UNIVERSIDAD AUTÓNOMA DE MADRID
ESCUELA POLITÉCNICA SUPERIOR

MÁQUINAS DE VECTORES SOPORTE (SVM) PARA RECONOCIMIENTO DE LOCUTOR E IDIOMA

-PROYECTO FIN DE CARRERA-

Ismael Mateos García
Julio de 2007

MÁQUINAS DE VECTORES SOPORTE (SVM) PARA RECONOCIMIENTO DE LOCUTOR E IDIOMA

AUTOR: Ismael Mateos García
TUTOR: Joaquín González Rodríguez

Área de Tratamiento de Voz y Señales
Dpto. de Ingeniería Informática
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Julio de 2007

PROYECTO FIN DE CARRERA

Título: *Máquinas de vectores soporte (SVM) para reconocimiento de locutor e idioma*

Autor: D. Ismael Mateos García

Tutor: D. Joaquín González Rodríguez

Tribunal:

Presidente: D. Javier Ortega García

Vocal: D. José Colás Pasamonte

Vocal secretario: D. Joaquín González Rodríguez

Fecha de lectura:

Calificación:

Palabras clave

Reconocimiento biométrico de locutor, reconocimiento de idioma, máquinas de vectores soporte, clasificación, regresión, NIST SRE 2006, NIST LRE 2005, MFCC, SDC, fusión.

Resumen

En este proyecto se presentan nuevos métodos además del estado del arte de las técnicas existentes para reconocimiento de locutor e idioma. El trabajo se centra en SVM *(Support Vector Machines)*, una técnica consolidada en el campo del reconocimiento de patrones, cuya eficiencia en este tipo de problemas ha sido ampliamente demostrada en los últimos años. Se realizará un examen completo de estos sistemas, desde las formas de extraer las características de la señal de voz, el conjunto de datos de entrenamiento, influencia de distintas variables en los modelos entrenados, fusión de sistemas, etc.

En la parte experimental del proyecto se llevan a cabo diversas pruebas con el fin de obtener resultados objetivos de la experimentación con todo este tipo de sistemas. Los experimentos llevados a cabo se realizan siguiendo el protocolo de evaluaciones NIST *(National Institute of Standards and Technology)*. El sistema de reconocimiento de locutor se evaluará sobre el protocolo NIST SRE *(Speaker Recognition Evaluation)* 2006, el sistema de reconocimiento de idioma sobre NIST LRE *(Language Recognition Evaluation)* 2005. Las evaluaciones más recientes realizadas por NIST a día de hoy en ambos campos.

Por último, se presentan las conclusiones y proponen las líneas de trabajo futuras.

Abstract

In this master's thesis we present new methods and the state of the art of the existing techniques for speaker and language recognition. Our work is focused on SVM *(Support Vector Machines)*, a well-know pattern recognition technique which has demonstrated its adequacy to these problems in the last years. We report a study of the state of the art regarding speaker and language recognition reviewing, feature extraction process of the speech signal, training set selection, system behavior with respect some variables, fusion techniques, etc.

Along the experimental part of the master's thesis several experimental results showing the system behavior are reported. Experiments have been performed using the evaluation protocols proposed by NIST *(National Institute of Standards and Technology).* On the one hand, speaker recognition system will be evaluated over NIST SRE *(Speaker Recognition Evaluation)* 2006 protocol. On the other hand, the language recognition system over NIST LRE *(Language Recognition Evaluation)* 2005 protocol. These are the most recent evaluations realized by NIST in both fields.

Finally, conclusions are drawn, and future lines of work are proposed.

Agradecimientos

Me gustaría agradecer en primer lugar a mi tutor, Joaquín González, la oportunidad de colaborar en un grupo de investigación puntero en el reconocimiento biométrico, como es el ATVS.

También me gustaría agradecer a todos los miembros del grupo el apoyo prestado durante la realización de este Proyecto Fin de Carrera. Todos de alguna manera han contribuido a que este trabajo viese la luz pero especialmente Ignacio López, cuyo consejo y ayuda fue determinante a la hora de realizar este proyecto.

Por último me gustaría mencionar el apoyo de mis padres a lo largo de estos años, sin ellos y su ánimo estos estudios no hubieran podido llevarse a cabo.

Ismael Mateos García
Julio de 2007.

Este proyecto ha sido realizado en el Área de Tratamiento de Voz y Señales (ATVS) en la Escuela Politécnica superior de la Universidad Autónoma de Madrid. El proyecto ha sido financiado parcialmente por el Ministerio de Educación y Ciencia a través del proyecto TEC2006-13170-C02-01.

ÍNDICE

Índice de Figuras

Índice de Tablas

1. Introducción

1.1 Motivación del proyecto

Los rápidos avances llevados a cabo en el campo de las redes de comunicación y la movilidad han propiciado la aparición de un nuevo conjunto de *tele-aplicaciones*. Dentro de este nuevo conjunto están englobadas todas aquellas aplicaciones que permiten una comunicación remota entre el usuario y cualquier tipo de sistema. La banca telefónica o la venta de entradas *on-line*, son sólo algunos ejemplos.

Todo este tipo de aplicaciones requiere una autenticación por parte del usuario. Tradicionalmente, se empleaban esquemas de identificación clásicos, los cuales hacían uso de claves secretas, códigos o llaves. En la actualidad, los sistemas basados en reconocimiento biométrico se presentan como una buena alternativa a los métodos clásicos.

Entre las principales ventajas de este tipo de sistemas podemos destacar, su bajo coste de mantenimiento, el alto nivel de seguridad que ofrecen y la comodidad para el usuario. Mientras que las claves de los sistemas clásicos eran fácilmente olvidables los rasgos biométricos, como por ejemplo la voz, huella dactilar, etc. son características que siempre porta consigo el individuo.

Para que una característica o comportamiento sea considerado rasgo biométrico deberá cumplir una serie de propiedades, que detallaremos en la sección 2.2. La voz es un rasgo biométrico que además de cumplir esta serie de propiedades cuenta con muchas ventajas, entre ellas podemos destacar su fácil adquisición. Puede ser adquirida de una manera muy sencilla, sin métodos invasivos ni dispositivos especializados, lo que hace de la voz un rasgo biométrico ideal para aplicaciones a distancia.

Otra de las características de la señal de voz es la gran cantidad de información que contiene: identidad del locutor, idioma, edad, estado de ánimo, nivel de educación, etc. Los sistemas de reconocimiento biométrico harán uso de la información sobre la identidad del locutor para identificar a los usuarios. Por otra parte, la información acerca del idioma del hablante será importante para aplicaciones orientadas a la seguridad, información, etc.

El reconocimiento automático del idioma comparte muchas técnicas con el reconocimiento de locutor, por tanto ambos problemas podrán ser abordados de un modo similar.

Los sistemas de reconocimiento basados en voz tienen algunas limitaciones, las más importantes son las debidas al ruido y al canal. Distintos tipos de micrófonos, ruido de ambiente, limitaciones en los medios de transmisión, etc. son factores que deberemos tener en cuenta e intentar compensar de manera que el comportamiento del sistema se vea afectado lo menos posible.

1.2 Objetivo y enfoque

El presente proyecto abordará dos problemas relacionados con la señal de voz, por un lado el reconocimiento biométrico basado en dicha señal y por otro el reconocimiento del idioma del hablante, la técnica utilizada para estos fines será las máquinas de vectores soporte. Como se mencionó en el apartado anterior, la creciente demanda de aplicaciones a distancia hace cada vez más evidente la necesidad de mejorar e investigar en el campo de los sistemas de reconocimiento automático. Por otro lado, la proliferación de servicios telefónicos multilingües hace que los sistemas de detección de idioma en voz espontánea cumplan una labor muy importante, gracias a este tipo de sistemas las llamadas podrán ser tratadas convenientemente en poco tiempo.

El proyecto comienza en el capítulo 2, con una explicación de los sistemas de reconocimiento automático, sus módulos principales y los posibles modos de operación. También se clasifican e introducen los rasgos biométricos en función de sus características.

El capítulo 3 se centra en los sistemas de reconocimiento automático de locutor, desarrollándose con más profundidad los aspectos relacionados con la voz, y las técnicas empleadas para dicho fin. En este apartado se realiza un recorrido por el estado del arte en este campo que nos ayudará situar las máquinas de vectores soporte dentro del conjunto de técnicas posibles.

Un desarrollo similar al del capítulo 3 se realizará en el capítulo 4, sólo que en este capítulo nos centraremos en el reconocimiento automático del idioma.

Un paso importante para este tipo de sistemas es la extracción de características diferenciadoras de locutores e idiomas de la señal de voz. En el capítulo 5 se explican las distintas parametrizaciones y normalizaciones llevadas a cabo para tal fin. Secciones posteriores de resultados hacen uso de estas parametrizaciones y normalizaciones, de manera que podremos comparar objetivamente sus efectos.

En el capítulo 6 se explican en detalle las máquinas de vectores soporte (SVM). A lo largo del capítulo se introducen las dos variantes de está técnica más importantes para nuestra tarea: las máquinas de vectores soporte basadas en clasificación, ampliamente utilizadas por este tipo de sistemas, y las basadas en regresión, la nueva aproximación propuesta en el proyecto.

Para llevar a cabo las tareas de reconocimiento de forma sistemática, y lo que es más importante, poder comparar los resultados con otros tipos de sistemas, es necesario el establecimiento de ciertos protocolos. Estos protocolos, junto con las bases de datos utilizadas a lo largo de los experimentos y la manera de presentar los resultados, se detallan en el capítulo 7.

El objetivo final del proyecto es estudiar, mejorar, investigar y documentar en el campo del reconocimiento automático de locutor e idioma a través de sistemas basados en máquinas de vectores soporte. Para ello se llevan a cabo distintos experimentos, en el capítulo 8 se muestran los relacionados con el reconocimiento biométrico de locutor y en el capítulo 10 los relacionados con el reconocimiento de idioma. El capítulo 9, al igual que el capítulo 8, contiene experimentos vinculados al reconocimiento de locutor,

pero en este caso no usaremos máquinas de vectores de soporte sino un sistema híbrido compuesto por dichas máquinas y modelos de mezclas de gaussianas.

Todos estos experimentos serán llevados a cabo sobre un ordenador personal, con sistema operativo Linux (distribución Debian), en el que se hará uso de software para la creación y compilación de programas en C++, así como scripts de tipo BASH, biblioteca LibSVM, Matlab 7.0, etc. con los que podremos implementar y probar los algoritmos necesarios para la investigación. A parte de estos medios serán necesarias bases de datos, tanto de locutor como de idioma y bibliografía especializada.

Por último, en el capítulo 11 se discuten los resultados extrayendo conclusiones y posibles vías de trabajo futuro.

2. Sistemas de reconocimiento automático

2.1 Introducción

En una primera aproximación, podemos definir los sistemas de reconocimiento automático como una técnica mediante la cual analizamos ciertas características de un elemento con el fin de clasificarlo o distinguirlo frente otros.

Dentro de los sistemas de reconocimiento encontramos los especializados en reconocimiento biométrico. El reconocimiento biométrico se basa en la medida y el análisis de las características y/o el comportamiento humano con fines de autenticación. Estas características y comportamiento están englobadas dentro de lo que conocemos como rasgos biométricos.

Otro tipo de sistemas de reconocimiento son los sistemas de reconocimiento de idioma. Este tipo de sistemas se basan en la medida y el análisis de las características particulares de cada idioma para distinguirlo frente a otros.

La fuerte expansión de los sistemas de reconocimiento, más concretamente los basados en rasgos biométricos y los de idioma, hace que sea necesario automatizar estos procesos, ya que resulta prácticamente imposible realizar estas tareas a mano para bases de datos de grandes dimensiones.

2.2 Rasgos biométricos

Cualquier característica del ser humano, tanto psicológica como fisiológica puede ser empleada como rasgo biométrico siempre que reúna las condiciones siguientes [Maltoni *et al.*, 2003]:

- **Universalidad:** todo el mundo debe poseer esa característica.
- **Distintividad:** los individuos deberán ser suficientemente diferentes en términos de ese rasgo.
- **Estabilidad:** la característica debe permanecer invariable a lo largo de un periodo de tiempo aceptable.
- **Evaluabilidad:** el rasgo debe poder ser medido cuantitativamente.

Los sistemas de reconocimiento biométrico reales deberán cumplir tres condiciones más:

- **Rendimiento:** hace referencia a la precisión, velocidad y robustez con la que el sistema evalúa ese rasgo.
- **Aceptabilidad:** mide la predisposición de los usuarios a emplear ese rasgo.
- **Seguridad:** los sistemas basados en ese rasgo deben ser suficientemente robustos frente a posibles ataques.

Existen una gran cantidad de rasgos biométricos que cumplen estas propiedades, los sistemas de reconocimiento biométrico se basarán en unos u otros dependiendo de sus necesidades [Jain *et al.,* 2004]. Debemos tener en cuenta que el nivel de cumplimiento de estas propiedades por parte de cada rasgo variará en función de la naturaleza misma del rasgo. En la Figura 1 se enumeran por orden alfabético los rasgos más utilizados en

biometría, la Tabla 1 muestra el nivel de cumplimiento de las propiedades anteriormente mencionadas por parte de la voz.

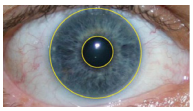
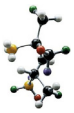
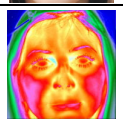
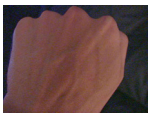
ADN		Iris	
Dinámica de tecleo		Olor	
Escáner de retina		Oreja	
Firma		Rostro	
Forma de caminar		Termograma facial	
Geometría de la mano		Venas de la mano	
Huella dactilar		Voz	

Figura 1. Principales rasgos biométricos

	Universalidad	Distintividad	Estabilidad	Evaluabilidad	Rendimiento	Aceptabilidad	Seguridad
Voz	M	B	B	M	B	A	A

Tabla 1. Propiedades de la voz. A, M y B denotan niveles Alto, Medio y Bajo respectivamente. Tabla adaptada de [Maltoni *et al.*, 2003]

2.3 Funcionamiento sistema de reconocimiento automático

En lo sucesivo nos centraremos en sistemas de reconocimiento automático basados en rasgos biométricos y sistemas de reconocimiento de idioma.

Un sistema de reconocimiento automático es básicamente un reconocedor de patrones que clasificará a los usuarios o idiomas en base a uno o varios rasgos prefijados de antemano. La Figura 2 muestra el esquema de funcionamiento de este tipo de sistemas.

La línea punteada marca la frontera entre la interfaz con el usuario y el propio sistema. El sensor será el encargado de capturar las características del usuario o idioma, por otra parte el sistema puede solicitar la identificación del usuario o idioma, este proceso dependerá de si el sistema trabaja en verificación o en identificación (términos que se explicarán en el siguiente apartado). Los módulos marcados con líneas discontinuas hacen referencia a etapas opcionales que podrán ser obviadas durante el diseño.

