falso rechazo.
- *Cliente contra los demás*. Se realiza la verificación de cada elocución de la base de test de un cliente contra las identidades de todos los demás clientes. El error que se puede cometer en este tipo de verificación es una falsa aceptación.

Para las verificaciones de cliente consigo mismo se utilizan las 16 señales de test, mientras que para las verificaciones de cliente contra todos se utilizan sólo 6 señales. Si se utilizan *cohorts* en el experimento se debe calcular además la verosimilitud de cada señal de test de cada cliente con los modelos de cada impostor, elevando la cantidad de verificaciones totales. En la tabla 4.2 se muestra la cantidad de verificaciones que se realiza en un experimento completo. En un experimento sin *cohorts* se necesitan realizan 5.700 verificaciones, mientras que en un caso con *cohorts* se deben llevar a cabo 19.200 verificaciones.

4.1.3. Sistema de Referencia

Para validar los resultados obtenidos con los softwares implementados en este trabajo, se ha utilizado un software de referencia llamado ALIZE [25]. ALIZE es un *toolkit* para reconocimiento de voz, de código abierto con licencia GNU/LGPL, y es desarrollado por el consorcio ELISA y la Universidad de Avignon, Francia. ALIZE implementa los métodos de verificación de locutor JFA e i-Vectors, y se considera que tiene un rendimiento *state-of-the art* dentro del medio científico, por lo que es una buena referencia para comparar con los resultados obtenidos por el software desarrollado en este trabajo.

Tabla 4.2. Número de verificaciones realizadas en un experimento

	Número de Locutores	Número de Verificaciones	Verificaciones Totales
Cliente	30	1·16 + (30 -1)·6	5.700
Impostor	40	30·16	19.200
TOTAL			24.900

4.2. Mejoras en el sistema baseline

Cuando se comparó inicialmente el sistema GMM-UBM del laboratorio con el sistema de referencia, se encontraron diferencias importantes en el valor de EER de experimentos cuando se ocupaban distorsiones como Tilt de 6dB. Al realizar un examen al sistema completo, se encontraron dos diferencias que explicaban los resultados dispares:

- En el sistema del laboratorio el UBM se calculaba utilizando el algoritmo de K-means, mientras que en el sistema de referencia se utilizaba el algoritmo EM, presentado en la revisión bibliográfica.
- En el sistema del laboratorio se realizaba la adaptación de los modelos de los locutores según [9], es decir, se adaptaban los pesos, medias y varianzas de cada *speaker* a partir del UBM. En el sistema de referencia, sólo se realiza la adaptación de medias, mientras los pesos y varianzas son simplemente copiados del UBM.

Debido a que el UBM del sistema *baseline* es utilizado en JFA e i-Vectors, fue necesario implementar estas diferencias para que los sistemas implementados en este trabajo alcanzaran rendimientos comparables a los del sistema de referencia.

Se realizaron experimentos de verificación bajo tres configuraciones: el sistema base GMM-UBM, GMM-UBM con la aplicación de CMN, y GMM-UBM con la aplicación de RASTA. Cada una de las configuraciones anteriores fue probada con las tres bases de datos para pruebas, y utilizando 40 *cohorts*. Los resultados obtenidos se muestran en la tabla 4.3.

Se observa que los resultados de ambos sistemas, LPTV y ALIZE, se encuentran dentro de los mismos rangos, y esto es lo que se quería lograr. Es importante notar que para obtener estos resultados se debieron aplicar los dos cambios mencionados (utilizar algoritmo EM y adaptar sólo las medias).

Tabla 4.3. Comparación sistema GMM-UBM

	LPTV (EER%)			ALIZE (EER%)		
Base de Datos de Test	GMM-UBM	GMM-UBM + CMN	GMM-UBM + RASTA	GMM-UBM	GMM-UBM + CMN	GMM-UBM + RASTA
Clean	1.458	1.250	1.264	1.458	1.458	1.667
Tilt 6dB	7.567	1.875	2.395	7.708	2.083	2.241
Tilt Dyn 6dB	2.500	1.875	1.875	2.292	1.628	1.667

Nota. — Los resultados fueron normalizados con 40 *cohorts*.

4.3. Joint Factor Analysis

4.3.1. Validación del Sistema

En JFA, el resultado de un experimento está determinado por los siguientes factores:

- Parámetros utilizados en la extracción de características de las elocuciones de entrenamiento y test.
- UBM utilizado.
- Inicializaciones de las matrices V, U y D.
- Iteraciones de entrenamiento de las matrices V, U y D.

