4 TÉCNICAS DE VERIFICACIÓN AUTOMÁTICA DE LOCUTOR

Todos los pasos explicados detalladamente en el capítulo anterior correspondientes a la obtención de coeficientes cepstrales, SAD y posterior normalización de parámetros son etapas comunes en cualquier sistema de Verificación Automática de Locutor, y en general, en cualquier técnica que trabaje con reconocimiento de voz. Una vez obtenida la extracción de características hay que proceder a la creación del modelo estadístico del locutor necesario en la fase de entrenamiento y al scoring durante el test. En este capítulo se procederá a describir las técnicas más importantes en Verificación Aautomática de Locutor que se han ido utilizando en los últimos diez años, analizándolas con total detalle desde un punto de vista completamente estadístico. En capítulos posteriores se simularán dichas técnicas, lo cual nos permitirá comparar los resultados obtenidos entre ellas y comprobar las mejoras que se han ido produciendo tras el transcurso de los años.

4.1 Modelos generativos frente a discriminativos

Dado un segmento de entrenamiento $\boldsymbol{X} = \{\vec{x}_t\}_{t=1,\dots,N_X}$ formado por N_X vectores cepstrales $\vec{x}_t$ (es decir coeficientes MFCC correspondientes a la trama t juntos con los términos de velocidad y aceleración y que en el capítulo anterior representamos mediante $\vec{c}_t$, pero que a partir de ahora diferenciaremos entre $\vec{x}$ si es para entrenamiento e $\vec{y}$ si es para evaluación), un modelo generativo es un modelo puramente estadístico que intenta representar lo más fielmente posible la distribución de probabilidad conjunta entre los vectores de características y el locutor del que provienen $p(\boldsymbol{X}|Speaker)$. Puesto que la función de probabilidad conjunta entre los datos de entrenamiento y la clase a la que pertenecen es conocida, es posible "generar" nuevos datos que se ajusten a la distribución de probabilidad de los datos de entrenamientos, y de ahí que se le denomine generativo.

Por otra parte, los modelos discriminativos intentan predecir, que no modelar, la clase o el locutor al que pertenecen los datos de entrenamiento modelando la función de distribución condicional $p(Speaker|\boldsymbol{X})$ a través de las fronteras de separación de las clases (locutores). Para ello, no se requiere conocer la verdadera forma funcional de la distribución de la probabilidad conjunta entre el locutor y los vectores de entrenamiento, por lo que tampoco es posible generar nuevas muestras de los vectores de características. Por lo tanto en este caso estamos interesados en modelar la frontera de separación entre los locutores, mientras que en el modelo generativo se intenta modelar al locutor.

De las cuatro técnicas que se incluirán en este proyecto, la única que se puede considerar totalmente como generativa es la de Modelos de Mezclas Gaussianas (GMM,

Gaussian Mixture Models) mientras que SVM (Support Vector Machines), NAP (Nuissance Atribution Proyection) y FA (Factor Analisis) las podemos encuadrar dentro de las técnicas discriminativas.

4.2 Modelos de Mezclas Gaussianas (GMM, Gaussian Mixture Models)

Tal y como se explicó en la sección 2.2, dado un segmento de entrenamiento que presentamos mediante $\boldsymbol{X} = \{\vec{x}_t\}_{t=1,\dots,N_X}$ y uno de evaluación que identificamos como $\boldsymbol{Y} = \{\vec{y}_t\}_{t=1,\dots,N_Y}$, en cualquier sistema de verificación existen dos posibles hipótesis:

- H_0: los segmentos $\boldsymbol{X}$ e $\boldsymbol{Y}$ fueron hablados por la misma persona.
- H_1: los segmentos $\boldsymbol{X}$ e $\boldsymbol{Y}$ no pertenecen a la misma persona.

Si X denota al modelo predicho en la fase de entrenamiento con el segmento $\boldsymbol{X}$, durante la fase de test, dado $\boldsymbol{Y}$, nos gustaría saber cuál de las dos hipótesis es más probable. Es decir, desde un punto de vista estadístico, queremos comparar $p(X|\boldsymbol{Y})$ y $p(\bar{X}|\boldsymbol{Y})$ (donde $\bar{X}$ se conoce como el modelo alternativo a X). Por lo tanto, la hipótesis H_0 será aceptada si:

$$p(X|\boldsymbol{Y}) > p(\bar{X}|\boldsymbol{Y}). \tag{4.1}$$

Mediante el teorema de Bayes esto es equivalente a afirmar que:

$$\frac{p(\boldsymbol{Y}|X)p(X)}{p(\boldsymbol{Y})} > \frac{p(\boldsymbol{Y}|\bar{X})p(\bar{X})}{p(\boldsymbol{Y})}, \tag{4.2}$$

$$p(\boldsymbol{Y}|X)p(X) > p(\boldsymbol{Y}|\bar{X})p(\bar{X}), \tag{4.3}$$

$$\frac{p(\boldsymbol{Y}|X)}{p(\boldsymbol{Y}|\bar{X})} > \frac{p(\bar{X})}{p(X)}, \tag{4.4}$$

$$\frac{p(\boldsymbol{Y}|X)}{p(\boldsymbol{Y}|\bar{X})} > \theta. \tag{4.5}$$

$p(X)$ representa la probabilidad a priori de tener un juicio en el que los datos de entrenamiento y evaluación son pronunciados por el mismo locutor ("*true trial*"), de tal forma que el cociente entre $p(X)$ y $p(\bar{X})$ se puede considerar como el umbral θ de nuestro sistema. Sin embargo, tal y como se adelantó en el capítulo 2, en función de la aplicación en la que trabajemos, nos interesará un umbral más bajo o más alto, con lo cual ese consciente es un parámetro perfectamente ajustable a los intereses de la aplicación. Así que estamos interesados en calcular dos modelos estadísticos, X y $\bar{X}$, que representen al usuario y al modelo alternativo, respectivamente, de tal forma que se

pueda enfrentar el fichero de evaluación a ambos modelos para obtener una puntuación que nos permita seleccionar una de las dos hipótesis posibles.

En GMM [D.A. Reynolds, 1992], dado un modelo estadístico λ de un locutor o conjunto de locutores (que se puede corresponder con X o $\bar{X}$), la probabilidad de que un vector cepstral de evaluación $\vec{y}_t$ pertenezca a dicho modelo es representado mediante una combinación lineal de distribuciones de probabilidad gaussianas D-dimensionales (D es la dimensión del vector de características $\vec{y}_t$)

$$p(\vec{y}_t|\lambda) = \sum_{i=1}^{M} w_i p_i(\vec{y}_t), \tag{4.6}$$

donde M representa el número de componentes gaussianas, w_i sus correspondientes pesos sujetos a la restricción de que $\sum_{i=1}^{M} w_i = 1$ y $p_i(\vec{y}_t)$ se expresa de la siguiente forma:

$$p_i(\vec{y}_t) = \frac{1}{(2\pi)^{D/2}|\Sigma_i|^{1/2}} e^{-\left(\frac{1}{2}\right)(\vec{y}_t-\vec{\mu}_i)'\Sigma_i^{-1}(\vec{y}_t-\vec{\mu}_i)} \qquad i = 1, \dots, M, \tag{4.7}$$

donde no podemos olvidar la restricción adicional de $\int_{-\infty}^{\infty} p_i(\vec{y})\, d\vec{y} = 1$.

Es decir, cada componente gaussiana del modelo de locutor o del modelo alternativo tendrá su correspondiente vector de medias y matriz de covarianza, de tal forma que al modelo lo identificaremos mediante la notación $\lambda = \{w_i, \vec{\mu}_i, \Sigma_i\}, i = 1, \dots, M$. Comentar también que el software empleado en este proyecto trabaja con una matriz de covarianzas diagonal, debido a que se ha demostrado que experimentalmente produce mejores resultados y por supuesto el coste computacional es muchísimo menor.

Para entender con más claridad el concepto de representación de modelos mediante componentes gaussianas, se puede observar el ejemplo de la siguiente figura. En la parte de la izquierda se encuentra el histograma correspondiente a la primera componente de los vectores de características D-dimensionales de un fichero de entrenamiento de un locutor. A continuación se puede verificar cómo con 4 o 32 componentes gaussianas y con sus respectivos pesos reproducimos dicha distribución, es decir, generamos la distribución estadística de la primera componente cepstral de dicho locutor. Este análisis se ha centrado en una variable mono-dimensional para facilitar el entendimiento. Se podría haber realizado con cualquiera de las D componentes.