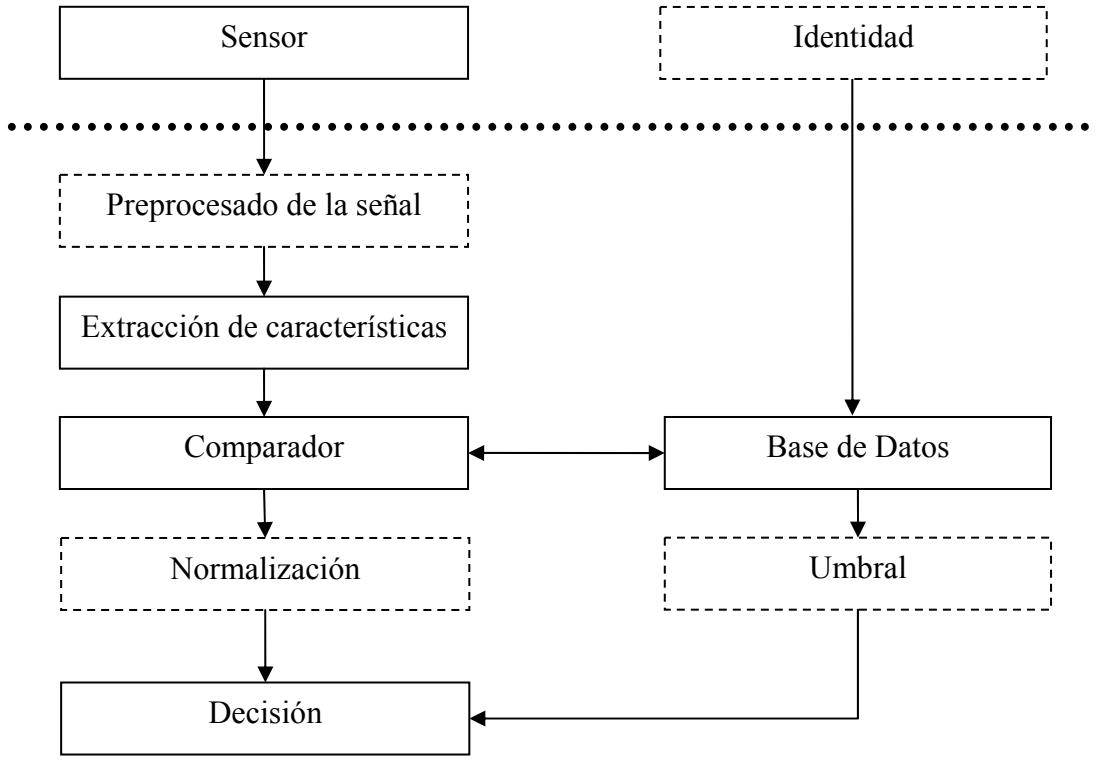


Figura 2. Esquema de funcionamiento de un sistema de reconocimiento

Dependiendo del tipo de aplicación los sistemas podrán trabajar *on-line* u *off-line*. Los sistemas *on-line*, como por ejemplo los de acceso restringido basados en rasgos biométricos, necesitarán generar la decisión de una manera rápida, casi inmediata, por lo que serán sistemas completamente automáticos. Por el contrario, los sistemas *off-line*, como por ejemplo los usados en ciencia forense, pueden permitirse emplear un cierto tiempo en el reconocimiento. Gracias a esta demora será posible la intervención humana durante el proceso, lo que mejorará los resultados. También será viable que el sistema devuelva una lista de posibles candidatos, la cual podrá ser manualmente examinada por un experto forense.

Dentro del ámbito de los sistemas de reconocimiento biométricos, éstos pueden tener dos finalidades: el reconocimiento positivo y el negativo. El reconocimiento positivo es aquél que busca comprobar que un usuario es realmente quien dice ser. En el caso de reconocimiento negativo, se trata de lograr determinar que un usuario no es quien afirma ser. Cabe destacar que la identificación negativa sólo puede ser realizada mediante rasgos biométricos, y no mediante métodos clásicos como contraseñas o llaves.

2.4 Modos de operación en un sistema de reconocimiento automático

Distinguiremos tres modos de operación, en dos de ellos, verificación e identificación, se puede considerar que el sistema está en funcionamiento. El modo registro, por el contrario, es una fase previa y común a estos dos modos de funcionamiento.

- **Modo registro**

En este modo los usuarios o idiomas son dados de alta en el sistema, para ello se extrae el rasgo o característica correspondiente y se almacena junto con la información del usuario o idioma (identidad). En los sistemas de reconocimiento biométrico, dependiendo de la aplicación, la información de usuario será guardada en la base de datos del sistema o en otro tipo de dispositivos (tarjetas inteligentes, magnéticas, etc.).

Una vez creada la base de datos el sistema podrá entrar en funcionamiento en uno de estos dos modos:

- **Modo Verificación**

Este modo de funcionamiento es utilizado por los sistemas de reconocimiento biométrico de locutor para comprobar la identidad de un usuario. Para ello debe llevarse a cabo una comparación "uno a uno", siendo necesarias dos aportaciones por parte del usuario. Por un lado se necesitarán sus características, por otro lado el usuario deberá indicar al sistema su identidad. De esta forma el sistema buscará la información de ese usuario concreto y podrá validarla con la suministrada.

La salida de este modo de operación suele ser verdadero o falso, dependiendo de si el usuario es o no el que le ha dicho al sistema que es.

- **Modo Identificación**

Tanto los sistemas de reconocimiento biométrico de locutor como los de reconocimiento de idioma operan en este modo. En el modo de identificación la comparación llevada a cabo por el sistema es "uno a varios", al contrario que sucedía en el modo verificación el sistema sólo necesita conocer las características, no la identidad. El sistema tratará de decidir si el usuario o idioma está o no en la base de datos, pudiendo darse la posibilidad de que no se encuentre en esta. Hay que tener en cuenta que este modo de operación tiene un coste computacional muy elevado, será proporcional al número de entradas que contenga la base de datos.

La salida del modo identificación será el usuario o idioma al que pertenecen los rasgos introducidos, o un mensaje de no encontrado.

En la Figura 3 se muestra de manera esquemática los modos de funcionamiento de un sistema de reconocimiento automático basado en características extraídas de la voz.

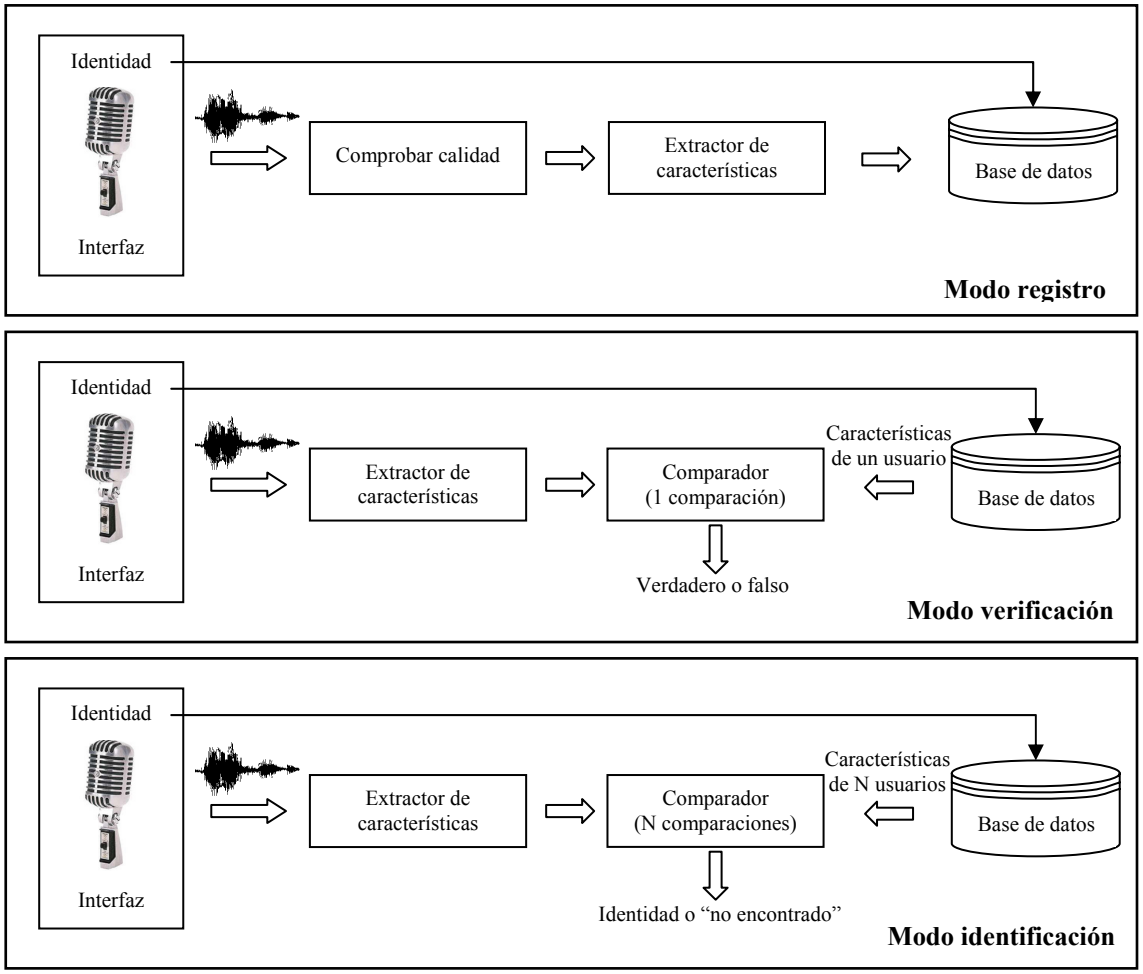


Figura 3. Modos de funcionamiento de un sistema automático de reconocimiento. Figura adaptada de [Maltoni *et al.*, 2003]

3. Estado del arte en reconocimiento biométrico de locutor

3.1 Identidad de la persona en la señal de voz

La comunicación mediante el habla es la forma más habitual de transmitir información entre personas. Al hablar hacemos uso de complejos mecanismos aprendidos durante la infancia, desde la construcción de mensajes lingüísticos correctos, hasta la expresión de dichos mensajes a través de nuestros órganos articulatorios en una señal capaz de transmitirse hasta el oyente, como será la señal de voz.

Los mecanismos mediante los cuales generamos esta señal dependen de múltiples variables a diferentes niveles, incluyendo desde factores sociolingüísticos (nivel de educación, contexto lingüístico y diferencias dialectales) hasta cuestiones fisiológicas (longitud y forma del tracto vocal, configuración de los órganos articulatorios). El resultado será una señal, la voz, que contiene el mensaje lingüístico que queremos transmitir además de mucha otra información entre la que se encontrará la identidad del locutor.

Toda esta información estará codificada en la señal de voz mediante la combinación de características temporales y espectrales. El objetivo de los sistemas de reconocimiento biométrico basados en voz será el de centrarse en las características que individualizan al hablante. Por el tipo de información de la que hacen uso podemos distinguir dos grandes grupos de reconocedores, los de alto nivel y los de bajo nivel.

Los reconocedores de locutor de alto nivel se centrarán en la información específica del individuo procedente de la fase de generación del mensaje en el cerebro, los de bajo nivel, por el contrario, se centrarán en la información característica procedente de la fase de producción de voz.

3.1.1 Niveles de identidad

Existen multitud de estudios en los que se muestran los mecanismos mediante los que las personas reconocen la identidad de los distintos locutores. Todos ellos parecen apuntar a que la clave está en la combinación de los distintos niveles de información, así como en el peso que se le da a cada uno de ellos. Los sistemas de reconocimiento automático de locutor trataran de asemejarse al comportamiento humano, combinando las distintas fuentes de información de la mejor manera posible [Reynolds *et al.,* 2003].

Las particularidades de la voz pueden agruparse en cuatro grandes grupos según el nivel en el que se den, a continuación se presentan estos niveles desde el más alto al más bajo: nivel lingüístico, nivel fonético, nivel prosódico y nivel acústico.

Las características idiolectales [Doddington, 2001] forman parte del nivel más alto, el **nivel lingüístico**, en el que se pueden clasificar las particularidades de la voz. Estas características describen la forma en la que el locutor hace uso del sistema lingüístico, y se verán influenciadas entre otros aspectos por la educación, el origen y las condiciones sociológicas del hablante. Basándonos en estas características podremos tener sistemas que modelen locutores por la frecuencia de uso de palabras o secuencias de palabras.

El **nivel fonético** está compuesto por las características fonotácticas [Carr, 1999], es decir, los fonemas y secuencias de fonemas. Está demostrado que el uso de estos fonemas conforma un patrón único para cada locutor, por lo tanto podremos tener reconocedores basados en este tipo de características.

El tercer grupo, **nivel prosódico**, está compuesto por la prosodia. La prosodia se define como una combinación de energía, duración y tono de los fonemas, es la principal responsable de dotar a la voz de sentido y naturalidad. Aunque la prosodia consta de elementos comunes para todos los hablantes, por ejemplo, ayuda a distinguir el tipo de mensaje (declarativo, interrogativo, imperativo), cada locutor usará dichos elementos prosódicos de una manera distinta. Dos de los elementos prosódicos más representativos de la persona son el tono y la energía, los reconocedores automáticos de locutor basados en este nivel tratarán de extraer esta información de manera rápida y automática, para posteriormente usarla en el reconocimiento.

En el nivel más bajo, conocido como **nivel acústico**, se encuentran las características espectrales a corto plazo de la señal de voz. Estas características están directamente relacionadas con las acciones articulatorias de cada individuo, la forma en la que se produce cada sonido, y la configuración fisiológica del mecanismo de producción de voz. La información espectral trata de extraer las particularidades del tracto vocal de cada locutor así como su dinámica de articulación. Esta información puede dividirse a su vez en dos grupos, el estático y el dinámico. La información estática es la extraída del análisis de cada trama individual, la información dinámica, por el contrario, se extrae del análisis de las tramas de forma conjunta, de esa manera es posible recoger los pasos de unas posiciones de articulación a otras.

Durante los últimos 20 años, los reconocedores automáticos de locutor se han basado principalmente en la información de más bajo nivel, las características espectrales [Reynolds *et al.,* 2000; Wan y Campbell, 2000; Campbell, 2002]. Como ya se ha mencionado la tecnología actual esta tratando alcanzar los niveles de precisión humana, para ello deberá ser capaz de procesar la mayor cantidad de información posible de cada uno de los niveles e integrarla de una manera inteligente.

3.2 Reconocimiento de locutor multinivel y fusión

En el apartado 3.1 se presentó la forma en la que la identidad de la persona estaba presente en la señal de voz. Una de las conclusiones importantes de dicho apartado fue la importancia de trabajar con información procedente del mayor número de niveles posibles.

El objetivo de la fusión de sistemas es conseguir un sistema global más robusto y con mejores prestaciones que los sistemas individuales por si solos, el resultado será mejor cuanto más dispar sea la información usada por los subsistemas individuales [Reynolds *et al.,* 2003]. La información de bajo nivel es más fácil de extraer y modelar que la de alto nivel, la desventaja es que este tipo de información presenta una alta sensibilidad a fuentes de variabilidad (canal, paso del tiempo, etc.). La información de alto nivel, por el contrario, presenta un comportamiento más insensible a estas fuentes de variabilidad, pero la extracción de este tipo de información resulta más compleja, necesitando longitudes de entrenamiento grandes, mayores de 10 minutos.

Los sistemas combinados multinivel tratarán de sacar el máximo partido a la complementariedad y seudo-ortogonalidad de los distintos tipos de informaciones. Para ello desde las fusiones más sencillas por regla (suma, producto, etc.), pasando por mecanismos más sofisticados (fusión bayesiana, redes neuronales, máquinas de vectores soporte [Fierrez-Aguilar *et al.,* 2003]), hasta llegar a los métodos más novedosos (fusión adaptada al usuario [Fierrez-Aguilar *et al.,* 2005], fusión mediante regresión logística), trataran de combinar de la mejor manera posible las informaciones contenidas en las puntuaciones modelo-usuario de los distintos sistemas.