Estos factores pueden ser vistos como condiciones iniciales del sistema. Luego, si se varían las condiciones iniciales, el sistema tendrá un rendimiento diferente.

Para comprobar la correcta implementación del sistema JFA, se diseñó un experimento que tuviera exactamente las mismas condiciones iniciales que el sistema de referencia. Si el sistema está correctamente diseñado, los resultados deberían ser iguales. Se definen tres condiciones iniciales para probar el sistema:

- *Escenario 1.* Las condiciones iniciales para ambos sistemas será la misma. Esto se logra guardando las matrices de inicialización para V, U y D y el UBM del sistema de referencia, y cargándolo en el sistema implementado.
- *Escenario 2.* En este caso no se utiliza la inicialización para V, U y D, pero sí el UBM del sistema de referencia.
- *Escenario 3.* No se carga ninguna matriz ni el UBM.

En todos los escenarios anteriores se utilizan los mismos parámetros para extracción de características y se realiza la misma cantidad de iteraciones para entrenar las matrices del sistema. En la tabla 4.4 se muestran los resultados de realizar un experimento, con y sin *cohorts*, para los tres escenarios descritos. Se observa que para las mismas condiciones iniciales el sistema implementado alcanza exactamente el mismo rendimiento que el sistema de referencia, lo que implica que la implementación de JFA realizada es correcta.

Tabla 4.4. Comparación rendimiento sistema JFA

	Equal Error Rate (EER)%			
Normalización de Verosimilitud	ALIZE	LPTV Escenario 1	LPTV Escenario 2	LPTV Escenario 3
Sin Normalización	6,0153	6,0153	4,1667	3,4483
40 Cohorts	3,6398	3,6398		2,0115

Nota. — Se utilizó la base de datos de test *clean*.

Tabla 4.5. Variaciones porcentuales de experimentos JFA respecto al sistema de referencia

Normalización de Verosimilitud	LPTV Escenario 1	LPTV Escenario 2	LPTV Escenario 3
Sin Normalización	0,00%	-30,73%	-42,67%
40 Cohorts	0,00%		-44,74%

Nota. — Se utilizó la base de datos de test *clean*.

Cuando se realizan los experimentos con diferentes condiciones iniciales, el sistema mejora el rendimiento. En la tabla 4.5 se observa la diferencia porcentual entre el rendimiento del software implementado y el de ALIZE, obteniéndose mejoras entre un 30% y un 45% respecto al sistema de referencia.

Los resultados obtenidos permiten concluir que el sistema fue correctamente implementado, y que alcanza un rendimiento *state-of-the-art*, al igual que el sistema de referencia.

4.3.2. Resultados

Una vez validado el sistema, se realizaron experimentos para observar el resultado de variar la cantidad de *eigenvoices* y *eigenchannels* en el rendimiento del verificador. En la tabla 4.6 se muestra el EER del sistema para 50, 100, 150 y 200 *eigenvoices*, manteniendo en 10 la cantidad de *eigenchannels*. En la tabla 4.7 se muestra el EER para la misma configuración anterior, pero con normalización de verosimilitud.

Se observa que la aplicación de *cohorts* disminuye el EER, pero no en todos los casos. Además, no se observa una tendencia clara entre el aumento de *eigenvoices* y un aumento en el rendimiento del sistema. En la tabla 4.8 se muestran experimentos con 100 *eigenvoices*, y con 10,50, 100 y 150 *eigenchannels*.

Tabla 4.6. Variación de cantidad de *eigenvoices* (e.v.) para JFA

	Equal Error Rate (EER)%			
Base de Datos de Test	50 e.v.	100 e.v.	150 e.v.	200 e.v.
Clean	3,542	3,448	3,429	3,506
Tilt 6dB	9,176	9,167	9,234	9,167
Tilt Dyn 6dB	2,917	3,125	3,125	3,084

Nota. — Se fijó la cantidad de *eigenchannels* en 10 para todos los experimentos.

Tabla 4.7. Variación de cantidad de *eigenvoices* (e.v.) para JFA utilizando 40 *cohorts*

	Equal Error Rate (EER)%			
Base de Datos de Test	50 e.v	100 e.v	150 e.v.	200 e.v.
Clean	2,031	2,011	2,069	1,954
Tilt 6dB	9,176	9,167	9,234	9,167
Tilt Dyn 6dB	2,107	2,203	2,126	2,107

Nota. — Se fijó la cantidad de *eigenchannels* en 10 para todos los experimentos.