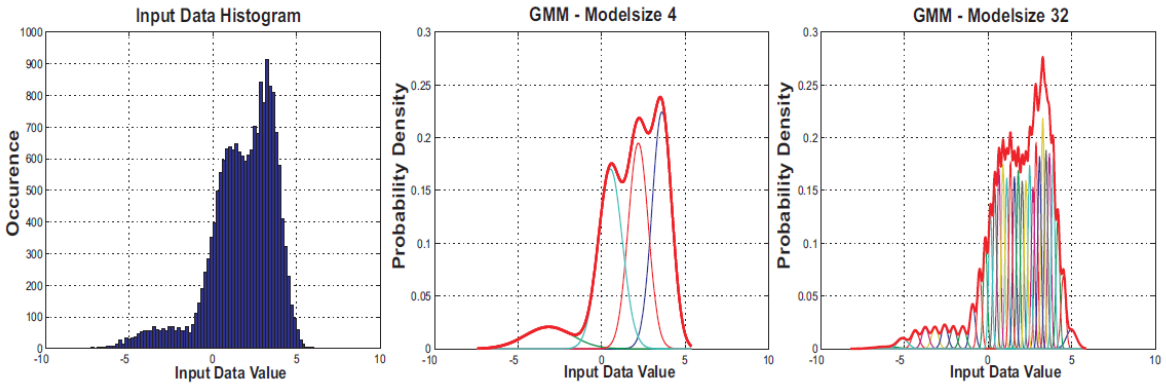

Figura 15. *Modelo de mezcla gaussiana mono-dimensional para una función de distribución de entrada. Histograma (a la izquierda) junto con dos aproximaciones GMM de diferente tamaño (4 y 32, respectivamente) [Robin Harald Priewald, 2009]*

A continuación se ha ilustrado el mismo ejemplo pero con las dos primeras componentes de los vectores de características. Es decir, en este caso se está trabajando con una variable aleatoria bidimensional gaussiana, donde además sus dos componentes son (teóricamente) conjuntamente gaussianas:

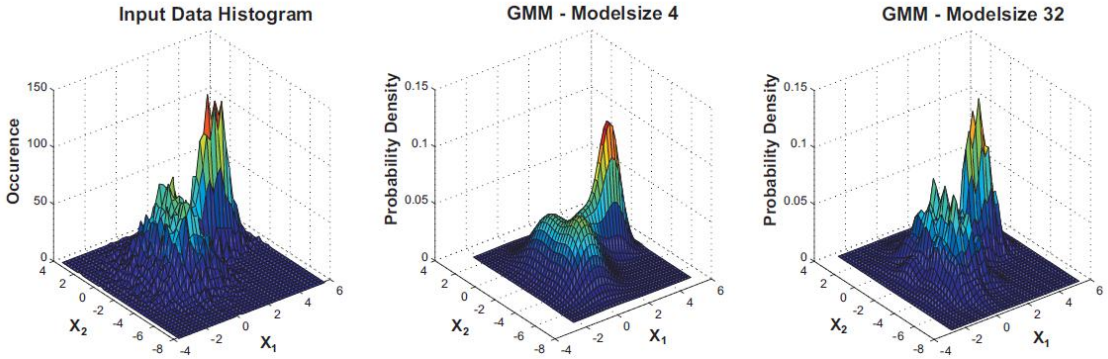

Figura 16. *Modelo de mezclas gaussianas bidimensional para una función de distribución de entrada. Histograma (a la izquierda) junto con dos aproximaciones GMM de diferente tamaño (centro y derecha) [Robin Harald Priewald, 2009]*

Hasta aquí, hemos visto que con un modelo de mezclas gaussianas podemos calcular la probabilidad de que un vector de test pertenezca a dicho modelo. Pero la cuestión importante es: ¿cómo calculamos los pesos, medias y matrices de varianza del modelo del locutor X y del alternativo $\bar{X}$? En este proyecto se ha trabajado con la alternativa conocido como GMM-UBM (Gaussian Mixture Models – Universal Background Model).

4.2.1 GMM-UBM

Fue introducida por Reynolds [D.A. Reynolds, 1995] en el año 1995 y se puede considerar como la primera gran técnica contemporánea dentro del campo de la verificación de interlocutor, teniendo una transcendencia fundamental en gran parte de las aplicaciones actuales. En este proyecto juega un papel fundamental debido a que el resto de técnicas que se analizarán e implementarán tienen en cuenta muchos de los principios de GMM-UBM y además parten de esta.

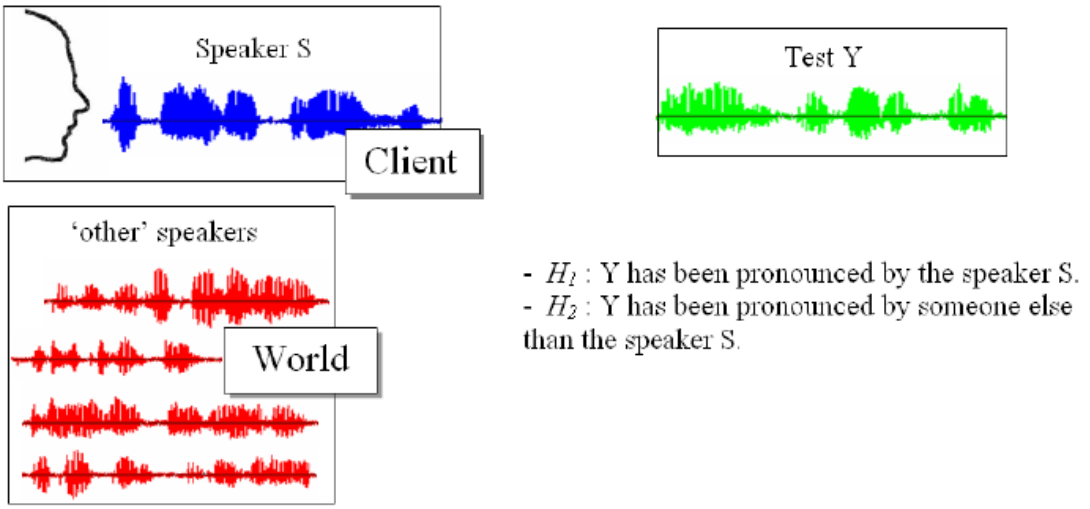

Figura 17. *Ilustración sobre los tres tipos de segmentos o grabaciones necesarias en GMM-UBM.*

La propuesta de Reynolds es que el modelo alternativo que hemos identificado como $\bar{X}$ es generado a través de un conjunto de segmentos pertenecientes a muchos locutores de tal forma que seamos capaces de generar un modelo GMM universal (UBM) que represente a la población mundial. Interesaría que los segmentos de audio provengan de personas con un alto grado de variabilidad acústica entre ellas, de tal forma que el modelo represente de manera fiel las características comunes de los locutores que pueden participar en nuestro sistema de verificación. Un detalle importante respecto a esto es que las grabaciones para crear dicho modelo universal no tienen que pertenecer a locutores que posteriormente vayamos a entrenar o testear, sino simplemente locuciones de muchas personas donde se intente captar de manera general las características acústicas de la población mundial (de cualquier posible locutor). A continuación, dado un fichero de entrenamiento de un participante, mediante un tipo de entrenamiento que explicaremos detalladamente en la sección siguiente, se adaptan ciertas componentes gaussianas del UBM, generando su correspondiente modelo X. La estructura del proceso se refleja perfectamente en la siguiente figura, en donde vemos que una vez obtenido el modelo universal representado en rojo a través de datos

pertenecientes a muchísimos locutores, para obtener el modelo GMM de una persona únicamente hay que modificar parámetros de las mezclas gaussianas del UBM.

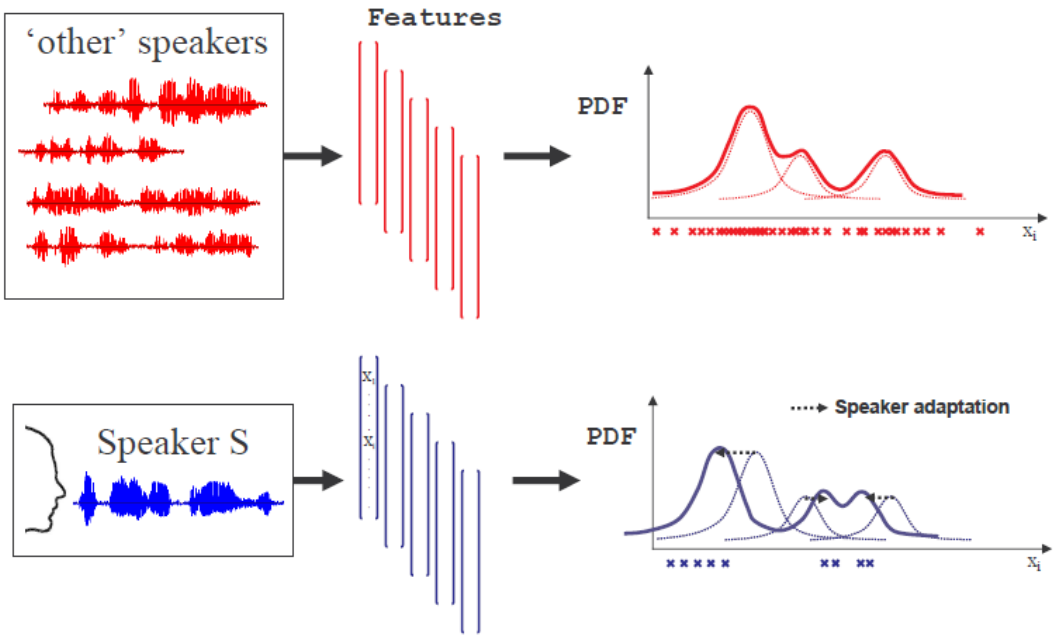

Figura 18. *En rojo, generación del modelo universal en GMM, y en azul, la fase de entrenamiento para un locutor específico mediante adaptación de ciertos parámetros del UBM. Figura adaptada de [B. Fauve, 2009].*

4.2.1.1 Modelo Universal

Partiendo de que se disponen de T vectores cepstrales D-dimensionales pertenecientes a muchísimos locutores (con un alto grado de variabilidad acústica entre ellos) y que representamos mediante el conjunto de vectores $\boldsymbol{Z} = \{\vec{z}_t\}_{t=1,\dots,T}$ (se ha representado con $\vec{z}_t$ en lugar de $\vec{x}_t$ e $\vec{y}_t$ para diferenciarlo del tradicional fichero de entrenamiento o de test de un usuario determinado), tenemos que diseñar un algoritmo de entrenamiento que nos permita representar de manera fiel a la población mundial. Naturalmente, la fidelidad y precisión del UBM dependerá también de la variabilidad y calidad de los datos que estamos utilizando para generarlo. Este modelo universal o alternativo $\bar{X} = \lambda_{UBM} = \{w_i, \vec{\mu}_i, \Sigma_i\}, i = 1, \dots, M$ se caracterizará por maximizar la función de probabilidad condicionada $p(\lambda|\boldsymbol{Z})$:

$$\lambda_{UBM} = \underset{\lambda}{argmax}\, p(\boldsymbol{Z}|\lambda) = \underset{\lambda}{argmax}\, p(\lambda|\boldsymbol{Z}). \qquad (4.8)$$

El algoritmo del que se ha hecho uso en este proyecto recibe el nombre de EM (Expectation Maximization) [A. Dempster, N. Laird, and D. Rubin, 1977], con un procedimiento muy similar al algortimo Baum-Welch utilizado en los modelos ocultos de Markov (HMM). Se trata de un entrenamiento iterativo en el que en cada iteración se incrementa el valor de $p(\boldsymbol{Z}|\lambda)$:

$$p(\boldsymbol{Z}|\lambda^{k+1}) > p(\boldsymbol{Z}|\lambda^{k}), \tag{4.9}$$

donde λ^k representa el modelo obtenido después de la iteración k-ésima.