En la sección 9.4 y en la sección 10.9 de los experimentos se mostrarán los resultados de combinar dos sistemas mediante la fusión por la regla de la suma.

3.3 Técnicas empleadas

La principal fuente de información codificada en la señal de voz es indudablemente el contenido lingüístico. Por este motivo no es sorprendente que dependiendo de cómo usemos o controles este contenido, podamos distinguir dos tipos diferentes de tecnologías de reconocimiento de locutor.

En primer lugar mencionaremos las tecnologías dependientes de texto *(text-dependent technologies)*. Se caracterizan porque el sistema conoce de antemano lo que va a decir el usuario, una clave específica, pudiendo ser una frase (ábrete sésamo) o una secuencia de números (1, 2, 3, 4) [Wagner *et al.,* 2006].

En segundo lugar presentaremos las tecnologías independientes de texto *(text-independent technologies)*, es el caso completamente opuesto al anterior. Suponen un reto mayor para la comunidad científica ya que carecen del contenido lingüístico, la principal fuente de información codificada en la voz, por lo que los sistemas deberán ser más ambiciosos. Este campo ha sido el más estudiado durante las últimas dos décadas, produciéndose avances muy significativos.

Tecnologías dependientes de texto

Este tipo de tecnologías toma su mayor relevancia en aplicaciones de autenticación de identidades, donde sea requerida la colaboración de los usuarios. La aplicación ejemplo típica de estos sistemas sería la banca telefónica.

Desde el punto de vista de la aplicación podemos clasificar los sistemas dependientes de texto en dos grandes grupos, texto fijo *(fixed-text)* y texto variable *(variable-text)*. La diferencia radica en el contenido léxico utilizado durante las fases de entrenamiento y reconocimiento, mientras que en los sistemas de texto fijo es siempre idéntico, en los sistemas de texto variable cambia, pudiendo incluso hacerlo en cada una de las veces que el usuario se enfrente al sistema. Este hecho hace que los sistemas de texto variable sean más flexibles y por tanto más robustos frente a ataques. Cabe la posibilidad de generar una clave aleatoria para cada nueva entrada del usuario, de esta forma los intentos de engañar al sistema mediante la grabación y reproducción de la clave se verían frustrados.

El reconocimiento de locutores se reduce a comparar las características de la locución de entrenamiento con las que genere el usuario en cada intento de acceso al sistema. Para ello podemos usar varias aproximaciones, en este trabajo explicaremos dos de

ellas, métodos basados en ajuste de plantillas, como el alineamiento temporal dinámico *(DTW o Dynamic Time Warping)* y métodos estadísticos, como modelos ocultos de cadenas de Markov *(HMMs o Hidden Markov Models)*.

- **DTW**

El objetivo de este tipo de algoritmos es obtener una medida de similitud entre dos locuciones, la almacenada en la base de datos y la locución de prueba. Esta medida de similitud se consigue mediante la comparación de ambas locuciones, el problema surge en dicha comparación, ya que no puede ser realizada de una manera directa.

Entre dos realizaciones de una misma frase, por parte del mismo u otro locutor, existirá un cierto desalineamiento. Este desalineamiento es debido a la variabilidad misma de la realización del acto de voz, siendo casi imposible generar dos realizaciones completamente alineadas. Ilustraremos con un ejemplo el problema mencionado, una misma frase pronunciada con distinta velocidad puede parecer diferente cuando en realidad es misma.

El algoritmo DTW se basa en técnicas de programación dinámica, mediante estas técnicas será capaz de encontrar el camino de alineamiento con distancia total mínima. Esta tarea es equivalente a encontrar el camino óptimo a través de un diagrama de trellis. Para ello se divide la locución completa en tramos de corta duración, el camino total mínimo estará compuesto por los caminos óptimos entre dicho tramos. La medida de similitud utilizada por el algoritmo en la comparación será la longitud total del camino óptimo.

Este tipo de técnicas comenzaron a usarse en el reconocimiento de palabras aisladas, comparando la palabra con las existentes en la base de datos [Rabiner *et al.,* 1978]. Posteriormente se vio su utilidad en la comparación de claves en sistemas de seguridad basados en voz, control de accesos.

La principal limitación de este tipo de sistemas es el hecho de que sean dependientes de texto, el sistema debe tener en su base de datos al menos una realización acústica de la misma clave que vaya a utilizar el usuario. Este hecho hizo que los sistemas fueran muy poco flexibles, llevando a la técnica DTW a caer en desuso.

- **HMM**

Un modelo oculto de Markov es un modelo estadístico en el que se asume que el sistema a modelar es un proceso de Markov de parámetros desconocidos. Definimos este proceso o cadena como una serie de eventos, la probabilidad de que ocurra uno de estos eventos depende del evento inmediatamente anterior. Basándonos en este hecho podríamos decir que la cadena tiene memoria, los eventos futuros dependerán de eventos pasados. El objetivo de este tipo de sistemas será determinar los parámetros desconocidos.

Los HMMs aplicados a reconocimiento de locutores toman la señal de voz como salida de una secuencia de estados de Markov. Los estados o eventos de la cadena de Markov serán ocultos pero observables de manera indirecta a partir de las secuencias de vectores espectrales producidos, siendo los parámetros característicos del HMM las

probabilidades de transición entre estados y las probabilidades de observación de los vectores espectrales en cada estado.

Las transiciones permitidas entre estados será determinante a la hora de elegir la forma en que realizaremos el reconocimiento de locutor. En el caso que nos ocupa, tecnologías dependientes de texto, tendremos arquitecturas de izquierda a derecha. Aunque los HMM se han incluido en la sección de las tecnologías dependientes de texto, es posible utilizarlos en sistemas independientes de texto, tan sólo habrá que variar las transiciones permitidas, dando lugar a estructuras totalmente conectadas.

Para más información sobre esta técnica puede consultarse [Rabiner, 1989].

Tecnologías independientes de texto

Los sistemas independientes de texto han sido claros dominantes durante las últimas décadas, más concretamente los basados en características espectrales a corto plazo. En el 2000 empezaron a desarrollarse sistemas de alto nivel con buenas prestaciones, pero por el momento no han superado los resultados de sistemas basados en características espectrales.

El estado del arte en sistemas basados en características espectrales, también conocidos como sistemas acústicos, lo componen los modelos de mezclas de gaussianas *(GMM o Gaussian Mixture Model)*, las máquinas de vectores soporte *(SVM o Support Vector Machines)* y sistemas híbridos GMM-SVM *(SuperVectors)*. Este proyecto se centrará en sistemas acústicos, más concretamente en las máquinas de vectores soporte.

A parte de estos sistemas acústicos, existen otro tipo de sistemas que explotan las particularidades de la voz a más alto nivel, en el nivel fonético y prosódico.

- **GMM**

Esta técnica se basa en el modelado de los parámetros de entrada al sistema mediante modelos de mezcla de gaussianas multidimensionales [Reynolds *et al.*, 2000]. Como es habitual, el sistema se divide en dos fases, entrenamiento y test. En la fase de entrenamiento se obtienen los parámetros del modelo que mejor se ajustan a cada locutor, es decir, entrenaremos un modelo por cada locutor que constará de sus parámetros más representativos. En la fase de test se decidirá si las locuciones de entrada se corresponden con los modelos mediante el cómputo de una medida de similitud entre ambos. A continuación presentaremos el problema de una manera más formal:

Sea $O = \{o_1, o_2, ..., o_N\}$ la secuencia de observaciones de un vector de dimensión *d* extraído de un segmento de voz, y λ_t el modelo generado usando la información del usuario. Definiremos la medida de similitud entre ambos, puntación, como:

$$s(O, \lambda_t)$$

En el caso del modelado estadístico, las distribuciones de las características de la voz del usuario pueden representarse por un modelo de mezclas de gaussianas de *d* dimensiones:

$$f(x \mid \lambda_t) = \sum_{i=1}^{M} w_i f_i(x)$$

$$f_i(x) = \frac{1}{(2\pi)^{d/2} \left|\sum i\right|^{1/2}} \exp\left\{ -\frac{1}{2}(x-\mu_i)^T \left(\sum i\right)^{-1} (x-\mu_i) \right\}$$

En este caso, la puntuación puede calcularse a través de la siguiente fórmula de similitud:

$$s(O, \lambda_t) = \log f(O \mid \lambda_t) - \log(O \mid \lambda_{UBM})$$

Donde $f(X \mid \lambda_t)$ y $f(X \mid \lambda_{UBM})$ son las funciones densidad de probabilidad para el supuesto modelo y un modelo universal *(UBM o Universal Background Model)*, los cuales son modelados como mezclas de gaussianas, como definimos anteriormente. El modelo UBM es entrenado mediante técnicas de máxima similitud *(Maximum Likelihood)*. Una vez generado el modelo universal, los modelos de usuario serán derivados de él mediante la adaptación desde ese modelo universal.

La principal ventaja de este sistema es su relativa sencillez, su principal limitación es que únicamente modela información acústica a muy corto plazo.

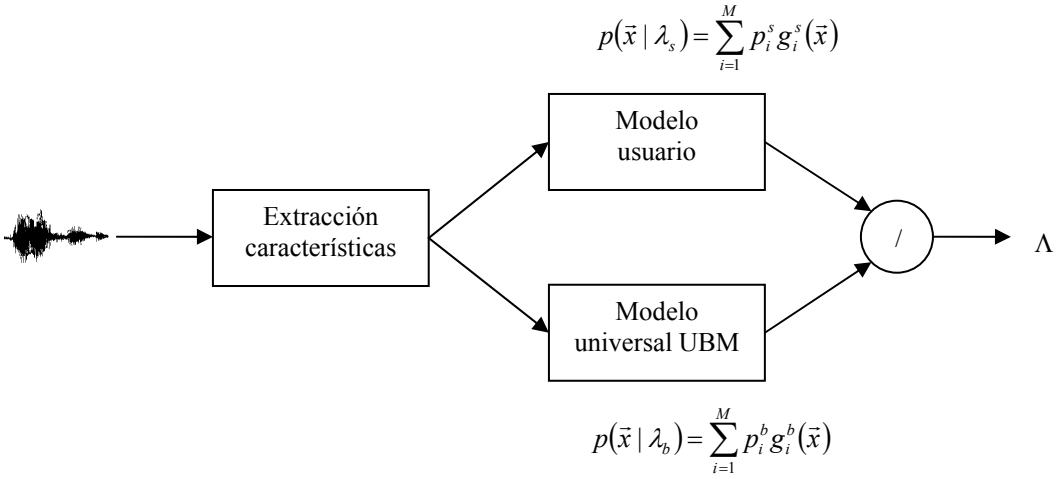


Figura 4. Esquema sistema GMM, figura adaptada de [Reynolds *et al.*, 2000]

- **SVM**

Las máquinas de vectores soporte conforman una técnica de aprendizaje discriminativo, su principal objetivo es establecer una frontera de separación entre clases [Vapnik, 1995]. Debido a su flexibilidad y buen comportamiento en una gran variedad de problemas, comenzaron a usarse en técnicas de reconocimiento de patrones, siendo una de las técnicas más relevantes en el estado del arte durante los últimos años.

El presente trabajo esta basado en estas técnicas. Sus innumerables ventajas y buen comportamiento fueron factores decisivos en la elección de las máquinas de vectores soporte para la tarea de reconocimiento automático de locutor [Reynolds, 2003a; Gonzalez-Rodriguez, 2007] e idioma. En la sección 6 se explicará en detalle dicha técnica y la formulación matemática subyacente.

- **Sistema híbrido GMM-SVM**

Esta técnica, también conocida como "SuperVectors" [Campbell *et al.*, 2006a; Krause y Gazit, 2006] fue propuesta inicialmente por Ran Gazit, y en estos últimos años ha demostrado sobradamente su capacidad dentro de la tarea de reconocimiento de patrones. Los SuperVectors son una técnica híbrida, aprovecha las propiedades de modelado generativo de los sistemas GMM, así como las de modelado discriminativo de los sistemas SVM. Este modelado híbrido supone una mejora, incluso sobre la fusión de ambos sistemas a nivel de puntuaciones.

La idea que reside detrás del sistema de SuperVectors es la siguiente, mediante un sistema SVM se modelan las desviaciones de los vectores de medias de los modelos, estos vectores de medias se obtienen del modelado GMM de los diferentes locutores.

Seguidamente se describe el procedimiento empleado por el sistema:

1. Se entrena un modelo de GMM para cada locución que intervenga en el experimento, tanto para locuciones de entrenamiento, como de desarrollo y test.

2. De cada modelo GMM entrenado extraemos el vector de medias de cada una de sus gaussianas, ponderamos dicho vector por su peso y covarianza para posteriormente agrupar todos los vectores de medias ponderados en un único vector. Dicho "supervector" es de dimensión m*d, donde "m" representa el número de mezclas del modelo de GMM y "d" es el número de dimensiones del vector de características.
   Por tanto, de cada locución obtenemos un único supervector, que representa un único punto en lo que se conoce como "el dominio de los modelos de GMMs".

3. La discriminación entre supervectores correspondientes a locutores impostores, *NonTarget*, y locutores usuarios, *Target*, la llevaremos a cabo mediante un sistema SVM, que es básicamente un clasificador binario.

4. La puntuación de un enfrentamiento entre un modelo y un fichero de test se obtendrá de la misma forma que en un SVM clásico. Enfrentando el modelo del locutor correspondiente con el supervector de la locución de test.

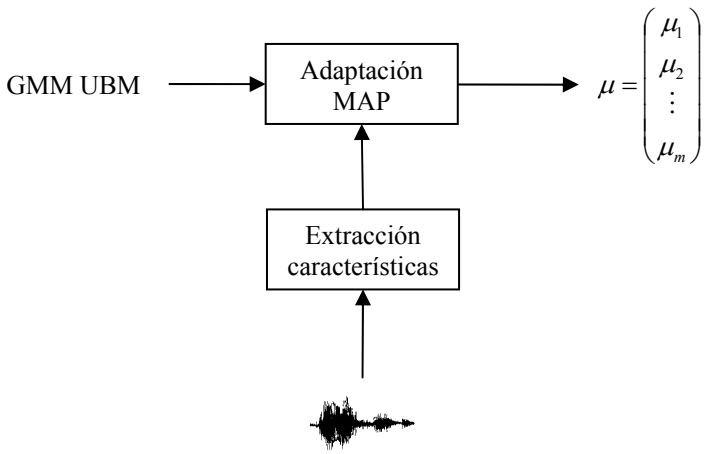


Figura 5. Concepto de supervector GMM. Figura adaptada de [Campbell *et al.*, 2006a]

- **Sistemas de alto nivel: fonéticos y prosódicos.**

Tanto los sistemas fonéticos como los prosódicos están compuestos por dos bloques. En el caso de los sistemas fonéticos el primer bloque es un decodificador fonético, su función es transformar la voz en una secuencia de etiquetas fonéticas. El primer bloque de los sistemas prosódicos hará algo similar al de los sistemas fonéticos, sólo que en este caso analizará la prosodia representándola como una secuencia de etiquetas. El segundo bloque de ambos sistemas es prácticamente el mismo, se conoce como la etapa de modelado estadístico de lenguaje, en ella se modela la frecuenta de fonemas y sonidos propios de cada locutor.