Tabla 4.8. Variación de cantidad de *eigenchannels* (e.ch.) para JFA utilizando 40 *cohorts*

	Equal Error Rate (EER)%			
Base de Datos de Test	10 e.ch.	50 e.ch.	100 e.ch.	150 e.ch.
Clean	2,011	1,875	2,011	2,011
Tilt 6dB	9,167	8,958	9,004	9,023
Tilt Dyn 6dB	2,203	2,083	2,261	2,069

Nota. — Se fijó la cantidad de *eigenvoices* en 100 para todos los experimentos.

4.4. i-Vectors

4.4.1. Validación del Sistema

En *i-Vectors*, el resultado de un experimento está determinado por los siguientes factores:

- Parámetros utilizados en la extracción de características de las elocuciones de entrenamiento y test.
- UBM utilizado.
- Inicialización de la matriz T.
- Iteraciones de entrenamiento de la matriz T.

Para comprobar la correcta implementación del sistema *i-Vectors* implementado, se diseñó un experimento que tuviera exactamente las mismas condiciones iniciales que el sistema de referencia.

- *Escenario 1.* Las condiciones iniciales para ambos sistemas será la misma. Esto se logra guardando la matriz de inicialización para T y el UBM del sistema de referencia, y cargándolo en el sistema implementado.
- *Escenario 2.* El sistema calculará la matriz de inicialización para T internamente, y calculará el UBM con el sistema del LPTV.

En los escenarios anteriores se utilizan los mismos parámetros para extracción de características y se realiza la misma cantidad de iteraciones para entrenar la matriz de variabilidad total T. En la tabla 4.9 se muestran los resultados de realizar un experimento, con y sin *cohorts*, para los escenarios descritos. Se observa que para las mismas condiciones iniciales el sistema implementado alcanza exactamente el mismo rendimiento que el sistema de referencia, lo que implica que la implementación de JFA realizada es correcta.

Cuando se realizan los experimentos con diferentes condiciones iniciales, el sistema mejora el rendimiento, pero la mejora no es tan marcada como para JFA. En la tabla 4.10 se observa la diferencia porcentual entre el rendimiento del software implementado y el de ALIZE, obteniéndose una mejora de un 5,88 % al utilizar *cohorts*.

Los resultados obtenidos permiten concluir que el sistema fue correctamente implementado, y que alcanza un rendimiento *state-of-the-art*, al igual que el sistema de referencia.

4.4.2. Resultados

Una vez validado el sistema, se realizaron experimentos para observar el resultado de variar el tamaño de los i-vectors en el rendimiento del verificador. En la tabla 4.11 se muestra el EER del sistema para i-vectors de tamaño 50, 100, 150 y 200. En la tabla 4.12 se muestra el EER para la misma configuración anterior, pero con normalización de verosimilitud.

Tabla 4.9. Comparación rendimiento sistema i-Vectors

	Equal Error Rate (EER)%		
Normalización de Verosimilitud	ALIZE	LPTV Escenario 1	LPTV Escenario 2
Sin Normalización	7,5287	7,5287	7,5000
40 Cohorts	7,0833	7,0833	6,6666

Nota. — Se utilizó la base de datos de test *clean*.

Tabla 4.10. Variaciones porcentuales de experimentos i-Vectors respecto al sistema de referencia

Normalización de Verosimilitud	LPTV Escenario 1	LPTV Escenario 2
Sin Normalización	0,00%	-0,38%
40 Cohorts	0,00%	-5,88%

Nota. — Se utilizó la base de datos de test *clean*.

Tabla 4.11. Variación de tamaño de i-vectors

	Equal Error Rate (EER)%			
Base de Datos de Test	50	100	150	200
Clean	8,542	7,500	5,843	5,625
Tilt 6dB	21,437	16,916	14,847	13,958
Tilt Dyn 6dB	10,249	8,542	7,395	6,954

Tabla 4.12. Variación de tamaño de i-vectors utilizando 40 *cohorts*

	Equal Error Rate (EER)%			
Base de Datos de Test	50	100	150	200
Clean	7,500	6,667	5,208	5,625
Tilt 6dB	21,073	18,448	14,847	12,663
Tilt Dyn 6dB	9,981	8,103	6,839	6,458

Capítulo 5

Conclusiones

En esta memoria se estudiaron e implementaron dos sistemas de verificación de locutor texto independiente basados en análisis factorial: JFA e *i-Vectors*.