Para ello, este algoritmo hace uso de las siguientes ecuaciones:

$$\gamma_i(t) = \frac{w_i^k p_i(\vec{z}_t)}{\sum_{j=1}^{M} w_j^k p_j(\vec{z}_t)}, \tag{4.10}$$

$$w_i^{k+1} = \frac{1}{T}\sum_{t=1}^{T} \gamma_i(t), \tag{4.11}$$

$$\vec{\mu}_i{}^{k+1} = \frac{\sum_{t=1}^{T} \gamma_i(t)\vec{z}_t}{\sum_{t=1}^{T} \gamma_i(t)}, \tag{4.12}$$

$$\Sigma_i{}^{k+1} = \frac{\sum_{t=1}^{T} \gamma_i(t)\vec{z}_t^2}{\sum_{t=1}^{T} \gamma_i(t)} - \vec{\mu}_t^{2\,k+1}, \tag{4.13}$$

$$i = 1, \dots, M, \tag{4.14}$$

en donde la notación $\vec{z}_t^2$ representa la diagonalización de la matriz $\vec{z}_t\vec{z}_t{}'$, ya que como se añadió anteriormente se está trabajando con matrices de covarianza diagonales. M es el número de componentes o mezclas gaussianas. Tal y como se puede constatar en las ecuaciones, mediante los parámetros obtenidos en la iteración k-ésima, calculamos $\gamma_i(t)$ que representa la probabilidad de ocupación de la mezcla i-ésima en la trama $\vec{z}_t$ y que nos permitirá calculas los nuevos parámetros gaussianos para la siguiente iteración. Generalmente son necesarios entre 5 y 10 iteraciones.

Una cuestión importante relativa a este apartado es cómo se inicializan los parámetros en la iteración 0. Es un detalle que no presenta mucha relevancia puesto que se creé [D.A. Reynolds, T.F. Quatieri and R.Dunn, 2000] que no afecta al modelo final generado por el algoritmo. De hecho, el software con el que se ha trabajado en este proyecto distribuye equitativamente los pesos para todas las mezclas gaussianas. Para los vectores de media escoge un vector aleatorio perteneciente a una trama de un fichero de entrenamiento de un locutor y la matriz de covarianza la inicializa a la matriz identidad.

4.2.1.2 *Adaptación MAP ("Maximum a posteriori")*

En este apartado se procederá a describir como a partir de un fichero de entrenamiento de un locutor $\boldsymbol{X} = \{\vec{x}_t\}_{t=1,\dots,N_x}$ es posible generar su modelo GMM correspondiente mediante adaptación MAP [Gauvain and Lee, 1994] de los parámetros del modelo UBM. Lógicamente, para crear el modelo estadístico de cualquier usuario, es imprescindible que previamente hayamos generado el UBM. Añadir también que mientras que para construir el UBM generalmente se dispone de una cantidad ilimitada de datos ya que cuanto más grande sea más exacto y preciso será el modelo universal generado, para construir el modelo de un usuario a partir de su segmento de entrenamiento dispondremos de cantidad de información limitada (generalmente 2.5 minutos de conversación del locutor).

En la adaptación MAP el UBM es utilizado como un modelo a priori, y los nuevos parámetros GMM son adaptados de éste de tal manera que el nuevo modelo se ajuste más a los datos de entrenamiento $\boldsymbol{X}$**.** Las ecuaciones, muy similares a las que se utilizaban en el algoritmo EM son las siguientes:

$$\gamma_i(t) = \frac{w_i p_i(\vec{x}_t)}{\sum_{j=1}^{M} w_j p_j(\vec{x}_t)}, \tag{4.15}$$

donde $\gamma_i(t)$ se obtiene exactamente con la misma expresión que en EM y w_i o p_i son los parámetros del UBM para la mezcla i-ésima. Luego necesitamos calcular los estadísticos de orden cero (para el peso), primer (para la media) y segundo orden (para la covarianza) de cada mezcla o componente gaussiana:

$$n_i = \sum_{t=1}^{N_x} \gamma_i(t), \tag{4.16}$$

$$E_i(\vec{x}) = \frac{1}{n_i} \sum_{t=1}^{N_x} \gamma_i(t) \vec{x}_t, \tag{4.17}$$

$$E_i(\vec{x}^2) = \frac{1}{n_i} \sum_{t=1}^{N_x} \gamma_i(t) \vec{x}_t^2, \tag{4.18}$$

con $i = 1, \dots, M$.

Con los nuevos estadísticos calculados a partir del segmento de entrenamiento, procedemos a actualizar cada una de las mezclas gaussianas del UBM mediante adaptación MAP:

$$\widehat{w}_i = \left[\frac{\alpha_i n_i}{N_x} + (1 - \alpha_i) w_i\right] \eta, \tag{4.19}$$

$$\widehat{\vec{\mu}}_i = \alpha_i E_i(\vec{x}) + (1 - \alpha_i) \vec{\mu}_i, \tag{4.20}$$

$$\hat{\Sigma}_i = \alpha_i E_i(\vec{x}^2) + (1 - \alpha_i)(\Sigma_i + \vec{\mu}_i^2) - \hat{\vec{\mu}}_i^2, \qquad (4.21)$$

donde $\alpha_i = \frac{n_i}{n_i + r}$ es el coeficiente de adaptación, y r es el "factor de relevancia" que debe de ser ajustado en función de la cantidad de audio disponible para adaptar el UBM. En los experimentos r toma el valor de 14. En las ecuaciones anteriores, $\{w_i, \vec{\mu}_i, \Sigma_i\}$ y $\{\hat{w}_i, \hat{\vec{\mu}}_i, \hat{\Sigma}_i\}$ representan los parámetros de la i-ésima mezcla gaussiana para el UBM y para el locutor que está siendo entrenado mediante MAP, respectivamente. Además, tenemos que tener en cuenta que los pesos $\hat{w}_i$ de un locutor deben de estar normalizados, de ahí que se haga necesario el uso del factor η. Una ilustración muy útil y clara del entrenamiento MAP puede contemplarse en la siguiente figura:

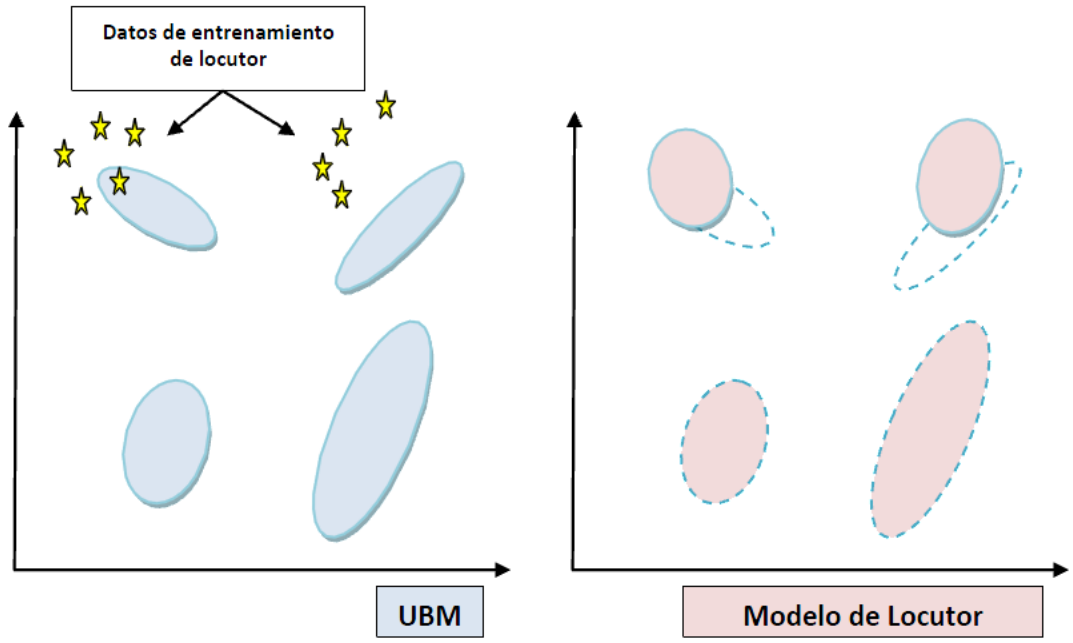

Figura 19. *Ejemplo de adaptación de un modelo de locutor. Adaptada de [D.A. Reynolds, T.F. Quatieri and R.Dunn, 2000]*

En la parte izquierda vemos representado el UBM (en color azul) suponiendo que estamos trabajando solamente con 4 mezclas gaussianas y asumiendo que los vectores cepstrales tienen únicamente 2 dimensiones (esto nunca sucede ya que el número de dimensiones siempre es muchísimo mayor). También observamos los vectores cepstrales de un segmento de entrenamiento de un locutor del que estamos interesados en calcular su modelo mediante adaptación MAP del UBM. Pues bien, aquellas mezclas gaussianas del UBM que guarden mayor similitud con los vectores de entrenamiento, es decir, aquellas cuyos estadísticos n_i sean relativamente altos (en el caso de la figura anterior, las dos mezclas gaussianas superiores) sufrirán cambios grandes, es decir, sus parámetros se adaptarán a los datos del locutor en cuestión. Por otra parte, aquellas mezclas gaussianas con $n_i \approx 0$ no sufrirán cambio alguno con respecto al UBM. En la parte derecha de la figura podemos ver el modelo del locutor,

donde las dos componentes superiores han sufrido cambios con respecto a UBM, mientras que las dos inferiores permanecen inalteradas.