Como se comentó anteriormente, el reconocimiento de locutor está dominado por los sistemas acústicos, por lo que no entraremos en detalle en este tipo de técnicas. En la sección siguiente se explicarán en más detalle.

4. Estado del arte en reconocimiento de idioma

4.1 Información del idioma en la señal de voz

Como ya vimos en la sección 3.1, la señal de voz contiene una gran cantidad de información. Esta información la clasificábamos en dos grandes grupos, información de bajo nivel e información de alto nivel. En un análisis más fino dividíamos cada uno de estos grandes grupos en dos, dando como resultado una clasificación de las particularidades de la señal de voz en cuatro niveles.

Al igual que sucedía con los aspectos relacionados con la identidad del locutor, las particularidades específicas de cada idioma se encuentran esparcidas por todos los niveles. Si bien es verdad que los niveles superiores parecen ser los más relevantes en la tarea de reconocimiento de idioma.

Las particularidades articulatorias y la configuración fisiológica independiente de cada idioma nos harán tener diferencias a nivel acústico. También tendremos diferencias a nivel prosódico ya que cada idioma presentará unos fonemas representativos, es decir, fonemas con una energía, duración y tono característicos [Obuchi y Sato, 2005]. Estos fonemas en sí y su combinación nos darán las diferencias en el nivel prosódico. Ya en el nivel más alto podremos observar las diferentes palabras y estructuras gramaticales de cada idioma.

Las tendencias actuales a fusionar sistemas de reconocimiento de locutor, explicadas en la sección 3.2, son perfectamente aplicables a idioma. Sigue siendo válido el hecho de que cuantos más subsistemas se fusionen y más dispares sean entre sí, mejores resultados se obtendrán.

4.2 Técnicas empleadas

Al igual que sucedía con el reconocimiento biométrico de locutor, la señal de voz es la portadora de la información relativa al idioma. Por este motivo no es de extrañar que las técnicas aplicadas al reconocimiento de locutor sean extrapolables al reconocimiento de idioma.

Las técnicas explicadas en la sección 3.3 pueden ser empleadas en la nueva tarea de reconocimiento de idioma. Como se resaltó en el apartado anterior, la información acerca del idioma del hablante está más presente en los niveles superiores, por tanto, las técnicas basadas en dichos niveles de información obtendrán mejor rendimiento en la tarea de reconocimiento de idioma.

De entre las técnicas basadas en los niveles superiores de información es preciso destacar los sistemas de reconocimiento fonético. Estos sistemas son: PRLM *(Phone Recognition followed by Language Modelling)*, PPRLM *(Parallel PRLM)* y PPR *(Parallel Phone Recognition)*.

- **Sistemas de reconocimiento fonético: PRLM, PPRLM y PPR**

Las técnicas presentadas en este apartado se asientan en la combinación del reconocimiento fonético, basado en modelos ocultos de Markov, y el modelado estadístico del lenguaje, conjunto de fonemas y frecuencias de aparición.

La combinación de estas técnicas permite hacer uso tanto de las particularidades acústicas, como de las particularidades fonéticas del lenguaje. La forma en la que combinemos estas técnicas dará lugar a un tipo u otro de sistema, siendo PRLM, PPRLM y PPR los más comunes.

- PRLM es una técnica en la que se usa un modelo estadístico de lenguaje, habitualmente un n-grama, de las secuencias de fonemas probablemente reconocidas por un único reconocedor fonético, pudiendo ser este del mismo idioma o distinto, para reconocer al idioma. La identificación del idioma consiste en determinar el modelo de lenguaje que habría generado la secuencia de fonemas reconocida con mayor probabilidad, para ello debemos aplicar el reconocedor de fonemas a la locución. La Figura 6 muestra el esquema de funcionamiento.

- PPRLM es una técnica extendida de la anterior. El sistema dispondrá de varios reconocedores fonéticos, correspondientes a distintos idiomas, de cada uno de ellos obtendremos una probabilidad o puntuación. La decisión final se obtendrá combinando todas las puntuaciones obtenidas.

- PPR consiste en aplicar la locución a un reconocedor que combina HMMs fonéticos y modelos de lenguaje por cada locutor a reconocer. De esta forma se combina el reconocimiento fonético y el modelo de lenguaje, a diferencia de lo que se hacía en las técnicas anteriores donde se aplicaban secuencialmente existiendo un desacoplamiento total.

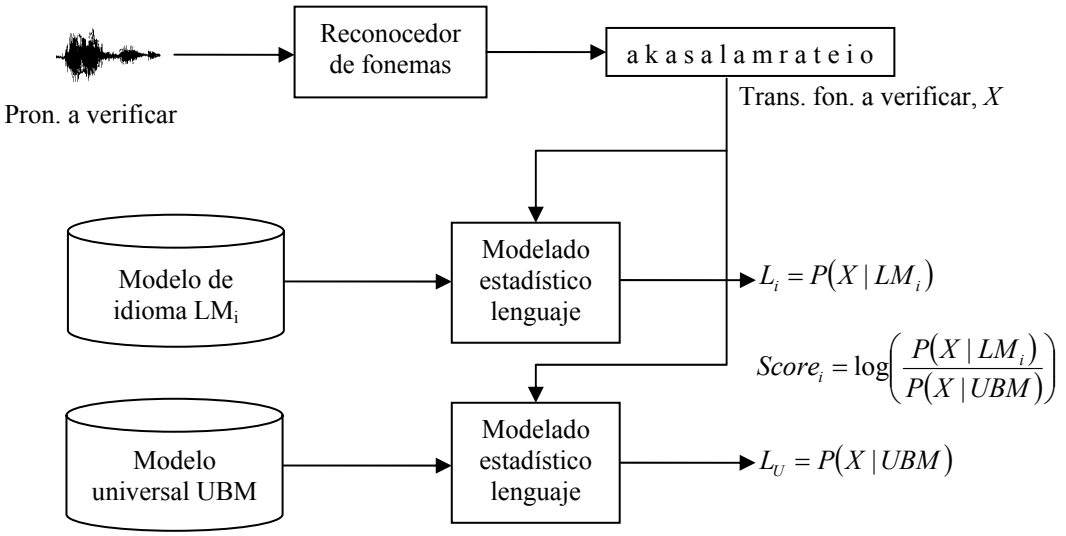


Figura 6. Esquema PRLM de verificación de un idioma

PPRLM es la técnica más popular de las tres explicadas anteriormente, desde sus comienzos hasta la fecha ha demostrado un rendimiento elevado [Zissman, 1996].

PPRLM ha sido ampliamente utilizada desde la evaluación NIST LRE 1996, en dicha evaluación PPRLM superaba ampliamente a cualquier otra técnica. En las siguientes evaluaciones de NIST, 2003 y 2005, los sistemas basados en GMMs y SVMs se acercaron bastante a los resultados obtenidos por PPRLM. Esta mejora protagonizada por los sistemas acústicos se debió en gran medida a la cantidad de avances llevados a cabo en diversos campos, entre ellos destacaremos la implementación de un nuevo tipo de parametrización conocida como SDC *(Shifted Delta Cepstral)* [Torres-Carrasquillo, 2002]. Esta parametrización amplía la ventana de tiempo donde se calculan los parámetros, haciendo que el sistema trabaje con unos vectores de características cuya información temporal es mucho mayor. En la sección 5 se explica esta nueva técnica de parametrización en detalle junto con más aspectos relacionados con la extracción de parámetros.

Los progresos mencionados en sistemas como GMM, SVM o SuperVector han obligado a la comunidad científica a desarrollar sensibles mejoras para mantener a PPRLM en el estado del arte. Algunas de estas mejoras son la extracción de más información a nivel fonético *(lattices)* [Hatch *et al.,* 2005], o el empleo de SVM como criterio de decisión [Campbell *et al.,* 2004a], en lugar de comparar probabilidades.

La próxima evaluación mundial de reconocimiento de idioma, NIST LRE 2007, mostrará el comportamiento de todos estos tipos de sistemas con sus nuevos avances, con lo que podremos confirmar si PPRLM sigue siendo la técnica dominante.

5. Extracción de características en locutor e idioma

La extracción de características es el paso previo a cualquier sistema de reconocimiento automático. En primer lugar se captará la señal que deseemos utilizar mediante un sensor, en nuestro caso al tratarse de la señal de voz será un micrófono. El proceso de extracción de parámetros se realiza a partir de la representación discreta de dicha señal, por tanto deberemos digitalizar la señal analógica.

La extracción de parámetros está basada habitualmente en el análisis a corto plazo de la señal de voz, para ello una de las técnicas más habituales en reconocimiento automático de locutor es MFCC *(Mel-Frequency Cepstral Coefficients)*, en reconocimiento de idioma haremos uso de esta técnica y de otra conocida como SDC *(Shifted Delta Cepstral)*, las cuales pasaremos a detallar más adelante.

Sobre estas técnicas básicas pueden llevarse a cabo mejoras con el fin de paliar las distorsiones sufridas por la señal de voz y mejorar el rendimiento de los sistemas, estas mejoras son conocidas como normalizaciones y en este trabajo se utilizarán cuatro de las más importantes: normalización por la media cepstral *(CMN o Central Mean Normalization)*, Feature Mapping, Feature Warping y RASTA filtering.

MFCC *(Mel-Frequency Cepstral Coefficients)*

Los coeficientes MFCC se extraen a partir de la representación de la señal de voz en el dominio espectral [Deller *et al.*, 1999]. Diversas investigaciones llevadas a cabo hasta la fecha, han demostrado que los coeficientes obtenidos del dominio espectral representan más fielmente las características de la voz que los obtenidos del dominio temporal. Esta peculiaridad es debida a que las personas utilizan este mismo dominio para distinguir sonidos, por tanto, cabe esperar que un sistema que trabaje con características del dominio espectral se acerque más al comportamiento humano.

El proceso seguido por la señal de voz hasta obtener los coeficientes MFCC es el siguiente. En primer lugar se realiza un enventanado de la señal de voz, típicamente divide la locución en ventanas de 20ms con solapamiento del 50% (10ms) a través de ventanas de tipo hamming. Tras pasar al dominio espectral se filtra la señal resultante mediante un banco de filtros de diferentes frecuencias y amplitudes, el objetivo de este filtrado es dar más resolución a las bajas frecuencias, como sucede en el sistema auditivo humano. De la salida de cada filtro se calcula una energía promedio, obteniendo de este modo una señal con tantos valores de energía como filtros. Al pasar esta señal a través de una transformada DCT *(Discrete Cosine Transform)*, se obtienen los coeficientes MFCC ortogonales entre sí, típicamente de 13 a 20. La Figura 7 esquematiza el proceso mostrado hasta ahora.

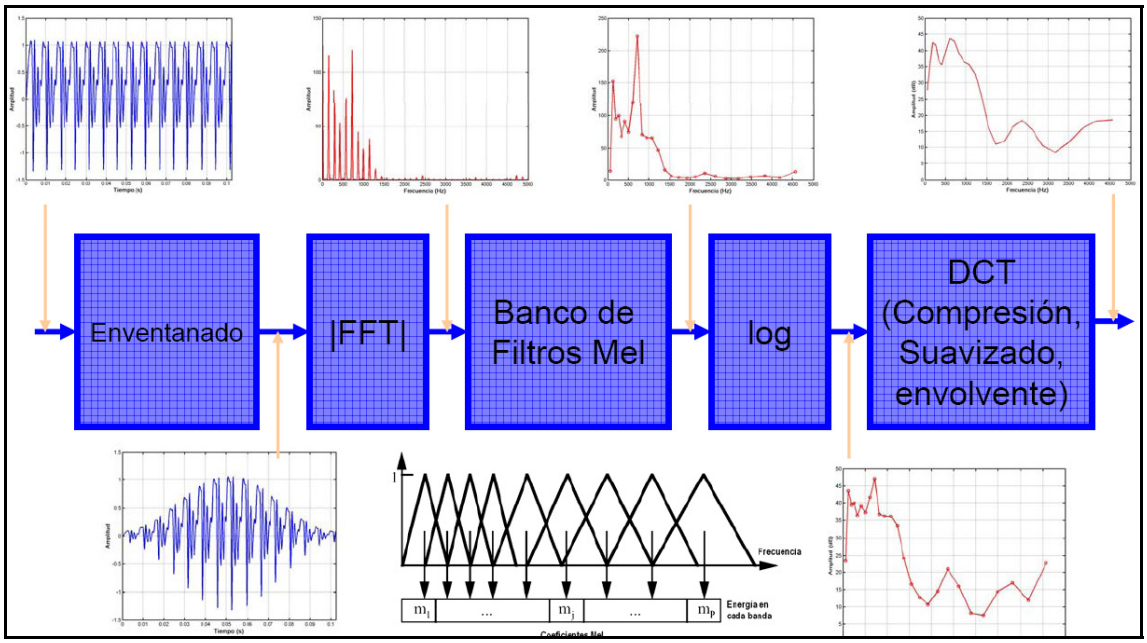


Figura 7. Extracción de coeficientes MFCC

A parte de estos coeficientes se suelen utilizar otros conocidos como deltas o coeficientes de primera y segunda derivada. Estos coeficientes tratan de representar la información de coarticulación entre fonemas, por ello miden velocidades y aceleraciones alrededor del instante de tiempo dado. El vector resultante seguirá la notación MFCC + Delta + Delta-Delta, en el caso de que se usen todos los coeficientes mencionados. La Figura 8 muestra un ejemplo del cálculo de los coeficientes delta.

		t_0	t_1	t_2	…	t_{n-1}	t_n	t_{n+1}	…
7 coeficientes MFCC	0					C_0	C_0	C_0	
	1					C_1	C_1	C_1	
	2					C_2	C_2	C_2	
	3					C_3	C_3	C_3	
	4					C_4	C_4	C_4	
	5					C_5	C_5	C_5	
	6					C_6	C_6	C_6	
7 coef. delta	0				…		$C_0(t_{n+1}) - C_0(t_{n-1})$		…
	1						$C_1(t_{n+1}) - C_1(t_{n-1})$		
	…								
	6				…		$C_6(t_{n+1}) - C_6(t_{n-1})$		…

Figura 8. Ejemplificación del cálculo de unos posibles coeficientes delta sobre la trama t_n

SDC *(Shifted Delta Cepstral)*

La parametrización SDC [Torres-Carrasquillo, 2002] podemos tratarla como una parametrización derivada de la MFCC, es una extensión de las deltas calculadas en MFCC. En lugar de medir la velocidad o la aceleración de los coeficientes MFCC de manera estándar, los coeficientes SDC tratan de hacerlo de una manera más genérica, de esta forma representaremos la información de cada ventana en función de las adyacentes.

Los coeficientes SDC vienen especificados por cuatro parámetros, N-d-P-k, donde: **N** es el número de coeficientes cepstrales, **d** representa el desplazamiento en tiempo para el cálculo de las deltas (hacia delante y hacia atrás), **P** es el desplazamiento entre bloques consecutivo, por último, **k** es el número bloques que serán concatenados para formar el vector final.

Para cada instante t los coeficientes SDC se calculan siguiendo la siguiente fórmula:

$$\Delta C_n(t,i) = C_n(t + iP + d) - C_n(t + iP - d)$$
$$n = 0,....,N-1$$
$$i = 0,....,k-1$$

En la Figura 9 podemos ver un pequeño ejemplo del cálculo de la parametrización SDC, en este caso la configuración elegida será 3-2-1-3. Esto quiere decir que tendremos 3 coeficientes por bloque (N=3), desplazamientos de dos tramas hacia delante y hacia atrás (d=2), un desplazamiento entre bloques consecutivos (P=1) y tres bloques (k=3). El vector resultante estará compuesto por 9 parámetros (N*k).