En primer lugar, se debió equiparar el rendimiento del sistema *baseline*, GMM-UBM, con el sistema de referencia que se utilizaría para validar los experimentos, dado que los sistemas JFA e *i-Vectors* dependían de parámetros del sistema base. Se debió realizar un estudio exhaustivo de las implementaciones de GMM-UBM en el software LPTV y en el software de referencia. Se realizaron dos modificaciones en el sistema LPTV:

- Se programó un algoritmo EM con un punto de partida aleatorio para realizar la estimación del UBM.
- Para adaptar los modelos de locutor inicialmente se adaptaban medias y varianzas desde el UBM con datos de entrenamiento. Esto provocaba que al realizar un experimento en ciertas condiciones de *mismatch* (por ejemplo un tilt de -6 [dB] por octava) el EER se disparara. Se realizó una adaptación más simple: sólo adaptar las medias y copiar las varianzas del UBM.

Los cambios descritos permitieron mejorar el rendimiento del sistema *baseline*, permitiendo así que tuviera sentido realizar una comparación entre el software de referencia y los sistemas implementados en esta memoria. Además, realizar estos cambios permitió que el software del LPTV fuera mejorado, lo que es útil para investigaciones futuras.

Para realizar la validación de los sistemas implementados se identificaron las condiciones iniciales del sistema. Al alimentar los verificadores con estas condiciones iniciales, se obtuvo el mismo EER que el sistema de referencia, por tanto, se concluye que ambos verificadores están bien diseñados.

Bibliografía

[1] J. Wayman, A. Jain, D. Maltoni y D. Maio, "An introduction to biometric authentication systems", en *Biometric Systems*, Springer, 2005, págs. 1-20.

[2] M. Liu, T. S. Huang y Z. Zhang, "Robust local scoring function for text-independent speaker verification", en *Proceedings of the 18th International Conference on Pattern Recognition, ICPR 2006*, Hong Kong, 2006, págs. 1146-1149.

[3] F. Bimbot, J.-F. Bonastre, C. Fredouille, G. Gravier, I. Magrin-Chagnolleau, S. Meignier, T. Merlin, J. Ortega-García, D. Petrovska-Delacrétaz y D. A. Reynolds, "A tutorial on text-independent speaker verification", *EURASIP Journal on Applied Signal Processing*, vol. 2004, págs. 430-451, 2004.

[4] J. P. Campbell Jr, "Speaker recognition", en *Biometrics*, Springer, 1996, págs. 165-189.

[5] M. J. Torres, *Robustez a efectos de canal en verificación de locutor*. Tesis Ingeniero Civil Electricista, Departamento de Ingeniería Eléctrica, Universidad de Chile, 2009.

[6] T. Kinnunen y H. Li, "An overview of text-independent speaker recognition: from features to supervectors", *Speech Communication*, vol. 52, n.º 1, págs. 12-40, 2010.

[7] R. Togneri y D. Pullella, "An overview of speaker identification: accuracy and robustness issues", *Circuits and Systems Magazine, IEEE*, vol. 11, n.º 2, págs. 23-61, 2011.

[8] S. Davis y P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences", *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 28, n.º 4, págs. 357-366, 1980.

[9] D. A. Reynolds, T. F. Quatieri y R. B. Dunn, "Speaker verification using adapted gaussian mixture models", *Digital signal processing*, vol. 10, n.º 1, págs. 19-41, 2000.

[10] J. A. Bilmes, "A gentle tutorial of the EM algorithm and its application to parameter estimation for gaussian mixture and hidden marko models", *International Computer Science Institute*, vol. 4, n.º 510, pág. 126, 1998.

[11] P. Kenny y P. Dumouchel, "Disentangling speaker and channel effects in speaker verification", en *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2004*, vol. 1, Montreal, QC, Canada, 2004, págs. I-37-40.

[12] L. Wang, N. Kitaoka y S. Nakagawa, "Robust distant speech recognition by combining position-dependent cmn with conventional cmn", en *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2007*, Honolulu, Hawaii, 2007, págs. 817-820.

[13] H. Hermansky y N. Morgan, "Rasta processing of speech", *IEEE Transactions on Speech and Audio Processing*, vol. 2, n.º 4, págs. 578-589, 1994.