Por último, añadir que en este proyecto, al igual que en casi todas las aplicaciones actuales, el software adapta únicamente las medias, porque se considera que la adaptación de las matrices de covarianza y de los pesos no influye notablemente en el rendimiento final del sistema, teniendo el inconveniente fundamental que ralentiza el proceso de entrenamiento (mayor coste computacional). Por lo tanto, el modelo de un usuario y del UBM únicamente pueden discrepar en las medias de las mezclas gaussianas.

4.2.1.3 Puntuación en GMM-UBM

Dado un segmento de audio de entrenamiento $\boldsymbol{X} = \{\vec{x}_t\}_{t=1,\dots,N_X}$ del que se genera su modelo estadístico X mediante adaptación MAP del UBM, y dado uno de evaluación $\boldsymbol{Y} = \{\vec{y}_t\}_{t=1,\dots,N_Y}$, ya tenemos toda la información necesaria para poder calcular el scoring, es decir, para enfrentar el segmento $\boldsymbol{Y}$ al modelo X y $\bar{X}$ y comprobar si la puntuación es superior o inferior al umbral. Siguiendo (4.5) y asumiendo que todas las tramas $\{\vec{y}_t\}$ son estadísticamente independientes entre ellas, tenemos que:

$$\prod_{t=1}^{N_Y} \frac{p(\vec{y}_t|X)}{p(\vec{y}_t|\bar{X})} > \theta^{N_Y.} \tag{4.22}$$

Aplicando el logaritmo a la ecuación anterior, se llega a la siguiente expresión:

$$\sum_{t=1}^{N_Y} [\log\left(p(\vec{y}_t|X)\right) - \log\left(p(\vec{y}_t|\bar{X})\right)] > N_{Y.} \log(\theta), \tag{4.23}$$

de tal forma que la puntuación final entre los segmentos de entrenamiento $\boldsymbol{X}$ y de test $\boldsymbol{Y}$ queda representada de la siguiente forma:

$$Score(\boldsymbol{X}, \boldsymbol{Y}) = \frac{1}{N_Y} \sum_{t=1}^{N_Y} [\log\left(p(\vec{y}_t|X)\right) - \log\left(p(\vec{y}_t|\bar{X})\right)]. \tag{4.24}$$

El parámetro $N_{Y.}$ representa la normalización con respecto al número de tramas del fichero de test. Si queremos relacionar la puntuación con las mezclas gaussianas característicos de la estructura GMM-UBM:

$$Score(\boldsymbol{X}, \boldsymbol{Y}) = \frac{1}{N_Y} \sum_{t=1}^{N_Y} \left[\log\left(\sum_{i=1}^{M} w_i \hat{p}_i(\vec{y}_t)\right) - \log\left(\sum_{i=1}^{M} w_i p_i(\vec{y}_t)\right)\right], \tag{4.25}$$

en donde las dos probabilidades condicionadas quedan representadas por:

$$p_i(\vec{y}_t) = \frac{1}{(2\pi)^{D/2}|\Sigma_i|^{1/2}} e^{-\left(\frac{1}{2}\right)(\vec{y}_t-\vec{\mu}_i)'\Sigma_i^{-1}(\vec{y}_t-\vec{\mu}_i)}, \quad (4.26)$$

$$\hat{p}_i(\vec{y}_t) = \frac{1}{(2\pi)^{D/2}|\Sigma_i|^{1/2}} e^{-\left(\frac{1}{2}\right)(\vec{y}_t-\hat{\vec{\mu}}_i)'\Sigma_i^{-1}(\vec{y}_t-\hat{\vec{\mu}}_i)}. \quad (4.27)$$

Se observa el hecho de que para los dos modelos (X y $\bar{X}$) los pesos y la matriz de covarianza de las mezclas gaussianas son exactamente los mismos, ya que se afirmó que únicamente los vectores de medias ($\vec{\mu}_i$ y $\hat{\vec{\mu}}_i$) pueden ser diferentes.

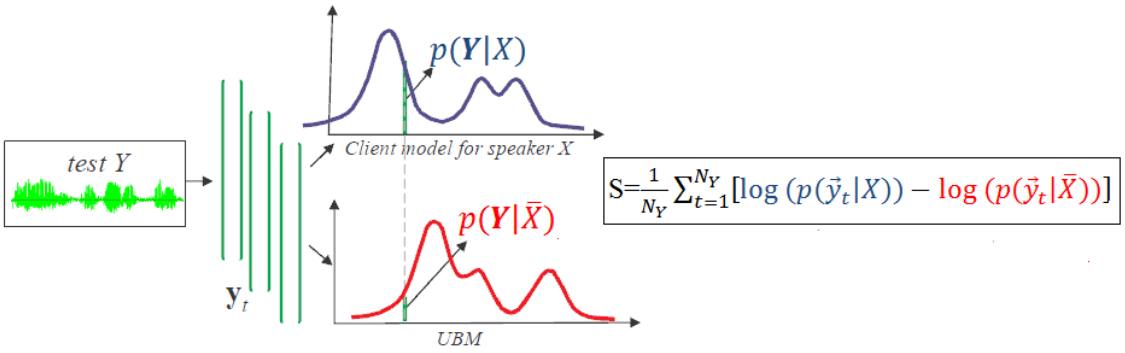

Figura 20. *Ilustración esquemática de la fase de test en la técnica de GMM-UBM. Figura adaptada de [B. Fauve, 2009].*

En páginas anteriores se ha expuesto todo el razonamiento matemático que conlleva el procedimiento de entrenamiento y puntuación en GMM-UBM. Sin embargo, se podría intentar abordar desde un punto de vista completamente teórico sin aludir a formulas y expresiones matemáticas. Cuando se entrena a un locutor, generalmente sus vectores cepstrales se caracterizarán en media por asemejarse a determinadas mezclas gaussianas (aquellas que producen una probabilidad $p_i(\vec{x}_t)$ alta) y por diferir fuertemente con medias de otras componentes gaussianas. Pues tal y como se describió en 4.2.1.2, las componentes más pesadas, es decir, aquellas que presentan una mayor probabilidad, sufrirán cambios (serán adaptadas) mientras que las otras mezclas permanecerán inalteradas. En la fase de evaluación, si la persona es quién dice ser, en líneas generales sus mezclas más pesadas deberían de coincidir con las que se detectaron en su entrenamiento (puesto que se trata de la misma persona), pero como éstas fueron modificadas con respecto al UBM, cuando enfrentemos el fichero de test a los dos modelos la diferencia entre los dos scores será relativamente grande y en teoría aceptaremos al usuario. Hablamos de diferencia puesto que en la ecuación (4.23) se observa que al aplicar el logaritmo la relación de puntuaciones (con respecto al locutor y al UBM) se transforma en una resta. Sin embargo, si el locutor en cuestión no es quién dice ser, sus mezclas más pesadas no coincidirán con las que se adaptaron en el entrenamiento y para ellas no habrá diferencia entre el modelo del locutor y el UBM, obteniendo una puntuación final muy baja.

Sin embargo, si todo resulta aparentemente tan simple, ¿por qué podemos equivocarnos en la decisión final? Hay que tener en cuenta que diferentes locutores pueden tener como mezclas más pesadas las mismas (o que en un alto porcentaje coincidan) y en ese caso el sistema podría obtener una puntuación mayor que el umbral para diferentes interlocutores. De aquí se deduce que incrementando el número de componentes gaussianas se podría reducir la probabilidad de error, puesto que es más difícil que las características acústicas de dos personas diferentes coincidan en las mismas mezclas si el abanico de posibilidades es muy amplio. De cualquier manera, este hecho se analizará con total detalle en el apartado de las simulaciones.

4.3 Máquina de Vectores Soporte (SVM, Support Vector Machines)

SVM es la segunda técnica que vamos a implementar y analizar con detalle en este proyecto después de GMM-UBM. Se encuadra dentro de las denominadas técnicas discriminativas. De manera general, las máquinas de vectores soporte son básicamente un algoritmo de clasificación de patrones binarios, cuyo objetivo es asignar cada patrón a una clase. Por ejemplo, si tenemos dos conjuntos de elementos, uno de ellos compuesto por ovejas blancas y otro por ovejas negras, el algoritmo tratará de diferenciar estas ovejas en función de su color (clase). Antes de adentrarse en esta técnica dentro del campo de la verificación de locutor, sería muy conveniente hacer una pequeña introducción sobre el concepto de máquinas de vectores.