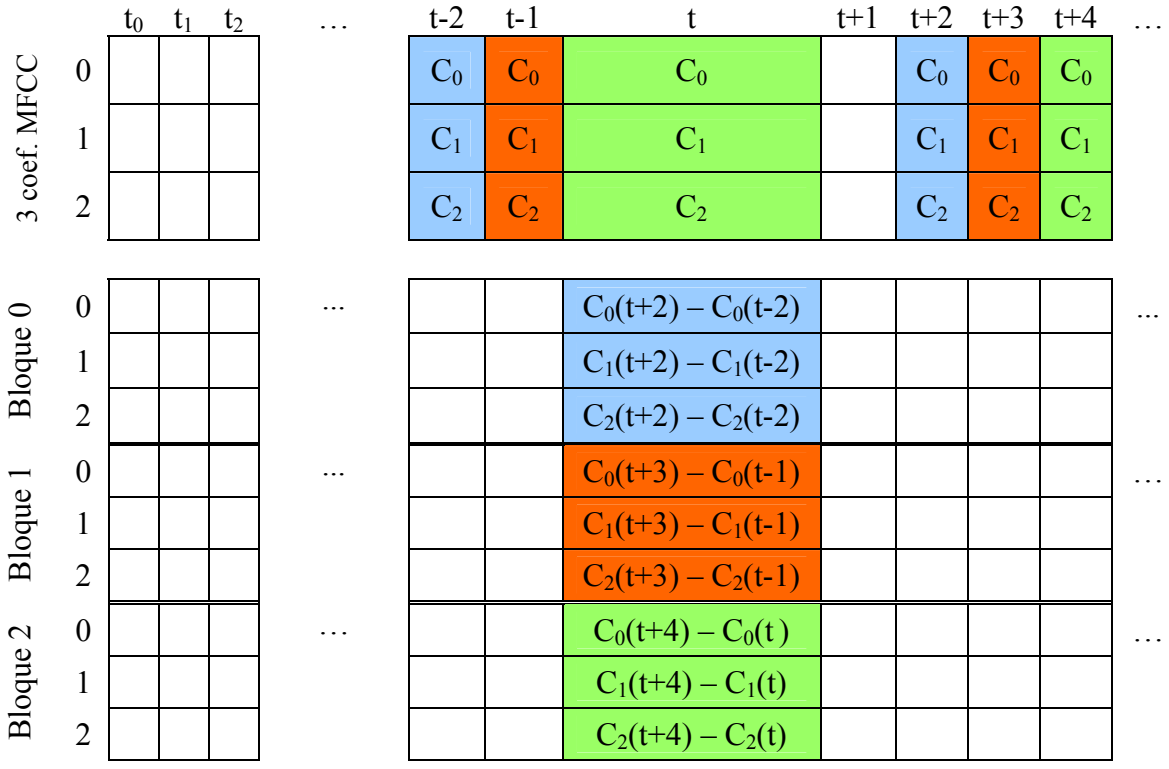


Figura 9. Ejemplificación del cálculo de parámetros SDC 3-2-1-3

Para el bloque 0 tendríamos i=0:
$\Delta C_n(t,0) = C_n(t+d) - C_n(t-d) = C_n(t+2) - C_n(t-2)$
Para el bloque 1 tendríamos i=1:
$\Delta C_n(t,1) = C_n(t+P+d) - C_n(t+P-d) = C_n(t+3) - C_n(t-1)$
Para el bloque 2 tendríamos i=2:
$\Delta C_n(t,0) = C_n(t+2P+d) - C_n(t+2P-d) = C_n(t+4) - C_n(t)$

Una de las configuraciones más típicas a la hora de emplear este tipo de parametrización es 7-2-3-7. De esta forma, a la hora de construir el vector de características para un instante t se tendrán en cuenta aportaciones de (k-1)*P+d tramas hacia delante, es decir (7-1)*3+2=20 tramas y d tramas hacia atrás, es decir 2 tramas.

En el caso de que las ventanas sean de 20ms con 50% de solapamiento, cada vector de coeficientes SDC tendrá una dependencia temporal de 100ms hacia delante y de 10ms hacia atrás.

Técnicas de compensación

En este apartado se presentarán distintas técnicas, que llevadas a cabo a nivel de parámetros, tratarán de compensar la influencia del ruido y otros efectos perturbadores en la señal.

- **Normalización por media cepstral *(CMN o Cepstral Mean Normalization)***

Esta técnica es una de las más populares desarrolladas en el campo de la normalización de canal. Consiste básicamente en restar a los vectores de parámetros la media de dichos vectores estimada a lo largo de todo el fichero, bajo la hipótesis de que el canal es un elemento de variación lineal en el dominio cepstral y que por tanto su contribución principal es a la media de los vectores cepstrales. En [Furui, 1981; Garcia y Mammone, 1999] puede encontrarse una descripción más amplia de la teoría cepstral y de CMN.

- **RASTA filtering**

El filtrado RASTA [Hermansky y Morgan, 1994], al igual que la normalización CMN, va orientado a eliminar las distorsiones introducidas por el canal. La complejidad de esta técnica es superior a la de CMN, haciendo que el espectro de la señal de voz resultante dependa de instantes pasados y realzando las transiciones espectrales.

- **Feature Warping y Feature Mapping**

La técnica *Feature Mapping* [Reynolds, 2003] es una técnica de normalización orientada a los datos que presenta unos resultados mejores que *Feature Warping* [Pelecanos y Sridharan, 2001]. En feature warping el objetivo era conseguir una distribución final gaussiana de media nula y varianza unidad (técnicas de transformación de histograma), tratando cada dimensión del vector de parámetros por separado. En Feature Mapping, por su parte, se tiene en cuenta la correlación entre dimensiones a la hora de realizar la normalización.

6. SVMs para reconocimiento de locutor e idioma

Las máquinas de vectores soporte son básicamente un algoritmo de clasificación de patrones binario, cuyo objetivo es asignar cada patrón a una clase [Campbell *et al.,* 2006b]. 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), clasificando cada una de las ovejas en el conjunto blanco o negro.

Comenzaremos la explicación de las máquinas de vectores soporte [Burges, 1998] haciendo uso del caso más simple, el caso donde los datos son linealmente separables. Más adelante extrapolaremos la solución a problemas donde los datos no cumplan esta característica.

Los datos con los que entrenaremos el sistema serán una serie de vectores etiquetados, de la forma: $\{\vec{x}_i, y_i\} \quad i = 1,....,l$

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

$\vec{y}_i \in \{-1, 1\}$ representa etiqueta de la clase a la que pertenece cada vector

El problema consistirá en asignar cada vector a su clase correspondiente, 1 ó -1, para ello se construirá un hiperplano de separación que divida el espacio R^d en dos regiones. Supongamos que tenemos dicho hiperplano, las muestras que caigan en una región pertenecerán a clase -1 y las que caigan en la otra a la clase 1. A este hiperplano se le conoce como hiperplano de separación.

Los puntos $\vec{x}$ que caen justo en este hiperplano satisfarán la ecuación: $\vec{w} \cdot \vec{x} + b = 0$

Donde: $\vec{w}$ es un vector normal al hiperplano de separación

b es una constante

A la hora de buscar este hiperplano de separación óptimo las distancias cobrarán una especial relevancia. Así, $\frac{|b|}{\|\vec{w}\|}$ será la distancia perpendicular desde el hiperplano al origen ($\|\vec{w}\|$ norma euclídea de $\vec{w}$). Llamaremos d_+ y d_- a dos distancias más, las existentes entre el hiperplano de separación y las muestras más cercanas de la clase 1 y -1 respectivamente.

En este momento estamos en condiciones de definir el margen del hiperplano de separación, este margen será la distancia entre las muestras más cercanas de las clases, haciendo uso de d_+ y d_- podemos definir el margen como:

$$m = d_+ + d_-$$

Para el caso que nos ocupa, datos linealmente separables, el objetivo será encontrar, de entre todos los posibles, el hiperplano de separación que hace máximo este margen m.

A la hora de formular el problema formalmente supondremos que todos los datos de entrenamiento cumplen una de las siguientes restricciones:

$$\vec{x}_i \cdot \vec{w} + b \geq +1 \quad si\ y_i = +1$$

$$\vec{x}_i \cdot \vec{w} + b \leq -1 \quad si\ y_i = -1$$

Las restricciones anteriores se traducen en que los vectores son separables en dos clases, combinando ambas restricciones en una obtenemos:

$$y_i(\vec{x}_i \cdot \vec{w} + b) - 1 \geq 0 \quad \forall i$$

Ahora consideremos los puntos para los se cumple: $\vec{x}_i \cdot \vec{w} + b = +1$

Estos puntos, los más cercanos al hiperplano y conocidos como vectores soporte, estarán contenidos en un nuevo hiperplano, al que llamaremos H_1, y cuya ecuación es la expresada anteriormente. De manera análoga definimos el hiperplano H_2, cuya ecuación será:

$$\vec{x}_i \cdot \vec{w} + b = -1$$

Los hiperplanos H_1 y H_2 son paralelos al hiperplano de separación, por lo tanto su componente normal seguirá siendo $\vec{w}$, las correspondientes distancias al origen serán: $\frac{|1-b|}{\|\vec{w}\|}$ para el hiperplano H_1 y $\frac{|-1-b|}{\|\vec{w}\|}$ para el H_2.

Si el problema cumple las restricciones que hemos indicado anteriormente las distancias d_+ y d_- serán $\frac{1}{\|\vec{w}\|}$ por lo que el margen $m = d_+ + d_- = \frac{1}{\|\vec{w}\|} + \frac{1}{\|\vec{w}\|} = \frac{2}{\|\vec{w}\|}$.

Como comentamos al principio de la sección, el objetivo de las máquinas de soporte es encontrar el hiperplano de separación que maximiza el margen. Con la formulación mostrada hasta ahora el problema se reduce a minimizar $\|\vec{w}\|^2$ sujeto a la restricción:

$$y_i(\vec{x}_i \cdot \vec{w} + b) - 1 \geq 0 \quad \forall i\,.$$

Veamos un ejemplo gráfico de lo visto hasta el momento. Por comodidad y facilidad de interpretación se ilustrará en un espacio de 2 dimensiones, R^2. Los círculos negros representan muestras pertenecientes a la clase 1 y los cuadrados muestras de la clase -1. El objetivo del algoritmo será encontrar el hiperplano que separe estas muestras de una manera óptima. En la Figura 10 a) se ilustra la distribución espacial de las muestras y varios hiperplanos de separación posibles.

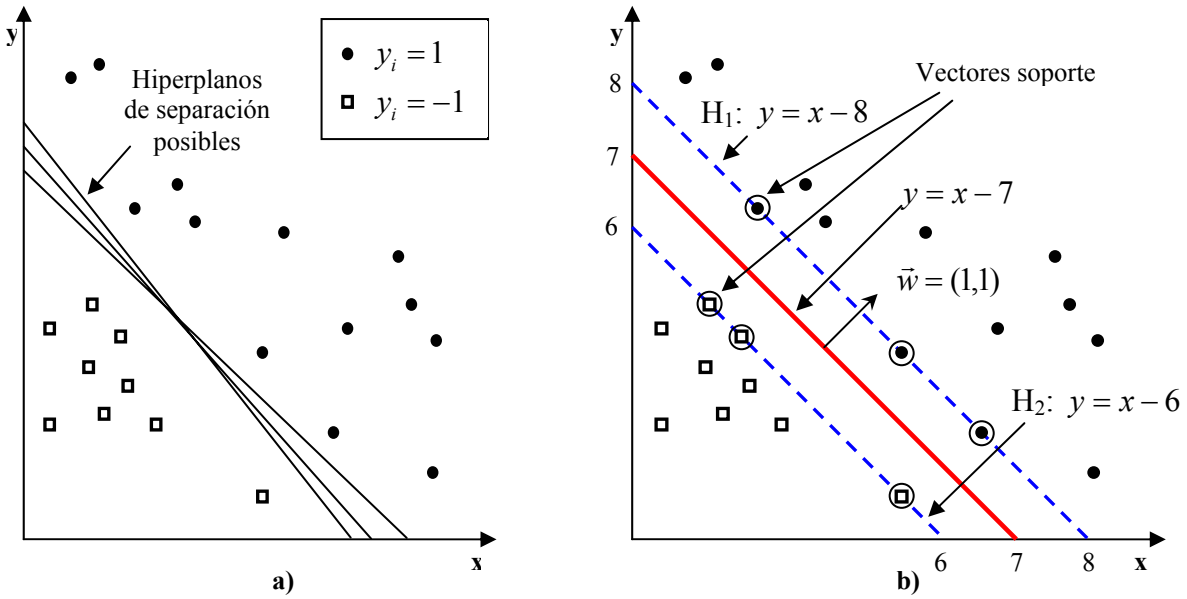


Figura 10. Representación de muestras pertenecientes a dos clases distintas, a) observaciones y posibles hiperplanos de separación, b) hiperplano de separación óptimo e hiperplanos H_1 y H_2

Las muestras cumplen las restricciones mostradas anteriormente, además, al estar en dos dimensiones los hiperplanos serán rectas, una dimensión menor que el espacio de características. Teniendo en cuenta esto y sabiendo que $\vec{w}=(1,1)$ y $b=-7$ definimos la recta de separación y las rectas H_1 y H_2 como sigue:

$$\vec{x}_i\cdot\vec{w}+b=0 \;\Rightarrow\; (x,y)\cdot(w_x,w_y)+b=0 \;\Rightarrow\; (x,y)\cdot(1,1)-7=0 \;\Rightarrow\; y=7-x$$
$$H_1:\;\; \vec{x}_i\cdot\vec{w}+b=+1 \;\Rightarrow\; (x,y)\cdot(w_x,w_y)+b=1 \;\Rightarrow\; (x,y)\cdot(1,1)-7=1 \;\Rightarrow\; y=8-x$$
$$H_2:\;\; \vec{x}_i\cdot\vec{w}+b=-1 \;\Rightarrow\; (x,y)\cdot(w_x,w_y)+b=-1 \;\Rightarrow\; (x,y)\cdot(1,1)-7=-1 \;\Rightarrow\; y=6-x$$

La recta de separación, H_1 y H_2 se muestran sobre los datos en la Figura 10 b). Es preciso hacer notar que H_1, H_2 y la recta de separación son paralelos, por lo que el vector normal a todos es $\vec{w}$, también debemos darnos cuenta de que ninguna de las muestras de entrenamiento caen entre las rectas H_1 y H_2.