[14] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel y P. Ouellet, "Front-end factor analysis for speaker verification", *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, n.º 4, págs. 788-798, 2011.

[15] P. Kenny, P. Ouellet, N. Dehak, V. Gupta y P. Dumouchel, "A study of interspeaker variability in speaker verification", *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 16, n.º 5, págs. 980-988, 2008.

[16] P Kenny, T Stafylakis, J Alam, P Ouellet y M Kockmann, "Joint factor analysis for text-dependent speaker verification", en *Odyssey 2014*, Joensuu, Finland, 2014.

[17] P. Kenny, "Joint factor analysis of speaker and session variability: theory and algorithms", *CRIM, Montreal,(Report) CRIM-06/08-13*, 2005.

[18] P. Kenny y P. Dumouchel, "Experiments in speaker verification using factor analysis likelihood ratios", en *Odyssey 2004*, Toledo, Spain, 2004, págs. 219-226.

[19] P. Kenny, G. Boulianne, P. Ouellet y P. Dumouchel, "Joint factor analysis versus eigenchannels in speaker recognition", *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, n.º 4, págs. 1435-1447, 2007.

[20] O. Glembek, L. Burget, N. Dehak, N. Brummer y P. Kenny, "Comparison of scoring methods used in speaker recognition with joint factor analysis", en *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2009*, Taipei, Taiwan, 2009, págs. 4057-4060.

[21] N. Dehak, *Discriminative and generative approaches for long-and short-term speaker characteristics modeling: application to speaker verification*. Ecole de Technologie Superieure (Canada), 2009, AAINR50490, ISBN: 978-0-494-50490-1.

[22] M. Senoussaoui, P. Kenny, N. Dehak y P. Dumouchel, "An i-vector extractor suitable for speaker recognition with both microphone and telephone speech.", en *Odyssey 2010*, Brno, Czech Republic, 2010, pág. 6.

[23] J. Campbell y A. Higgins, *Yoho speaker verification ldc94s16*, 1994.

[24] V. Poblete, F. Espic, S. King, R. M. Stern, F. Huenupan, J. Fredes y N. B. Yoma, "A perceptually-motivated low-complexity instantaneous linear channel normalization technique applied to speaker verification", *Computer Speech and Language*, vol. 31, n.º 1, págs. 1 -27, 2015.

[25] J.-F. Bonastre, F. Wils y S. Meignier, "Alize, a free toolkit for speaker recognition", en *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2005*, Philadelphia, Pennsylvania, 2005, págs. 737-740.

Capítulo 6

Anexos

6.1. Algoritmos Implementados

En esta sección se muestran los algoritmos que están implementados en el software desarrollado en este trabajo.

6.1.1. Algoritmos para JFA

En el algoritmo 1 se muestra el proceso de entrenamiento de la matriz V.

Algoritmo 1 Entrenamiento de matriz V para JFA

Entrada: UBM, rango de V, número iteraciones de entrenamiento, datos entrenamiento para V

Salida: Matriz V

Inicializar matriz V
for Impostor p en datos de entrenamiento **do**
  Calcular $N(p)$ según ecuación (3.10)
  Calcular $\tilde{F}(p)$ según ecuación (3.11)
end for
for Número de iteraciones de entrenamiento **do**
  for Impostor p en datos de entrenamiento **do**
    Calcular $l_V(p)$ según ecuación (3.13)
    Calcular $\tilde{y}(p)$ según ecuación (3.14)
  end for
  for Gaussiana i en número de gaussianas del UBM **do**
    Estimar $\mathfrak{A}_i$ según ecuación (3.15)
  end for
  Estimar $\mathfrak{C}$ según ecuación (3.16)
  Actualizar V según ecuación (3.17)
end for
return Matriz V

En el algoritmo 2 se muestra el proceso de entrenamiento de la matriz U.
En el algoritmo 3 se muestra el proceso de entrenamiento de la matriz D.