4.3.1 Aspectos generales de las máquinas de vectores

Dado un conjunto de T vectores de entrenamiento perteneciente a dos clases diferentes (clase 1 y clase 2), supongamos que estamos interesados en obtener el hiperplano que nos permita separar de la manera más optima posible los dos grupos. Los datos con los que entrenamos el sistema serán una serie de vectores etiquetados de la forma $\{\vec{x}_i, t_i\}_{i=1,\dots,T}$, donde:

$\vec{x}_i \in R^d$ es el vector de observaciones en un espacio de dimensión d

$t_i \in \{-1,1\}$ en función de si el vector i es de la clase 1 o 2, respectivamente

Así que con esta información se construirá un hiperplano de separación que divida el espacio R^d en dos regiones. A la hora de formular el problema se parte con el hecho de que todos los datos de entrenamiento cumplen una de las siguientes restricciones:

$$\vec{x}_i \cdot \vec{w} + b \geq +1 \quad \text{si } t_i = +1, \tag{4.28}$$

$$\vec{x}_i \cdot \vec{w} + b \geq -1 \quad \text{si } t_i = -1, \tag{4.29}$$

donde $\vec{w}$ es el vector normal al hiperplano de separación y b es una constante. Combinando las dos restricciones anteriores en una llegamos a la siguiente expresión:

$$t_i(\vec{x}_i \cdot \vec{w} + b) \geq +1, \qquad i = 1, \ldots, T. \tag{4.30}$$

En la figura siguiente hemos ilustrado un ejemplo en dos dimensiones con puntos pertenecientes a dos clases donde se han propuesto una serie de hiperplanos de separación:

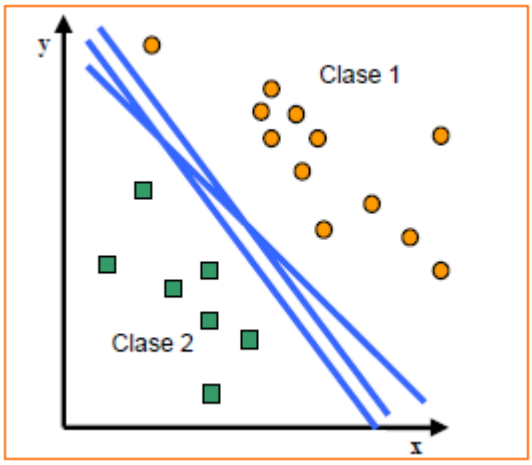

Figura 21. *Posilbes hiperplanos de separación para puntos bidimensionales pertenecientes a dos tipos de clase*

Sin embargo, el algoritmo intentará encontrar el hiperplano que maximiza la distancia m entre las dos clases o grupos (hiperplano de margen máximo):

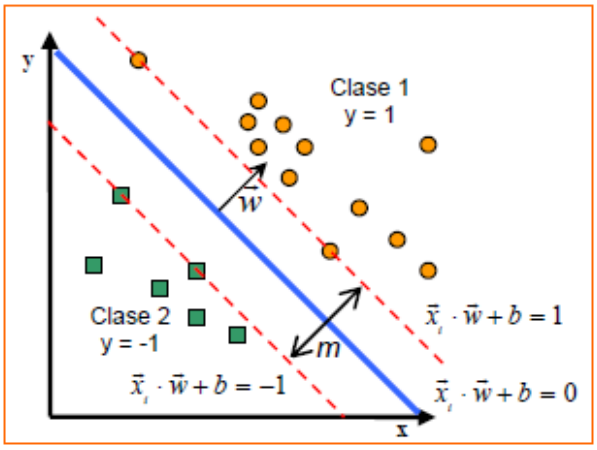

Figura 22. *Hiperplano óptima para el caso representado en la figura 20.*

A la luz de la naturaleza geométrica de la figura anterior, se entiende que los vectores $\vec{x}_i$ que cumplen $\vec{x}_i \cdot \vec{w} + b = 1$ o $\vec{x}_i \cdot \vec{w} + b = -1$, representados en la figura anterior en la línea roja discontinua se les denomina vectores soporte: son los vectores de entrenamiento que se encuentran más próximos al hiperplano y sobre los que éste descansa (esto es, "soportan el peso" del hiperplano).

El problema de la búsqueda del mejor hiperplano de separación se traduce por lo tanto en un problema de optimización. Existen varios métodos para su resolución, siendo uno de los más conocidos el paso a la formulación de Lagrange. Sin embargo, en este proyecto no se ha descrito este proceso, puesto que su complejidad es muy alta y el software ALIZE dispone de una función que implementa este algoritmo internamente.

Sin embargo, debido a que puede ser imposible hallar un hiperplano que separe los puntos de amblas clases en la dimensión de $\vec{x}_i$, SVM [Campbell et al., 2006] transforma los vectores de entrenamiento a una dimensión mayor donde resulta más fácil encontrar tal hiperplano. De tal forma que realiza el mismo proceso de optimización que se ha explicado en los párrafos anteriores pero en un espacio vectorial con más dimensiones donde encontrar una frontera entre dos clases es un mecanismo más simple. La transformación se hace a través de la función de un determinado kernel $\boldsymbol{\Phi}(\vec{x}_i)$ que mapea el vector $\vec{x}_i$ a una dimensión mayor. En la siguiente figura se ilustra la necesidad del kernel para encontrar el hiperplano:

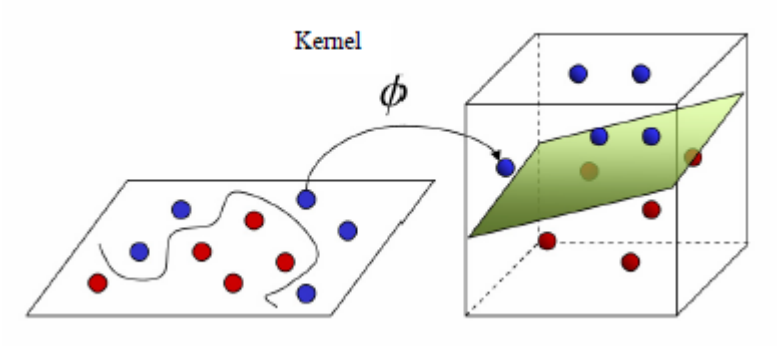

Figura 23. *Transformación de un espacio bidimensional a otro tridimensional a través de la función interna de un Kernel. Figura adaptada de [Alejandro Abejón, 2007].*

El kernel (hablaremos más detenidamente sobre él en el siguiente apartado) $K(\vec{x}_i, \vec{x}_j)$ para ser considerado como tal debe de satisfacer las condiciones de Mercer, una de ellas es que debe ser expresado como:

$$K(\vec{x}_i, \vec{x}_j) = \langle \boldsymbol{\Phi}(\vec{x}_i), \boldsymbol{\Phi}(\vec{x}_j) \rangle = \boldsymbol{\Phi}(\vec{x}_i)^T \cdot \boldsymbol{\Phi}(\vec{x}_j), \tag{4.31}$$

donde $\langle\ ,\ \rangle$ representa el producto interno.

Después de haber averiguado el plano de separación con los datos de entrenamiento y dado un vector determinado $\vec{y}$, el objetivo final es averiguar a qué clase pertenece dicho vector. Por lo tanto, un SVM básicamente es un clasificador binario discriminativo construido como una combinación lineal de funciones kernel $K(\vec{x}_i, \vec{x}_j)$ de la forma:

$$f(\vec{y}) = \vec{w}^T \cdot \boldsymbol{\Phi}(\vec{y}) + b = \sum_{s=1}^{S} \alpha_s t_s \; K(\vec{x}_s, \vec{y}) + b = \sum_{s=1}^{S} \alpha_s t_s \; \boldsymbol{\Phi}(\vec{x}_s)^T \cdot \boldsymbol{\Phi}(\vec{y}) + b. \quad (4.32)$$

$\vec{x}_s$ son los vectores soportes que se han obtenido durante el entrenamiento mediante el proceso de optimización (hemos supuesto que obtenemos S vectores soporte en el training) y los t_i son sus correspondientes etiquetas dependiendo de la clase a la que pertenezcan. Dados $\vec{x}_s$ e $\vec{y}$, $K(\vec{x}_s, \vec{y})$ se puede interpretar como una función que mide o evalúa el grado de similitud entre estos dos vectores pero no en sus correspondientes espacios vectoriales, sino en uno de mayor dimensión. Luego el kernel $K(\vec{x}_s, \vec{y})$ implícitamente lleva arraigado una transformación a otro espacio vectorial. El proceso de optimización (búsqueda del hiperplano óptimo) impone la condición de que $\sum_{s=1}^{S} \alpha_s t_s = 0$ y $\alpha_s > 0$. Partiendo de que los vectores pertenecientes al hiperplano anulan la ecuación (4.32), ésta se traduce en una forma muy útil de puntuar o comprobar a qué clase pertenece el vector de entrada: si $f(\vec{y}) > 0$ asumiremos que pertenece a la clase 1 y si $f(\vec{y}) < 0$ tomaremos la hipótesis alternativa.

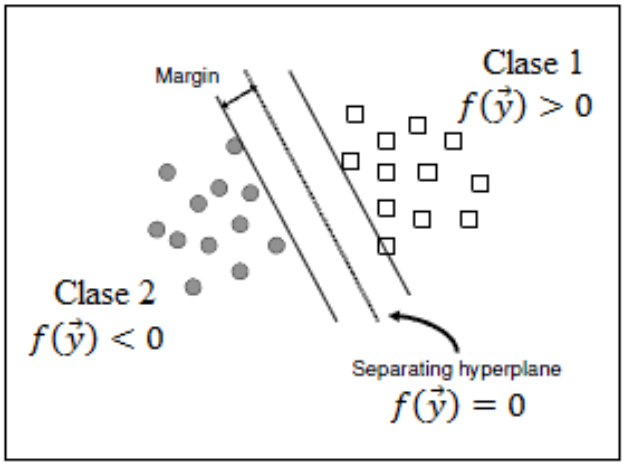

Figura 24. *Ejemplo de puntuación en máquinas de vectores soporte.*

Toda la teoría explicada sobre máquinas de vectores soporte concuerda muy bien con la estructura de un sistema de verificación de locutor en la que tenemos usuarios (clase 1) e impostores (clase 2). Además, según las dos etapas, en el entrenamiento, dado un fichero de un usuario y un conjunto de extractos pertenecientes a impostores, se obtiene el hiperplano frontera característico de dicho locutor. Finalmente, con un segmento de test, evaluaremos $f(\vec{y})$ para verificar si el locutor es realmente quién dice ser. De hecho, desarrollando un poco más (4.32):

$$f(\vec{y}) = \sum_{s=1}^{S} \alpha_s t_s \; K(\vec{x}_s, \vec{y}) + b = \sum_{s\in\{s|t_s=1\}} \alpha_s \, K(\vec{x}_s, \vec{y}) - \sum_{s\in\{s|t_s=-1\}} \alpha_s \, K(\vec{x}_s, \vec{y}). \quad (4.33)$$

Con la expresión anterior realmente lo que se está haciendo es puntuar por un lado la similitud entre el vector de evaluación y los vectores soportes del supuesto usuario y por otro la semejanza entre el vector de test y los vectores soportes correspondientes a los impostores. Si la persona es realmente quién dice ser, $\sum_{s\in\{s|t_s=1\}} \alpha_s \, K(\vec{x}_s, \vec{y})$ debería de proporcionarnos un valor relativamente alto mientras $\sum_{s\in\{s|t_s=-1\}} \alpha_s \, K(\vec{x}_s, \vec{y})$ saldría aproximadamente cero, con lo cual el score total quedará por encima del umbral de nuestro sistema y aceptaremos a dicho usuario.