Siguiendo con el ejemplo definiremos las distancias perpendiculares de las rectas al origen y el margen, variables que se representa sobre los datos en la Figura 11:

$$d=\frac{|b|}{\|\vec{w}\|}=\frac{|-7|}{\|(1,1)\|}=\frac{7}{\sqrt{2}}=4.95$$
$$d_{H1}=\frac{|1-b|}{\|\vec{w}\|}=\frac{|1+7|}{\|(1,1)\|}=\frac{8}{\sqrt{2}}=5.66$$
$$d_{H2}=\frac{|-1-b|}{\|\vec{w}\|}=\frac{|-1+7|}{\|(1,1)\|}=\frac{6}{\sqrt{2}}=4.24$$
$$m=d_{+}+d_{-}=\frac{1}{\|\vec{w}\|}+\frac{1}{\|\vec{w}\|}=\frac{2}{\|\sqrt{2}\|}=\sqrt{2}=1.41$$

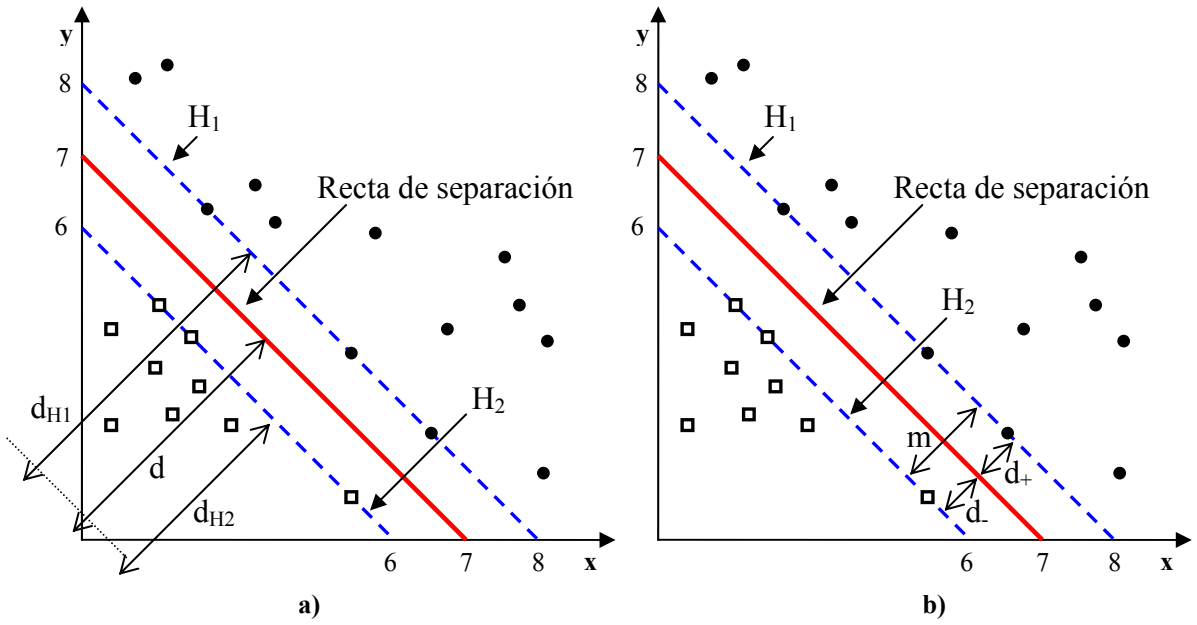


Figura 11. Representación sobre el plano de: a) distancias d, d_{H1} y d_{H2}, b) distancias d_+, d_- y margen m

Una vez visto este pequeño ejemplo seguiremos con el desarrollo teórico, para ello es necesario cambiar a la formulación de Lagrange. La formulación de Lagrange permite resolver un problema de optimización, como es nuestro caso, bajo una serie de restricciones mediante la introducción de unas nuevas variables, los multiplicadores de Lagrange, α_i. Puede demostrarse que es posible obtener el hiperplano óptimo de separación, $\vec{w}$, mediante una combinación lineal de los vectores soporte. El peso de cada uno de estos vectores se obtiene mediante los multiplicadores de Lagrange. Además de esto existen dos motivos más por los que cambiar la formulación:

- El primero de ellos es que la restricción $y_i(\vec{x}_i \cdot \vec{w} + b) - 1 \geq 0 \quad \forall i$ puede ser reemplazada por restricciones en los multiplicadores de Lagrange, lo que hace que la dificultad disminuya.
- En segundo lugar, una de las propiedades más importantes de la reformulación, los datos de entrenamiento sólo aparecerán en forma de productos escalares entre vectores. Esta propiedad es tan importante porque permite generalizar el procedimiento a casos no lineales.

Comenzamos introduciendo los multiplicadores de Lagrange (positivos) [Burges, 1998], $\alpha_i \quad i = 1,...,l$, uno por cada desigualdad de la restricción. Recordemos que la regla en general dice que para restricciones de la forma $c_i \geq 0$, como es nuestro caso, debemos introducir un multiplicador de Lagrange, α_i, que multiplique a la restricción, posteriormente restaremos estas restricciones de la función objetivo, que en nuestro caso es la función a minimizar, $\frac{1}{2}\|\vec{w}\|^2$. Esto da como resultado:

$$L_P \equiv \frac{1}{2}\|\vec{w}\|^2 - \sum_{i=1}^{l} \alpha_i \left(y_i (\vec{x}_i \cdot \vec{w} + b) - 1 \right) = \frac{1}{2}\|\vec{w}\|^2 - \sum_{i=1}^{l} \alpha_i y_i (\vec{x}_i \cdot \vec{w} + b) + \sum_{i=1}^{l} \alpha_i$$

Ahora debemos minimizar L_P con respecto a las variables fundamentales $\vec{w}, b$, además, simultáneamente necesitamos que las derivadas de L_P con respecto a las variables duales, α_i, desaparezcan. A todo esto junto con la restricción $\alpha_i \geq 0$, lo llamaremos conjunto de restricciones C_1.

Debido a que tanto la función como el conjunto formado por los puntos que satisfacen las restricciones son convexos, la minimización L_P se puede tratar como un problema de programación cuadrática convexo. Para solucionar el problema podemos hacer uso de la dualidad y resolver el problema dual, cuya resolución será equivalente. El problema dual consiste en maximizar L_P sujeto a unas condiciones. Por un lado el gradiente de L_P con respecto a $\vec{w}$ y b debe anularse, por otro debemos seguir cumpliendo $\alpha_i \geq 0$. Llamaremos a este conjunto de restricciones C_2.

Una de las propiedades más importantes del problema dual, conocido con *Wolfe dual*, es que el máximo de L_P sujeto al conjunto de restricciones C_2, se alcanza con los mismos valores de $\vec{w}$, b y α que se alcanza el mínimo de L_P sujeto al conjunto C_1. Esta propiedad es la que hace que sea equivalente la resolución de uno u otro problema.

El primer requisito del conjunto C_2, que el gradiente de L_P con respecto a $\vec{w}$ y b desaparezca nos da las condiciones:

$$\frac{\partial}{\partial b}L(\vec{w},b,\alpha)=0 \text{ y } \frac{\partial}{\partial w}L(\vec{w},b,\alpha)=0 \quad \Rightarrow \quad \begin{matrix}\vec{w}=\sum_{i=1}^{l}\alpha_i y_i=0 \\ \vec{w}=\sum_{i=1}^{l}\alpha_i y_i \vec{x}_i\end{matrix}$$

Que sustituyéndolas en el problema original nos quedará:

$$L_D=\sum_i \alpha_i - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j \vec{x}_i\cdot\vec{x}_j$$

Debemos darnos cuenta de que ahora estamos resolviendo el problema dual, identificado con el subíndice *D*, mientras que anteriormente comenzamos con el fundamental, *P*. Una de las formas de resolver este problema es mediante el algoritmo del gradiente.

Como vimos anteriormente, el vector solución, $\vec{w}$, puede escribirse en función de los vectores de entrenamiento, $\vec{x}_i$, $\vec{w}=\sum_{i=1}^{l}\alpha_i y_i \vec{x}_i$. Cada vector de entrenamiento tendrá asociado un multiplicador de Lagrange, α_i. El valor de este multiplicador será mayor que cero para los vectores que caigan en el hiperplano H_1 o H_2. Para el resto de vectores de entrenamiento, el multiplicador de Lagrange será cero, por lo que no tendrán ninguna relevancia en el entrenamiento. El hiperplano de separación dependerá sólo de los vectores soporte, las muestras más cercanas al límite entre ambas clases (véase Figura 10 b).

Una vez hallado el hiperplano de separación debemos definir una forma de clasificar las muestras, la función $f(\vec{x}_i)=\vec{w}\cdot\vec{x}_i+b$ mide la distancia de cada vector al hiperplano de separación. Esta función es justo lo que estábamos buscando, será positiva para las muestras pertenecientes a la clase 1 y negativa para las de la clase -1, lo que nos permitirá clasificar cualquier muestra en su clase correspondiente.

El elevado nivel de ruido y los efectos de canal, son dos de las causas que pueden provocar cierto solapamiento entre muestras de ambas clases. Con la formulación vista hasta el momento, esas muestras no cumplirían la restricción $y_i(\vec{x}_i\cdot\vec{w}+b)-1\geq 0 \quad \forall i$. Lo que debemos hacer para poder afrontar sistemas con este tipo de problemas es relajar esta restricción, para ello introduciremos un margen de error, $\xi_{c,i}$. La restricción será ahora:

$$y_i(\vec{x}_i\cdot\vec{w}+b)\geq 1-\xi_{c,i} \quad \forall i$$

Al añadir esta nueva variable pasaremos de uno a dos criterios a la hora de encontrar el hiperplano de separación:

- Maximizar el margen entre clases (criterio que ya teníamos anteriormente).
- Minimizar la función de pérdidas que será proporcional a las muestras incorrectamente clasificadas.

La relevancia de un criterio frente al otro se controla a través de una variable, a la que llamaremos coste, *C*.

La nueva función a maximizar será:

$$\text{Maximizar: } \tau(\vec{w},\xi_c)=\frac{1}{2}\|\vec{w}\|^2+C\frac{1}{l}\sum_{i=1}^{l}\xi_{c,i}$$

$$\text{Sujeto a: } 0\le\xi_{c,i}\le 1-y_i f(\vec{x}_i)$$

El primer término de la ecuación hace referencia a la minimización de $\vec{w}$ con vista a maximizar el margen entre clases, el segundo término tiene en cuenta las muestras incorrectamente clasificadas, más adelante veremos que este término está relacionado con la función de pérdidas.

$\xi_{c,i}$ será cero para aquellas muestras correctamente clasificadas, $y_i f(\vec{x}_i)>1$, es decir la etiqueta y la distancia coinciden en signo. Por el contrario, será distinto de cero para las muestras incorrectamente clasificadas $y_i f(\vec{x}_i)<1$. Debemos darnos cuenta de que las muestras de una clase 1 que caigan entre el hiperplano de separación y el plano H_1 estarán clasificadas correctamente pero tendrán un $\xi_{c,i}$ asociado distinto de cero. Lo mismo sucederá para las muestras de la clase -1. Estas muestras cumplirán:

$$0<y_i f(\vec{x}_i)<1$$

Siguiendo el ejemplo anterior, representaremos sobre él algunas muestras incorrectamente clasificadas, de forma que podamos ver el significado físico de la variable ξ_c.

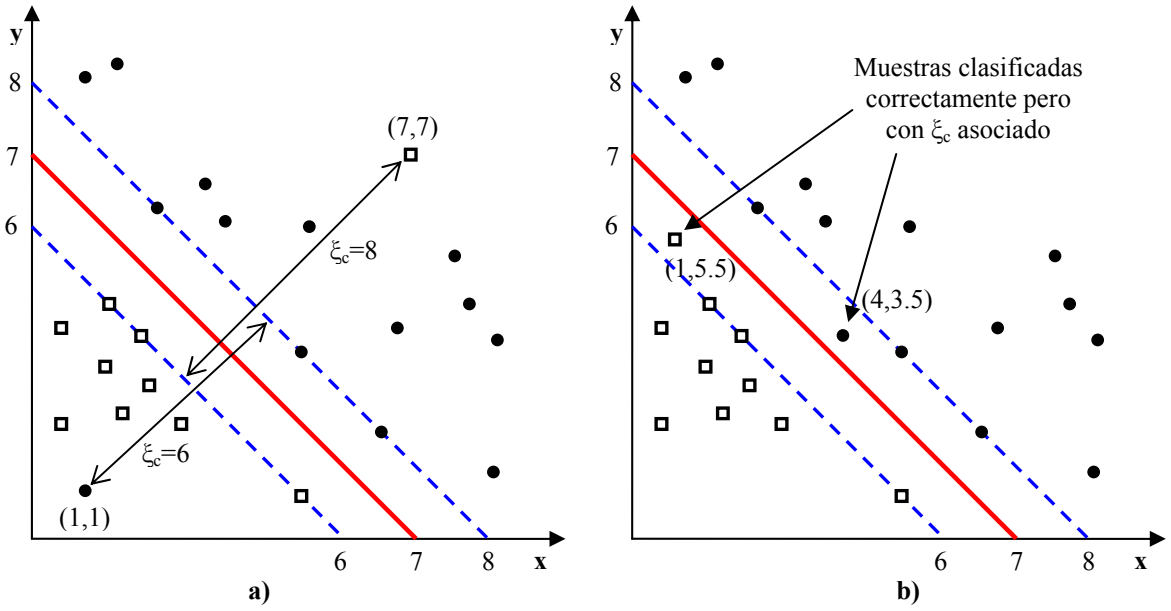


Figura 12. SVC: a) Muestras clasificadas incorrectamente, con su valor de ξ_c asociado, b) muestras clasificadas correctamente pero con ξ_c asociado

En la Figura 12 podemos ver la muestra (1, 1), correspondiente a la clase 1 y clasificada incorrectamente por el plano de separación:

$y_i f(\vec{x}_i) = y_i(\vec{w}\cdot\vec{x}_i + b) = 1((1,1)\cdot(1,1) - 7) = -5 < 0 \quad \Rightarrow$ Incorrectamente clasificada

Lo mismo sucede con la muestra (7, 7), correspondiente a la clase -1:

$y_i f(\vec{x}_i) = y_i(\vec{w}\cdot\vec{x}_i + b) = -1((1,1)\cdot(7,7) - 7) = -7 < 0 \quad \Rightarrow$ Incorrectamente clasificada

Los valores de ξ_c para estas dos muestras serán 6 y 8 respectivamente, de esta forma:

$y_i(\vec{x}_i \cdot \vec{w} + b) \geq 1 - \xi_{c,i} \quad \Rightarrow \quad 1((1,1)\cdot(1,1) - 7) = -5 \quad \Rightarrow \quad -5 \geq 1 - 6$

$y_i(\vec{x}_i \cdot \vec{w} + b) \geq 1 - \xi_{c,i} \quad \Rightarrow \quad -1((7,7)\cdot(1,1) - 7) = -7 \quad \Rightarrow \quad -7 \geq 1 - 8$

Las muestras (1, 5.5) y (4, 3.5) están correctamente clasificadas por el hiperplano de separación, pero tienen una penalización asociada, ξ_c, véase Figura 12.

$y_i f(\vec{x}_i) = y_i(\vec{w}\cdot\vec{x}_i + b) = -1((1,1)\cdot(1,5.5) - 7) = 0.5 \in [0,1] \Rightarrow$ Correctamente clasificada pero con ξ_c asociado.

$y_i f(\vec{x}_i) = y_i(\vec{w}\cdot\vec{x}_i + b) = 1((1,1)\cdot(4,3.5) - 7) = 0.5 \in [0,1] \Rightarrow$ Correctamente clasificada pero con ξ_c asociado.

La función de pérdidas del sistema será en definitiva las suma de los márgenes, $\xi_{c,i}$, sumados a cada una de las muestras, podemos expresar la función de pérdidas de cada muestra de la siguiente manera:

$$f_{pérdidas}(\vec{x}_i) = \max\{0, 1 - y_i f(\vec{x}_i)\}$$

Cuando la muestra esté incorrectamente clasificada, $1 - y_i f(\vec{x}_i)$ nos dará el valor de $\xi_{c,i}$ de esa muestra, en el caso de que esté bien clasificada, $1 - y_i f(\vec{x}_i)$ nos dará un número menor que cero, por lo que la función de pérdidas devolverá un cero.