Algoritmo 2 Entrenamiento de matriz U para JFA

Entrada: UBM, rango de U, matriz V entrenada, número iteraciones de entrenamiento, datos entrenamiento para U

Salida: Matriz U

for Impostor p en datos de entrenamiento **do**
  Calcular $\tilde{y}(p)$ según ecuación (3.14)
end for
Inicializar matriz U
for Impostor p en datos de entrenamiento **do**
  for Elocución r perteneciente a p **do**
    Calcular $N(p,r)$ según ecuación (3.19)
    Calcular $\tilde{F}(p,r)$ según ecuación (3.21)
  end for
end for
for Número de iteraciones de entrenamiento **do**
  for Impostor p en datos de entrenamiento **do**
    for Elocución r perteneciente a p **do**
      Calcular $l_U(p,r)$ según ecuación (3.22)
      Calcular $\tilde{x}(p,r)$ según ecuación (3.23)
    end for
  end for
  for Gaussiana i en número de gaussianas del UBM **do**
    Estimar $\mathfrak{A}_i$ según ecuación (3.24)
  end for
  Estimar $\mathfrak{C}$ según ecuación (3.25)
  Actualizar U según ecuación (3.26)
end for
return Matriz U

Algoritmo 3 Entrenamiento de matriz D para JFA

Entrada: UBM, matrices V y U entrenadas, número iteraciones de entrenamiento, datos entrenamiento para D

Salida: Matriz D

for Impostor p en datos de entrenamiento **do**
  Calcular $\tilde{y}(p)$ según ecuación (3.14)
  for Elocución r perteneciente a p **do**
    Calcular $\tilde{x}(p,r)$ según ecuación (3.23)
  end for
end for
Inicializar matriz D
for Impostor p en datos de entrenamiento **do**
  for Elocución r perteneciente a p **do**
    Calcular $\tilde{F}(p,r)$ según ecuación (3.27)
  end for
end for
for Número de iteraciones de entrenamiento **do**
  for Impostor p en datos de entrenamiento **do**
    Calcular $l_D(p)$ según ecuación (3.28)
    Calcular $\tilde{z}(p)$ según ecuación (3.29)
  end for
  Estimar $\mathfrak{a}$ según ecuación (3.30)
  Estimar $\mathfrak{b}$ según ecuación (3.31)
  Actualizar D según ecuación (3.32)
end for
return Matriz D

En el algoritmo 4 se muestra el proceso de entrenamiento de la matriz de variabilidad total, T.

Algoritmo 4 Realización de un experimento completo de JFA

Cargar UBM entrenado previamente
Entrenar matriz V usando datos de entrenamiento para V.
Entrenar matriz U usando V y datos de entrenamiento para U.
Entrenar matriz D usando V, U y datos de entrenamiento para D.
for Locutor s a enrolar en el sistema **do**
    Estimar modelo de s usando V,U,D y datos de entrenamiento para s
    Guardar modelo de s
end for
for Test definido por una elocución r_o un locutor supuesto s_o **do**
    Cargar modelo de s_o
    Calcular $LLR_{lineal}(r_o|s_o)$ según ecuación 3.41.
    Guardar LLR resultante
end for
Calcular *equal error rate* en base a lista de $LLRs$ obtenidos
return

6.1.2. Algoritmos para i-Vectors

En el algoritmo 5 se muestra la forma en que se implementó un experimento completo de i-Vectors.

Algoritmo 5 Entrenamiento de matriz de variabilidad total, T

Cargar UBM entrenado previamente
Entrenar matriz T usando datos de entrenamiento para T.
for Locutor s a enrolar en el sistema **do**
    Calcular i-vector para s, usando T y datos de entrenamiento para s
    Guardar i-vector de s
end for
for Test definido por una elocución r_{test} un locutor objetivo s_{obj} **do**
    Calcular i-vector de r_{test}: w_{test}
    Calcular i-vector de s_{obj}: w_{obj}
    Calcular y guardar *Cosine Distance Scoring* entre w_{test} y w_{obj}, según ecuación (3.53).
end for
Calcular *equal error rate* en base a lista de *Cosine Distance Scorings* obtenidos
return

En el algoritmo 6 se muestra la forma en que se implementó un experimento completo de i-Vectors.

Algoritmo 6 Realización de un experimento completo de i-Vectors

Cargar UBM entrenado previamente
Entrenar matriz T usando datos de entrenamiento para T.
for Locutor s a enrolar en el sistema **do**
    Calcular i-vector para s, usando T y datos de entrenamiento para s
    Guardar i-vector de s
end for
for Test definido por una elocución r_{test} un locutor objetivo s_{obj} **do**
    Calcular i-vector de r_{test}: w_{test}
    Calcular i-vector de s_{obj}: w_{obj}
    Calcular y guardar *Cosine Distance Scoring* entre w_{test} y w_{obj}, según ecuación (3.53).
end for
Calcular *equal error rate* en base a lista de *Cosine Distance Scorings* obtenidos
return