*4.3.2 **SVM usando GMM supervector***

Dentro de las máquinas de vectores, la técnica que se ha analizado en este proyecto y de la que hace uso el software recibe el nombre de GMM Supervector Linear Kernel [Campbell et al., 2006c]. El hecho de que aparezca GMM en el título se debe a que juega un papel fundamental en esta nueva técnica. En el apartado anterior hemos comprobado que toda la teoría general de SVM concuerda perfectamente con la estructura clásica de un sistema de verificación de locutor. Sin embargo, tenemos que definir dos puntos: ¿cómo pasamos de un espació a otro de dimensión mayor donde obtengamos de manera más sencilla y óptima el hiperplano? ¿Y qué kernel vamos a usar?

Para responder a la primera pregunta, nos aprovecharemos de GMM. Con ésta técnica, dado un segmento de entrenamiento $\boldsymbol{X} = \{\vec{x}_t\}_{t=1,\dots,N_x}$ y el modelo universal $\bar{X}$, se explicó que mediante adaptción MAP creábamos el modelo gaussiano del locutor:

$$X = \{w_i, \vec{\mu}_i, \Sigma_i\}, i = 1, \dots, M. \quad (4.34)$$

Partiendo de que los vectores $\vec{x}_t$ eran D-dimensionales, una forma muy útil y simple de incrementar la dimensión es concatenando los vectores media de las M componentes gaussianas formando lo que se conoce en la literatura como un *"supervector"* de dimensión $M \cdot D$.

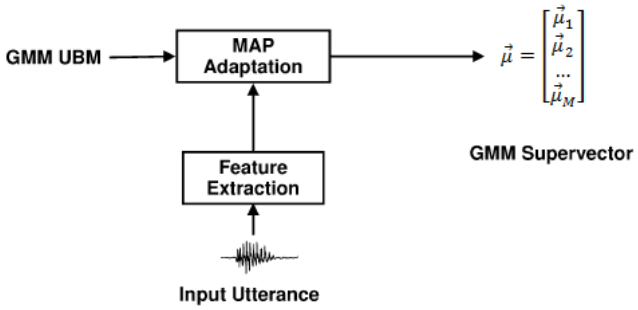

Figura 25. *Ilustración de la obtención del GMM-supervector. Figura obtenida de [Campbell et al., 2006c].*

De (4.33) se puede observar que en SVM para obtener la puntuación correspondiente a un vector de test $\vec{y}$ hay que someterlo previamente al kernel, y como bien sabemos el kernel traslada dicho vector a un espacio de mayor dimensión, con lo cual los vectores de evaluación también deben ser transformados a dicho espacio. Ello conlleva que, a diferencia de GMM-UBM, en donde los extractos de test se utilizaban únicamente para computar la puntuación final a través de la función de distribución condicionada $p(\boldsymbol{Y}|X)$ (véase ecuación 4.24), en este caso también tenemos que generar el modelo de la locución de test mediante adaptación MAP, puesto que es la única forma que disponemos para poder crear su supervector correspondiente.

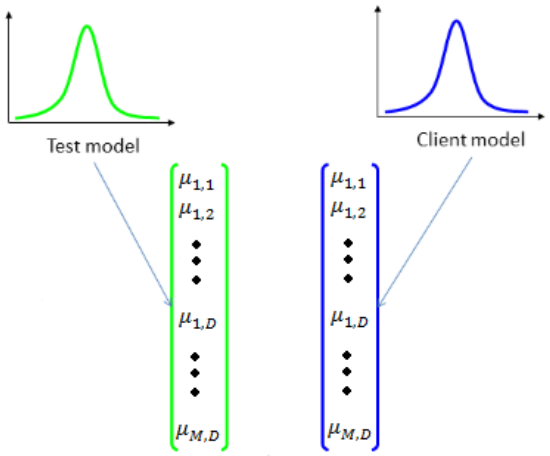

Figura 26. *Obtención de los supervectores de entrenamiento (azul) y de test (verde) en SVM-GMM.*

Una vez llegado hasta aquí, únicamente nos falta definir el kernel. Dado dos vectores $\vec{x}$ e $\vec{y}$, donde mediante adaptación MAP obtenemos sus correspondientes modelos $X = \{w_i, \vec{\mu}_i^X, \Sigma_i\}$ y $Y = \{w_i, \vec{\mu}_i^Y, \Sigma_i\}$ con $i = 1, \dots, M$ (donde los pesos y la matriz de varianza no varían puesto que no fueron modificados con respecto al UBM durante la adaptación MAP), representamos el kernel asociado a estos dos vectores como:

$$k(\vec{x},\vec{y}) = (\vec{\mu}^X \boldsymbol{R})^T \cdot (\vec{\mu}^Y \boldsymbol{R}), \tag{4.35}$$

donde $\vec{\mu}^X$ y $\vec{\mu}^Y$ representan los correspondientes supervectores del segmento de entrenamiento y test, respectivamente, siendo $\boldsymbol{R}$ una matriz diagonal de la forma:

$$\boldsymbol{R} = \begin{bmatrix} \sqrt{w_1}\Sigma_1^{-\frac{1}{2}} & 0 & \dots & \dots & 0 \\ 0 & \ddots & 0 & \dots & 0 \\ \vdots & 0 & \sqrt{w_i}\Sigma_i^{-\frac{1}{2}} & 0 & 0 \\ \vdots & \vdots & 0 & \ddots & 0 \\ 0 & 0 & 0 & 0 & \sqrt{w_M}\Sigma_M^{-\frac{1}{2}} \end{bmatrix}. \tag{4.36}$$

Por lo tanto, en este proyecto hemos trabajado con un kernel lineal, puesto que se define mediante el producto escalar entre dos vectores, los cuales a su vez son previamente obtenidos mediante el producto de un vector y una matriz.

La siguiente figura resume de manera detallada la estructura de la técnica de SVM que se ha implementado en este proyecto (GMM supervector):

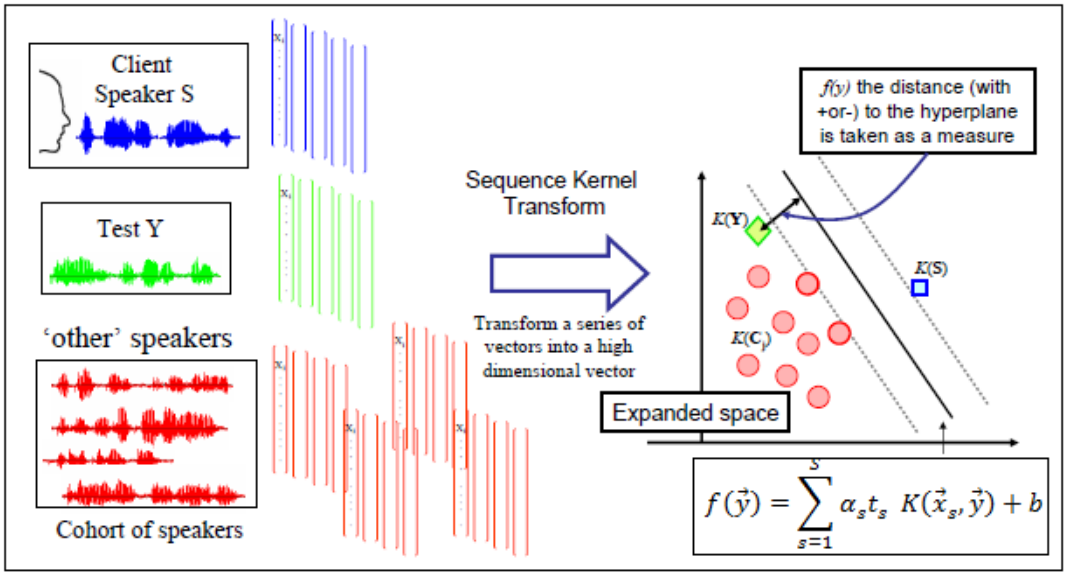

Figura 27. *Principios generales de SVM aplicados a Verificación Automática de Locutor. Figura adaptada de [B. Fauve, 2009].*

En ella podemos ver cómo dado un fichero de entrenamiento y una cohorte de impostores (siempre será la misma para cualquier usuario que vayamos a entrenar), tras crear sus correspondientes supervectores, mediante un proceso de optimización obtendremos un hiperplano que separa de la forma más óptima y sofisticada al usuario con el grupo de impostores. En la optimización, se obtendrán una serie de vectores soporte $\vec{x}_s$ acompañados de su correspondiente peso α_s y etiqueta t_s, junto con un valor

característico del hiperplano b necesario para la fase de evaluación. Así que habiendo entrenado un SVM por medio de $\boldsymbol{X} = \{\vec{x}_t\}_{t=1,\dots,N_X}$ y dado un segmento de test $\boldsymbol{Y} = \{\vec{y}_t\}_{t=1,\dots,N_Y}$, obtenemos la puntuación final mediante:

$$Score(\boldsymbol{X},\boldsymbol{Y}) = \prod_{t=1}^{Nx}\left(\sum_{s=1}^{S}\alpha_s t_s\ K(\vec{x}_s,\vec{y}_t) + b\right) = \prod_{t=1}^{Nx}\left(\sum_{s=1}^{S}\alpha_s t_s(\vec{\mu}^s\boldsymbol{R})^T\cdot(\vec{\mu}^{y_t}\boldsymbol{R})\ + b\right)$$

, con $\vec{\mu}^s$ y $\vec{\mu}^{y_t}$ representando a los supervectores asociados a uno de los locutores (usuario o impostor) que el proceso de optimización identificó como *"soporte del hiperplano"* y al locutor que está siendo testeado, respectivamente. Al igual que con GMM-UBM, podemos optar por trabajar con el logaritmo del score.