Hasta ahora hemos visto el funcionamiento de la máquina de vectores soporte basada en clasificación (SVC) partiendo de una premisa, los datos de entrenamiento eran linealmente separables. A continuación extrapolaremos los resultados a conjuntos de datos que no cumplan este requisito.

Comenzaremos el estudio con unos datos de entrenamiento no linealmente separables, hemos de tener en cuenta que estos datos están en un espacio de dimensión d. Una de las formas habituales de conseguir que estos datos sean linealmente separables es llevarlos a un espacio de características de dimensión mayor, $d' > d$. Para ello definiremos una función que mapee cada vector de características del espacio de dimensión d al espacio de dimensión d':

$$\varphi(\cdot): \quad R^d \rightarrow R^{d'}$$

Si nos fijamos en toda la formulación llevada a cabo hasta ahora, los vectores sólo aparecen como productos internos en el espacio de características (esta fue una de las causas por las que cambiamos a la formulación de Lagrange). Este hecho nos permitirá definir una función *kernel* que nos permita calcular el producto interno de dos vectores sin necesidad de conocer explícitamente el vector mapeado en el espacio de características final. A esto se le conoce como el truco del kernel *(kernel trick)*.

$$k(\vec{x}_i, \vec{x}_j) = \varphi(\vec{x}_i) \cdot \varphi(\vec{x}_j)$$

La Figura 13 muestra un ejemplo de un conjunto de datos en R^2 no separables linealmente, tras el mapeo de los vectores a un espacio R^3 las muestras son fácilmente separables por el hiperplano. El ejemplo es perfectamente extrapolable a espacios de dimensiones mayores.

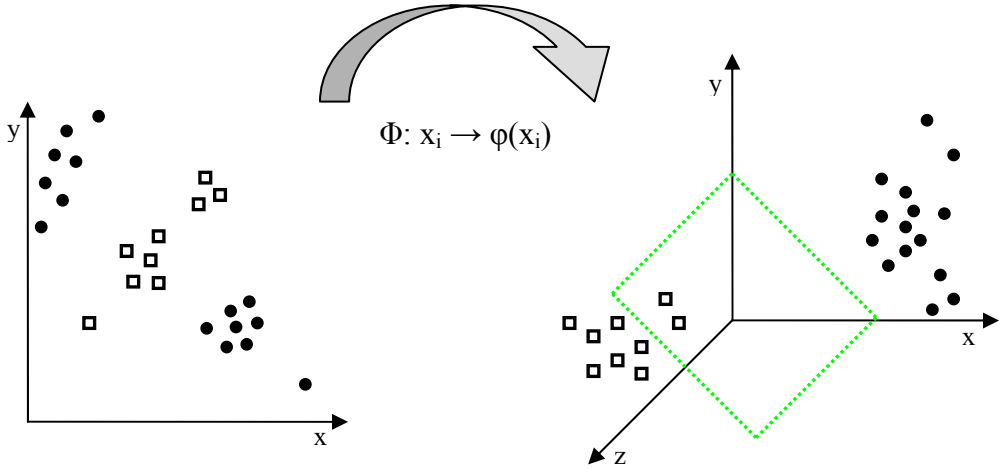


Figura 13. Mapeo de los vectores en un espacio de características de dimensión mayor, $R^2 \rightarrow R^3$

Existen muchos tipos de kernel: lineales, polinómicos, radiales, etc., como es de esperar cada uno tendrá unas u otras propiedades. En este trabajo no entraremos en detalle en este aspecto, es posible encontrar multitud de libros dedicados en exclusiva a su estudio y diseño [Cristianini y Shawe-Taylor, 2000]. Tan sólo haremos mención a un aspecto relevante, para que un kernel sea considerado como tal debe cumplir una serie de condiciones, conocidas como *condiciones de Merger*.

Uno de los puntos débiles del sistema SVM es que a priori desconocemos el kernel óptimo, ya que dependerá del tipo de problema. En reconocimiento de locutor el kernel GLDS *(Generalized Linear Discriminative Sequence)* [Campbell, 2002] presenta un buen comportamiento, por tanto será este el kernel usado por nuestros sistemas. Uno de sus pasos principales es la expansión polinómica de tercer grado [Wan y Campbell, 2000], cuyo resultado serán vectores de mayor tamaño (con un mayor número de dimensiones o características). En este nuevo espacio de características las probabilidades de encontrar una frontera lineal de separación entre clases (de un lado el usuario y de otro lado el resto de locutores) son mayores. Este efecto llevado al extremo consiste en tener más dimensiones que vectores, en este caso puede demostrarse que siempre vamos a poder encontrar una frontera (también conocido como hiperplano de separación) entre ambas clases. El número de dimensiones del vector expandido responde a la siguiente formula:

$$L' = \binom{L+G-1}{G} = \frac{(L+G-1)!}{G!((L+G-1)-G)!} = \frac{(L+G-1)!}{G!(L-1)!}$$

Donde G es el grado de expansión, L es la longitud inicial del vector de parámetros y L' es la longitud del vector de salida. La formula se corresponde con combinaciones con repetición, es decir combinaciones de L elementos tomados de G en G.

Máquinas de vectores soporte basadas en regresión (SVR)

Hasta el momento hemos visto las máquinas de vectores soporte basadas en clasificación, gracias a esta técnica nos era posible abordar problemas de reconocimiento de patrones, de forma que la tarea resultara sencilla. De ahora en adelante abordaremos una nueva aproximación que nos permitirá abordar problemas mucho más generales, para ello haremos uso de las máquinas de vectores soporte basadas en regresión.

El objetivo de la regresión es la estimación de una función, en lugar de predecir una etiqueta $y_i = \{\pm 1\}$ como era el caso de la clasificación. Para el sistema basado en regresión, y_i será considerado como una función dependiente de $\vec{x}_i$, esta función podrá tomar cualquier valor real, a diferencia de lo que sucedía en clasificación donde sólo podía tomar los valores correspondientes a las etiquetas de las clases.

Formalizando lo que hemos visto hasta ahora, $g_n(\vec{x}_i)$ será la función n-dimensional estimada por el sistema de forma que:

$$g_n(\vec{x}_i) = y_i \quad con \quad y_i \in \Re \quad \forall i$$

El objetivo de los sistemas basados en regresión será aproximar la función de decisión, $f(\vec{x}_i) = \vec{w} \cdot \vec{x}_i + b$, que recordemos que en clasificación nos devolvía la distancia del las muestras al hiperplano de separación, a la función estimada.

$$f(\cdot) \approx g_n(\cdot)$$

Una de las principales diferencias entre los sistemas de clasificación y regresión guarda relación con las muestras penalizadas. El sistema basado en clasificación sólo penalizaba a las muestras incorrectamente clasificadas y las que caían entre los hiperplanos H_1 y H_2, es decir, las muestras para las que:

$$f(\cdot) < g_n(\cdot) \quad \Rightarrow \quad y_i f(\vec{x}_i) < 1$$

El sistema basado en regresión penalizará cualquier muestra que caiga fuera del hiperplano H_1 o H_2. Es decir:

$$f(\cdot) < g_n(\cdot) \quad ó \quad f(\cdot) > g_n(\cdot)$$

Por tanto, la función de pérdidas del sistema basado en regresión será diferente a la empleada por el sistema en clasificación. Una de las funciones más populares en regresión es la función épsilon, ε, [Muller *et al.,* 1997]. Esta función introducirá un cierto grado de tolerancia a la hora de penalizar las muestras, tolerancia que será controlada a través del parámetro ε. De esta forma sólo se penalizarán las muestras que cumplan:

$$\left| f(\cdot) - g_n(\cdot) \right| > \varepsilon$$

Una vez definida la función de pérdidas que usaremos con el sistema basado en regresión pasaremos a plantear las ecuaciones necesarias para encontrar el hiperplano de separación. Los objetivos serán los mismo que se expusieron en clasificación, por un lado maximizar el margen y por otro minimizar la penalización. El parámetro *C*, coste,

será usado para dar más relevancia a un criterio frente al otro. Cuanto más aumentemos este valor más se aproximará $f(\cdot)$ a $g_n(\cdot)$.

$$\text{Minimizar: } \tau\left(\vec{w},\xi_{r,i},\xi'_{r,i}\right)=\frac{1}{2}\|\vec{w}\|^2+C\frac{1}{l}\sum_{i=1}^{l}\left(\xi_{r,i}+\xi'_{r,i}\right)$$

$$\text{Sujeto a: } \begin{aligned} 0 &\le f(\vec{x}_i)-y_i \le \xi_{r,i}+\varepsilon \\ 0 &\le y_i-f(\vec{x}_i) \le \xi'_{r,i}+\varepsilon \end{aligned}$$

De esta forma las muestras que estén a una distancia del plano menor que ε no serán penalizadas ni tenidas en cuenta por el sistema. Al igual que sucedía en el sistema basado en clasificación con las muestras que estaban bien clasificadas.

Al contrario de lo que sucedía en clasificación, en regresión hay dos tipos diferentes de márgenes, $\xi_{r,i}$ y $\xi'_{r,i}$. El primero de ellos se aplicará a muestras en las que $f(\vec{x}_i)>g_n(\vec{x}_i)+\varepsilon$, el segundo se aplicará al resto de muestras, $f(\vec{x}_i)<g_n(\vec{x}_i)-\varepsilon$.

A continuación presentaremos un pequeño ejemplo numérico para ilustrar lo explicado sobre los márgenes asociados a las muestras. La Figura 14 muestra la representación de los hiperplanos y los márgenes asociados a las muestras en un sistema SVR.

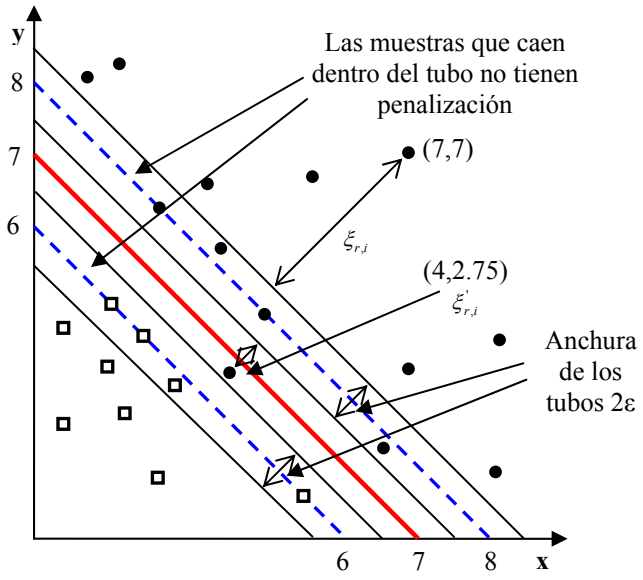


Figura 14. SVR, representación de las fronteras del sistema y muestras con ξ_r asociado

Para el punto (7, 7):

$f(\vec{x}_i)=\vec{w}\cdot\vec{x}_i-b=(1,1)\cdot(7,7)-7=7 \;\Rightarrow\; f(\vec{x}_i)>g_n(\vec{x}_i)+\varepsilon \;\Rightarrow\; 7>1+0.5=1.5$ por lo tanto deberemos aplicar una penalización al punto (7, 7), $\xi_{r,i}$.

Para el punto (4, 2.75):

$f(\vec{x}_i)=\vec{w}\cdot\vec{x}_i-b=(1,1)\cdot(4,2.75)-7=-0.25 \;\Rightarrow\; f(\vec{x}_i)<g_n(\vec{x}_i)-\varepsilon \;\Rightarrow\; -0.25<1+0.5=1.5$ por lo tanto deberemos aplicar una penalización al punto (4, 2.75), $\xi'_{r,i}$.

Dejando a un lado el ejemplo seguiremos la explicación definiendo la función de pérdidas del sistema:

$$f'_{pérdidas}(\vec{x}_i) = \max\{0, |y_i - f(\vec{x}_i)| - \varepsilon\}$$

De la misma forma que se hizo en clasificación, será necesario introducir los multiplicadores de Lagrange para resolver el problema, de este modo llegamos al siguiente problema de optimización (para valores de C y ε elegidos a priori):

$$\text{Maximizar: } W(\alpha,\alpha^*) = -\varepsilon\sum_{i=1}^{l}(\alpha_i^* + \alpha_i) + \sum_{i=1}^{l}(\alpha_i^* - \alpha_i)y_i = -\frac{1}{2}\sum_{i,j=1}^{l}(\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j)k(\vec{x}_i,\vec{x}_j)$$

$$\text{Sujeto a: } 0 \le \alpha_i, \alpha_i^* \le C \quad \forall i \quad y \quad \sum_{i=1}^{l}(\alpha_i - \alpha_i^*) = 0$$

Una propiedad interesante del sistema basado en regresión, que no es aplicable a los sistemas basados en clasificación, es que la función de pérdidas ε estimará el plano $\vec{w}$ mediante la técnica de máximo a posteriori *(MAP)*. Puede demostrarse que $e^{-f'_{pérdidas}(\cdot)}$ es proporcional a $p(\vec{w}\,|\,D,\varepsilon)$, es decir, la probabilidad a posteriori de $\vec{w}$ [Sollich, 1999] dados los datos y el valor de ε. Por lo tanto, minimizando la función de pérdidas maximizaremos la probabilidad de que $f(\cdot) = g_n(\cdot)$ (realmente maximizaremos el logaritmo de dicha probabilidad).

El mapeo de un espacio de características a otro de dimensión mayor empleado en SVC es perfectamente aplicable a SVR, de esta manera el sistema tendrá menos dificultades a la hora de estimar funciones no lineales.

Para concluir esta sección mostraremos una comparativa (Figura 15) entre las funciones de coste de las máquinas de vectores soporte basadas en clasificación (SVC) y regresión (SVR).