4.4 NAP (Nuissance Attribute Projection)

En el apartado 3.4 vimos que existen diferentes técnicas o métodos para intentar reducir los efectos nocivos de la variabilidad del locutor y del canal (RASTA, CMS, CMV o Feature warping). Estas técnicas tradicionalmente han demostrado proporcionar excelentes resultados para GMM. Sin embargo, tras la fuerte intrusión de las máquinas de vectores, se hace necesaria la búsqueda de técnicas adicionales que intenten compensar estos problemas.

El principal inconveniente del kernel lineal de SVM es que es muy sensible a las variaciones de canal, agravando el problema de la variabilidad intercesión (diferentes muestras de habla asociadas a una misma identidad se caracterizan por presentar grandes diferencias). Teniendo en cuenta que las condiciones de canal (tipo de transductor, canal de transmisión, situación conversacional o entorno) pueden ser diferentes para el entrenamiento y la evaluación, y aunque dichas condiciones fuesen exactamente iguales los dos supervectores medias no tendrían por qué coincidir, se hace necesario intentar minimizar la susceptibilidad del kernel a diferentes condiciones de sesión.

NAP [A. Solomonoff, et al., 2004] intentar captar el subespacio vectorial donde el canal está afectando de manera más notable al sistema. De esta manera, una vez identificado dicho subespacio, mediante una transformación determinada estaremos a disposición de poder suprimir los efectos nocivos del canal o mejor dicho, del desajuste de las condiciones de sesión. En [P. Kenny, G. Boulianne and P. Dumouchel, 2005] se considera que las dimensiones de dicho subespacio vectorial son mucho más pequeñas que las del espacio de los supervectores.

Por lo tanto, la cuestión fundamental a la que se debe hacer frente es cómo caracterizamos dicho subespacio vectorial que está disminuyendo el rendimiento del sistema. Para ello, previamente al entrenamiento o evaluación, para un grupo de usuarios (es totalmente irrelevante si dichos usuarios participarán en el entrenamiento o en el test) necesitamos varias grabaciones de cada uno de ellos en diferentes situaciones o momentos. Así que supongamos que tenemos N_S locutores con h_i segmentos de audio para cada uno de ellos:

$$\{\vec{\mu}^{1,s_1} \dots \vec{\mu}^{h_1,s_1} \dots \vec{\mu}^{1,s_{N_S}} \dots \vec{\mu}^{h_{N_S},s_{N_S}}\}, \tag{4.37}$$

con $\vec{\mu}^{h_i,s_j}$ representando el supervector (obtenido con la misma metodología que explicamos en SVM) correspondiente al segmento de audio i-ésimo del j-ésimo locutor. En la figura anterior hemos ilustrado un ejemplo con 3 usuarios únicamente y con $h_1 = 3$, $h_2 = 4$ y $h_3 = 5$.

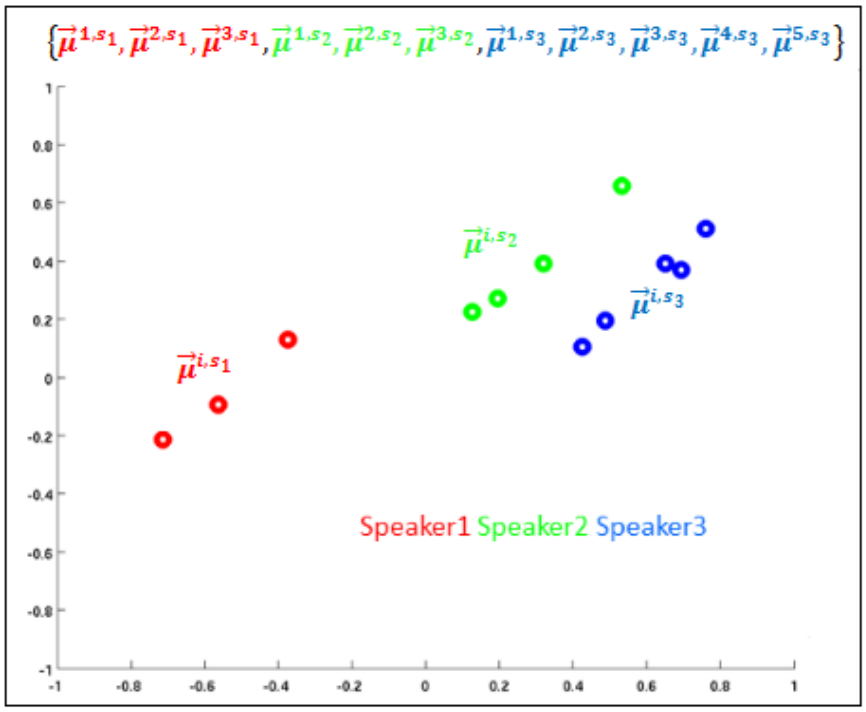

Figura 28. *Ejemplo de NAP con distintas grabaciones en diferentes condiciones para 3 locutores.*

Para cada usuario, se calcula la media de todos sus supervectores:

$$\overline{\vec{\mu}^{s_l}} = \frac{1}{h_l}\sum_{i=1}^{h_l} \vec{\mu}^{i,s_l}, \tag{4.38}$$

con $l = 1, \dots, N_S$.

Y después, a cada uno de los supervectores de cada usuario se le resta su correspondiente media:

$$\widetilde{\vec{\mu}^{i,s_l}} = \vec{\mu}^{i,s_l} - \overline{\vec{\mu}^{s_l}}, \tag{4.39}$$

donde $i = 1, \dots, h_l$.

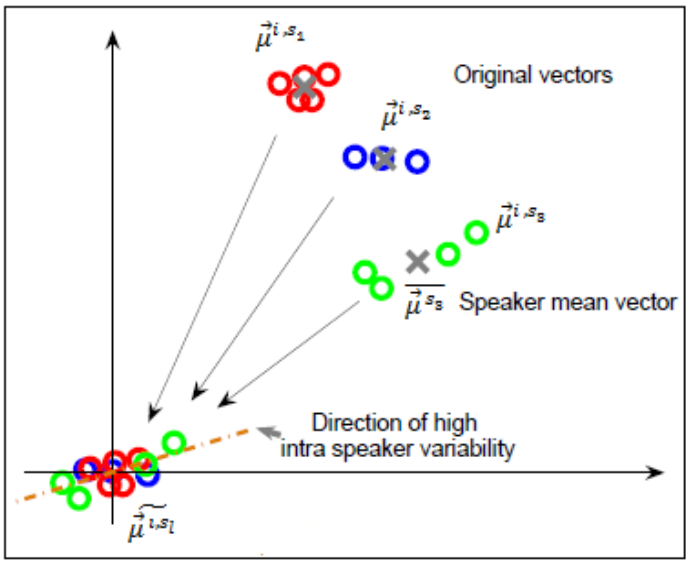

Figura 29. *Resta de todas las locuciones de cada usuario por su correspondiente supervector media en NAP. Figura adaptada de [B. Fauve, 2009].*

Tras la operación anterior, se procede a la creación de la matriz $\boldsymbol{M}$ de la forma:

$$\boldsymbol{M} = \left[\overline{\vec{\mu}^{1,s_1}} \dots \overline{\vec{\mu}^{h_1,s_1}} \dots \overline{\vec{\mu}^{1,s_{N_S}}} \dots \overline{\vec{\mu}^{h_{N_S},s_{N_S}}}\right]. \tag{4.40}$$

Siguiendo con la nomenclatura de apartados anteriores, donde los supervectores eran MD-dimensionales, la matriz $\boldsymbol{M}$ tendrá una dimensión de $MD \times N$, con $N = h_1 + \cdots + h_{N_S}$. A la luz de lo anteriormente descrito, esta matriz almacena la información sobre la variabilidad de las diferentes sesiones con respecto a sus correspondientes medias en el espacio dimensional de los supervectores. A continuación se calculan K autovectores (los que tengan los K autovalores más altos) de la matriz de covarianza $\boldsymbol{MM^T}$. Dichos autovectores suelen aparecer en la literatura etiquetados bajo el nombre de *"eigenchannels"*, y en cierta medida reflejan las direcciones que más se ven afectadas por el canal. El conjunto de estos K autovectores constituyen una base para el subespacio vectorial que queremos descartar debido a su fuerte dependencia con los efectos del canal. Debido a la baja dimensionalidad de este subespacio vectorial, se debe cumplir que $K \ll MD$. Si $\boldsymbol{S}$ es la matriz de dimensión $MD \times K$ cuyas columnas se

corresponden con los K autovectores, denominaremos como $\boldsymbol{P} = \boldsymbol{I} - \boldsymbol{SS^T}$ a la matriz de proyección característica de NAP que multiplica a los supervectores de entrenamiento y test.

Si $\vec{\mu}$ representa el supervector asociado a un fichero de evaluación o entrenamiento, antes de procesar la información tal y como lo hace SVM, se lleva a cabo una transformación de $\vec{\mu}$ para eliminar los componentes nocivos del canal, de tal forma que el nuevo supervector queda representado como:

$$\hat{\vec{\mu}} = \vec{\mu}\,\boldsymbol{P} = \vec{\mu}(\boldsymbol{I} - \boldsymbol{SS^T}) = \vec{\mu}\boldsymbol{I} - \vec{\mu}\boldsymbol{SS^T} = \vec{\mu} - \vec{\mu}\boldsymbol{SS^T}, \tag{4.41}$$

De acuerdo con la ecuación anterior, el término $\vec{\mu}\boldsymbol{SS^T}$ que se le resta al supervector $\vec{\mu}$ se puede interpretar como la proyección de $\vec{\mu}$ en el subespacio del canal, cuya base lo constituyen los K autovectores más altos obtenidos anteriormente y que deseamos eliminar puesto que es información de canal y no de locutor. Una vez obtenido $\hat{\vec{\mu}}$, se está en disposición de aplicar el kernel lineal de SVM.

Resumiendo, NAP presenta la misma estructura que SVM salvo con la excepción de que en lugar de trabajar con el kernel lineal definido en el apartado 4.2 de SVM, trabaja con un nuevo kernel menos sensible a los desajustes debido a las condiciones de canal o de sesión:

$$k'(\vec{x}, \vec{y}) = (\vec{\mu}^X \boldsymbol{R}')^T \cdot (\vec{\mu}^Y \boldsymbol{R}'), \tag{4.42}$$

donde $\boldsymbol{R}' = \boldsymbol{PR}$, siendo $\boldsymbol{P}$ la matriz de proyección y $\boldsymbol{R}$ la matriz del kernel característica que utilizábamos en SVM (ecucación (4.36)).

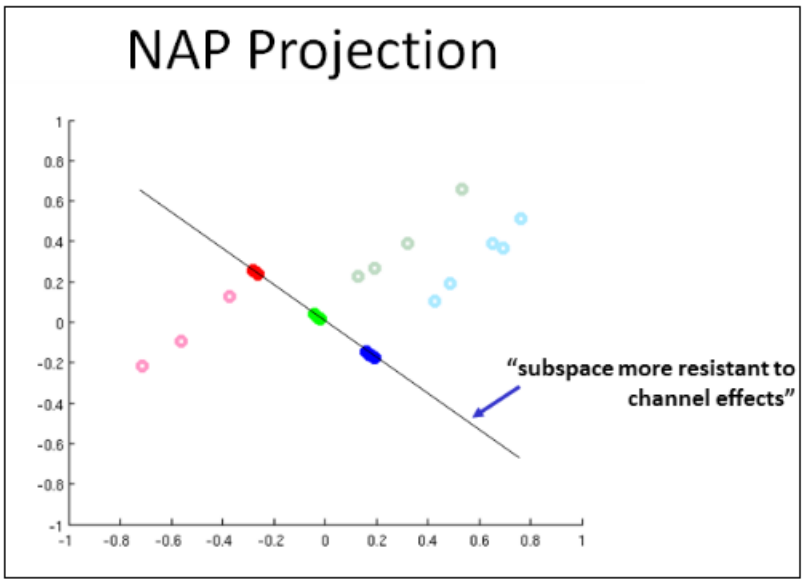

Figura 30. *Proyección en NAP para un ejemplo de supervectores bidimensionales.*

4.5 FA (Factor Analisis)

Esta técnica, al igual que NAP, intenta enfrentar el grave problema de la variabilidad intercesión, aunque opera mediante una metodología distinta. Fue introducida en el campo de la verificación de locutor por [Kenny et al., 2003], y posteriormente modificado y simplificado por [Vogt and Sridharan, 2006]. Como se podrá comprobar en breve, a diferencia de las tres técnicas anteriores, se puede implementar tanto a nivel generativo (mediante el tradicional GMM-UBM) o discriminativo (combinado con SVM).

Sea s un determinado locutor y h una grabación o sesión determinada correspondiente a dicho locutor, en FA el modelo de dicho locutor en el espacio de los supervectores (concatenación de las medias) se descompone de la siguiente forma:

$$\vec{\mu}^{(s,h)} = \vec{\mu}_{UBM} + \boldsymbol{D}\vec{y}_s + \boldsymbol{U}\vec{x}_{(s,h)}. \tag{4.43}$$

El primer término de la ecuación anterior se corresponde con el supervector de medias del modelo UBM y representa la componente independiente de sesión. Naturalmente $\vec{\mu}_{UBM}$ tiene MD componentes. El segundo término (componente dependiente de locutor) hace referencia al desplazamiento del offset de la media $\vec{\mu}_{UBM}$ como resultado de la adaptación clásica MAP, solo que en este caso, por conveniencia lo hemos representado de otra forma. La matriz diagonal $\boldsymbol{D}$ de dimensiones $MD \times MD$ cumple la ecuación $I = \tau \boldsymbol{D}^t \boldsymbol{\Sigma}^{-1} \boldsymbol{D}$, con $\boldsymbol{\Sigma}^{-1}$ representando la inversa de la matriz formada por la concatenación diagonal de las matrices de covarianza de cada modelo gaussiano $\boldsymbol{\Sigma_i}, i = 1, \dots, M$. τ recibe el nombre de factor de relevancia. $\vec{y}_s$ presenta las mismas dimensiones que $\vec{\mu}_{UBM}$. Hasta aquí no hemos modificado nada con respecto a la tradicional técnica de GMM, salvo con la excepción de haber concatenado las medias y trabajar en el espacio de los supervectores. El tercer término (componente dependiente de sesión) representa la variabilidad de canal debido a las diferentes condiciones de sesión o grabación h. El rango de $\boldsymbol{U}$ es de $MD \times K$ (con $K \ll MD$) por lo que $\vec{x}_{(s,h)}$ es un vector de dimensiones $K \times 1$.

Por lo tanto, la idea fundamental de FA es aislar perfectamente estos tres componentes básicos. Dado los parámetros fijos ($\vec{\mu}_{UBM}, \boldsymbol{D}, \boldsymbol{U}$), a través de un algoritmo (explicado y analizado con detalle en [Vogt and Sridharan, 2006]) se lleva cabo el aislamiento de los tres componentes maximizando la función de probabilidad de los datos dado dicha descomposición. Si $\boldsymbol{U} = \boldsymbol{0}$, dicho algoritmo se transforma en la tradicional adaptación MAP que analizamos en la sección de GMM.

Finalmente, con respecto a FA, queda únicamente como tarea pendiente la definición del método que vamos a llevar a cabo para obtener el scoring. Tal y como anunciamos en el inicio de este apartado, FA presenta la peculiar característica de que la puntuación final puede ser obtenida mediante GMM-UBM (ecuación (4.25)) o bien utilizando la poderosa herramienta de SVM.

4.5.1 FA-GMM-UBM (Factor Analysis puntuando con GMM-UBM)

Dado el segmento de entrenamiento $\boldsymbol{X}$, obtenemos el modelo correspondiente al usuario mediante:

$$\vec{\mu}^{(s_X,h_X)} = \vec{\mu}_{UBM} + \boldsymbol{D}\vec{y}_{s_X} + \boldsymbol{U}\vec{x}_{(s_X,h_X)}. \quad (4.44)$$

Por lo tanto, una vez detectada la componente de variabilidad de canal $\boldsymbol{U}\vec{x}_{(s_X,h_X)}$, debemos restársela a $\vec{\mu}^{(s_X,h_X)}$ para eliminar dicho defecto.

Puesto que en GMM no se trabajaba en el espacio de los supervectores, sino en el de los modelos gaussianos, tenemos que descomponer $\vec{\mu}^{sx}$ y $\vec{\mu}_{UBM}$ en sus correspondientes medias asociadas a cada una de las M componentes gaussianas. De esta forma ya tenemos todo predispuesto para aplicar la ecuación (4.25).

Sin embargo, hemos rectificado los efectos de canal para el fichero de entrenamiento, pero no se ha llevado a cabo ninguna modificación en el segmento de evaluación $\boldsymbol{Y} = \{\vec{y}_t\}_{t=1,\dots,N_Y}$. Cualquier variabilidad entre las condiciones de canal para el fichero de entrenamiento y test puede llevarnos a una decisión equivocada. Sin embargo, en GMM-UBM las tramas o vectores cepstrales del fichero de test eran directamente computadas en la función de probabilidad condicionada, sin previamente haber creado un correspondiente modelo estadístico donde tenemos la posibilidad de llevar a cabo la compensación. La única posibilidad es llevarla a cabo en el nivel de los coeficientes cepstrales, y para ello el software de este proyecto utiliza la alternativa propuesta por [Vair et al., 2006]:

$$\hat{\vec{y}}_t = \vec{y}_t - \sum_{i=1}^{M} p(i|\vec{y}_t)\, \boldsymbol{U}_i \vec{x}_{(s_Y,h_Y)}, \quad (4.45)$$

donde $p(i|\vec{y}_t)$ representa la probabilidad de ocupación de la componente gaussiana i-ésima en el vector $\vec{y}_t$. Puesto que el rango de $\boldsymbol{U}$ era de $MD \times K$, $\boldsymbol{U}_i$ equivale a la submatriz de $\boldsymbol{U}$ asociada a la componente gaussiana i-ésima de dimensión $D \times K$.

4.5.2 FA-SVM (Factor Analysis puntuación con SVM)

Con los segmentos $\boldsymbol{X}$ e $\boldsymbol{Y}$ de entrenamiento y test, respectivamente, obtenemos su descomposición característica mediante FA:

$$\vec{\mu}^{(s_X,h_X)} = \vec{\mu}_{UBM} + \boldsymbol{D}\vec{y}_{s_X} + \boldsymbol{U}\vec{x}_{(s_X,h_X)}, \quad (4.46)$$

$$\vec{\mu}^{(s_Y,h_Y)} = \vec{\mu}_{UBM} + \boldsymbol{D}\vec{y}_{s_Y} + \boldsymbol{U}\vec{x}_{(s_Y,h_Y)}. \quad (4.47)$$

Tras restarles a ambas descomposiciones la componentes de canal, habría que añadir que la cohorte de impostores necesaria en SVM también debe ser sometido a este proceso de descomposición mediante FA, con lo que tendríamos todos los elementos necesarios en el espacio de los supervectores, donde podemos obtener el hiperplano de separación y el kernel de puntuación característicos de SVM y que detallamos en la sección 4.3.