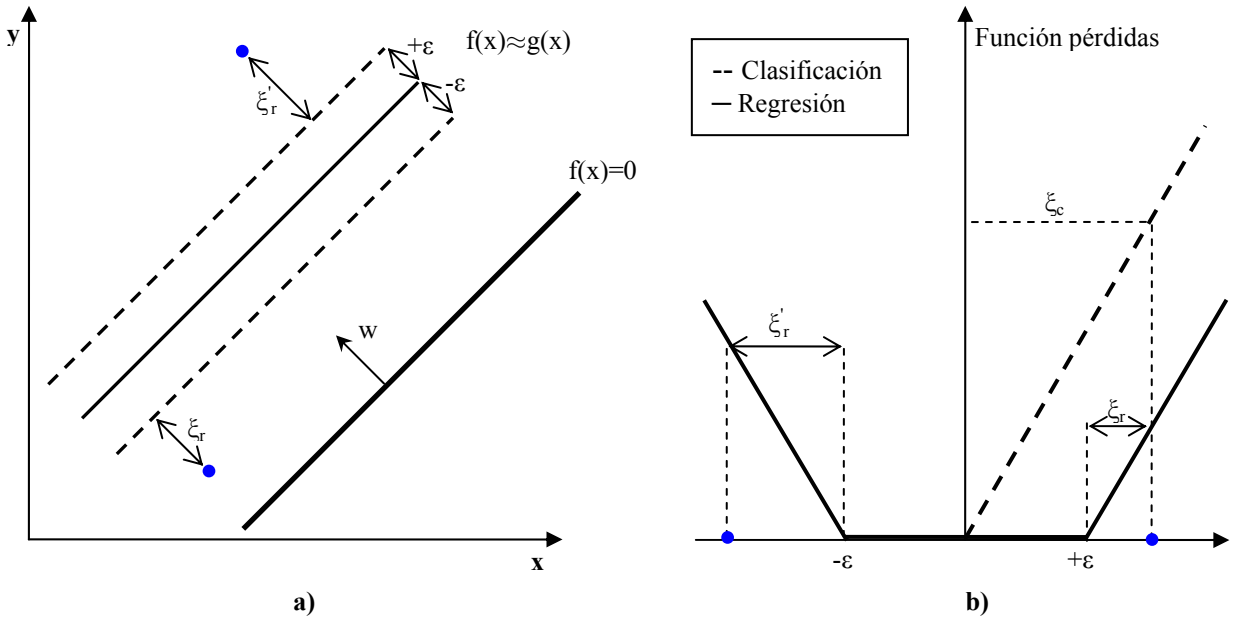


Figura 15. SVC vs. SVR: a) fronteras, b) función de pérdidas

La función de pérdida está centrada en $f(\vec{x}_i) = y_i$ para SVC ($f_{pérdidas}$) y en $f(\vec{x}_i) = g_n(\vec{x}_i)$ para SVR ($f'_{pérdidas}$). $f_{pérdidas}$ penaliza las muestras $\vec{x}_i$ tales que $y_i f(\vec{x}_i) < 1$, $f'_{pérdidas}$ penaliza las muestras $\vec{x}_i$ tales que $|f(\vec{x}_i) - g_n(\vec{x}_i)| > \varepsilon$.

Nuevo método implementado, épsilon-SVR-GLDS

Tanto el reconocimiento de locutor como de idioma es un problema de clasificación con únicamente dos clases. La mayor parte de los sistemas de reconocimiento de locutor e idioma están basados en SVC-GLDS, etiquetando ambas clases con las etiquetas +1 y -1 (valores adoptados a lo largo del desarrollo), para los vectores de la clase *Target* y *NonTarget* respectivamente.

El nuevo sistema SVM que implementaremos, épsilon-SVR-GLDS, basado en regresión, tendrá una función objetivo $g_n(\cdot)$ discreta, tomando el valor 1 para la clase *Target* y el valor -1 para la clase *NonTarget*. Debemos observar que en este caso, los vectores soporte no serán los más cercanos al hiperplano $\vec{w}$, como sucedía en clasificación, en ese caso minimizaban la función de pérdidas $f_{pérdidas}$, pero no tendrán porque coincidir con los que minimicen $f'_{pérdidas}$ (función pérdidas regresión).

Esta diferencia en el entrenamiento hace que el sistema sea más robusto a muestras espurias y vectores con ruido empleados en la construcción de $\vec{w}$, ya que en regresión los vectores soporte serán seleccionados de las regiones del espacio de características donde los vectores estén más concentrados. En el caso de SVC se usa un conjunto de vectores soporte cercano a la frontera entre clases, región en la cual los vectores suelen ser escasos. Este hecho hará que el hiperplano calculado por SVC sea más sensible que el calculado por SVR a los vectores espurios o con ruido comentados anteriormente.

Por último, un entrenamiento óptimo basado en épsilon-SVR requerirá una búsqueda exhaustiva de los parámetros *C* y ε. Algunos trabajos [Smola y Schoelkopf, 1998] relacionan el parámetro ε con el ruido o la variabilidad de la función a estimar. El valor óptimo de ε nos permitirá obtener una medida cuantitativa de variabilidad de las características en problemas de reconocimiento de locutor e idioma.

7. Protocolos, bases de datos y presentación de resultados

7.1 Protocolo de evaluación, evaluaciones NIST

La organización de las evaluaciones competitivas NIST (*National Institue of Standards and Technology)*, tanto en el desarrollo de las tecnologías de reconocimiento de locutor, NIST SRE *(Speaker Recognition Evaluation)*, como de idioma NIST LRE *(Language Recognition Evaluation)*, constituyen un foro científico y tecnológico que ha impulsado el desarrollo de los sistemas de reconocimiento basados en voz en la última década.

Estas evaluaciones están en constante revisión, intentando cubrir cada vez aspectos de mayor relevancia en los sistemas reales. En el campo de reconocimiento de locutor, se ha observado una evolución de las bases de datos y protocolos utilizados, esta evolución ha ido enfocada hacia las principales necesidades existentes en el campo de la verificación de locutor independiente de texto. En el campo de reconocimiento de idioma, la evolución ha ido orientada a incorporar un mayor número de idiomas y dialectos, además de una mayor variedad de condiciones de evaluación.

Las evaluaciones NIST (http://www.nist.gov/speech) tienen un carácter abierto, en ellas participan grupos de investigación de todo el mundo. Su intención es establecer condiciones competitivas que permitan determinar el rendimiento de los diferentes sistemas involucrados. Una de las principales finalidades de este tipo de evaluaciones es poder comparar los distintos sistemas, técnicas y configuraciones de cada uno de los integrantes. La comparación de dichos sistemas entre sí ha fomentado la competitividad, obteniendo sistemas con un grado de madurez suficiente como para funcionar en entornos con múltiples variabilidades de manera fiable.

El procedimiento de la evaluación define la medida de rendimiento y los datos sobre los que realizar la evaluación, es el mismo para todos los integrantes, y viene definido por: datos de entrenamiento, test y datos complementarios (para técnicas de fusión, normalización, etc.).

La tarea fundamental de las evaluaciones NIST SRE [NIST SRE] consiste en la verificación o detección de un determinado individuo en una grabación de prueba. Para ello, se dispone de una cantidad de datos de entrenamiento que varía desde 10 segundos (10s) de habla hasta 8 conversaciones (8c) de 2,5 minutos de habla de media. Del mismo modo, la longitud de la locución de prueba puede ser desde 10 segundos (10s) de habla hasta una conversación (1c) de 2,5 minutos de habla de media. La combinación entre una determinada cantidad de habla de entrenamiento y una cantidad determinada de habla de prueba se denomina *condición* de prueba. Todos los años existe una condición obligatoria, que desde 2004 es la denominada 1c-1c (una conversación de habla de entrenamiento y una conversación de habla de prueba).

Las evaluaciones NIST LRE [NIST LRE] constan de un número de idiomas y dialectos a identificar, al igual que sucede en el campo de locutor la evaluación está compuesta por diferentes pruebas de distinta dificultad. Estas pruebas irán orientadas detectar la presencia de un idioma en la grabación de prueba. Existen tres duraciones distintas de segmentos de test, dependiendo de la cantidad de voz que contengan se clasificarán en segmentos de 3, 10 y 30 segundos. Entre los datos suministrados para la evaluación se

encuentran algunos específicos para el entrenamiento de los modelos, pudiéndose ampliar este conjunto de datos por parte de cada participante.

Como forma de medir el rendimiento de cada sistema para poder establecer comparaciones se usa una función de coste, definida del siguiente modo:

$$C_{DET}(i) = C_{Miss} \cdot P_{Miss|T\arg et} \cdot P_{T\arg et} + C_{FalseAlarm|NonT\arg et} \cdot (1 - P_{T\arg et})$$

Donde C_{Miss} es el coste asociado a un falso rechazo, $C_{FalseAlarm}$ es el coste asociado a una falsa aceptación, $P_{T\arg et}$ es la probabilidad de que un fichero dado pertenezca al locutor o idioma en cuestión (establecida a priori), $P_{Miss|T\arg et}$ es el porcentaje de falsos rechazos (dado por el sistema) y $P_{FalseAlarm|NonT\arg et}$ es la probabilidad de una falsa aceptación (dada por el sistema).

Para la evaluación de locutor pasada, NIST SRE 2006, los valores fijados a priori fueron: $C_{Miss} = 1$, $C_{FalseAlarm} = 10$ y $P_{T\arg et} = 0.01$.
En el caso de reconocimiento de idioma, NIST LRE 2005, los valores fueron: $C_{Miss} = C_{FalseAlarm} = 1$ y $P_{T\arg et} = 0.5$.

7.2 Bases de datos

Todos los experimentos llevados a cabo durante la realización del presente proyecto, tanto de reconocimiento de locutor como de idioma, se hicieron siguiendo los protocolos de evaluación NIST. De este modo se pueden comparar los resultados con sistemas similares a nivel mundial.

El sistema de reconocimiento de locutor se evaluó sobre NIST SRE 2006 y el de reconocimiento de idioma sobre NIST LRE 2005, las evaluaciones más recientes realizadas por NIST en ambos campos a día de hoy.

Como base de datos en NIST SRE 2006, se ha utilizado el llamado corpus MIXER3, que es una extensión del corpus MIXER descrito en [Campbell *et al.,* 2004b] y utilizado en las evaluaciones NIST de 2004 y 2005. Presenta las mismas características que el corpus MIXER, es decir, múltiples canales de transmisión (telefónico terreno, celular e inalámbrico), múltiples terminales (micrófono de oreja, terminal de mano, manos libres, etc.) y múltiples idiomas (inglés, mandarín, árabe, ruso, español). Además, MIXER3 incorpora una nueva colección de datos grabados durante 2005 y que incluye muchos más idiomas (italiano, francés, varias variantes del hindú, etc.) y pronunciaciones (inglés hablado por hispanos, chinos y árabes no nativos).

Como datos de desarrollo, modelado de UBM y cohortes de Tnorm, se han utilizado datos de Switchboard-I [LDC] y de pasadas evaluaciones NIST de locutor (NIST SRE 2004 y NIST SRE 2005), correspondientes a Switchboard-II y MIXER.

La base de datos de NIST LRE 2005 incluye grabaciones de 7 idiomas distintos:

- Inglés
- Hindi
- Japonés
- Coreano
- Mandarín
- Español
- Tamil

La base de datos fue creada a partir de habla conversacional telefónica, un dato importante de estas evaluaciones es que no proporciona información acerca del género de los locutores.

Como datos de desarrollo, se han utilizado datos de pasadas evaluaciones NIST de idioma (NIST LRE 1996 y NIST LRE 2003), también se ha hecho uso de la base de datos Callfriend [LDC].

7.3 Rendimiento de los sistemas de reconocimiento, presentación de resultados

A la hora de diseñar e implementar sistemas de reconocimiento, necesitamos disponer de herramientas y procedimientos que nos permitan comprobar las capacidades y bondades del sistema en cuestión. Esto resultará en una serie de valores, curvas, etc. que servirán al desarrollador tanto para evaluar nuevas mejoras, como para comparar resultados con otros sistemas.

Los sistemas de verificación funcionan normalmente en dos pasos. En primer lugar, se calcula un valor de verosimilitud, puntuación *(score)*, entre la locución de prueba y el modelo de referencia correspondiente al locutor o idioma reclamado. En segundo lugar, este valor es comparado con un umbral, tomándose la decisión de aceptación si el valor es superior al umbral, o rechazo en caso contrario.

Podemos tener dos tipos de errores, bien que una locución auténtica sea rechazada, lo que llamaremos falso rechazo *(FAR o False Acceptance Rate)*, o que una locución falsa sea aceptada, lo que llamaremos falsa aceptación (*FRR o False Rejection Rate*).

El valor del umbral influye de forma directa en las tasas de falsa aceptación y falso rechazo. Para un valor muy pequeño pocos intentos de locuciones auténticas serán rechazados, pero un mayor número de locuciones falsas serán aceptadas. Con un umbral muy elevado, decrecerán las falsas aceptaciones a costa de incrementar los falsos rechazos.

El valor de dicho umbral podrá ser fijado *a priori,* si para ello se usa un conjunto de datos distinto del de prueba, o *a posteriori*, si se usa el conjunto de datos de prueba. El establecimiento del umbral estará condicionado a unas especificaciones de un punto de trabajo, generalmente será una de las tres opciones siguientes:

- Un valor especificado de falso rechazo.
- Un valor especificado de falsa aceptación.
- El punto de error igual, EER *(Equal Error Rate)*, que es el punto donde las curvas de falsa aceptación y falso rechazo en función del umbral se cruzan. Este suele ser el punto más popular para caracterizar con un único número el funcionamiento de un sistema. Sin embargo, en sistemas prácticos, no suele ser éste el punto de trabajo más interesante.

En la Figura 16 se muestra gráficamente la tasa de falsa aceptación y falso rechazo explicada anteriormente. La falsa aceptación se corresponde con el área bajo la función densidad de probabilidad de puntuaciones de impostores que queda por debajo del umbral, el falso rechazo con el área bajo la función densidad de probabilidad de puntuaciones de usuarios que queda por encima del umbral.

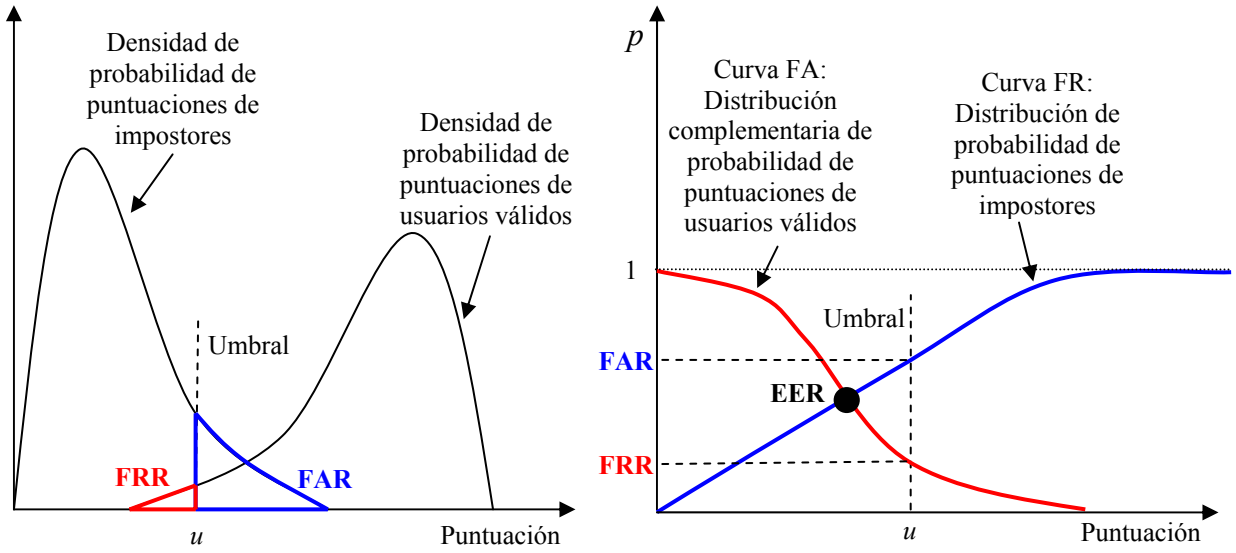


Figura 16. Densidades y distribuciones de probabilidad de usuarios e impostores.

Otra forma de representación son los que conocemos como curvas ROC *(Reciever Operating Curve)*. Este tipo de gráficas se generan representando la FAR frente a (1 - FRR) en función de diferentes valores para el umbral. Una alternativa comúnmente utilizada frente a las curvas ROC, son las curvas DET (*Detection Error Tradeoff*), cuya única diferencia con las ROC es un cambio de escala en los ejes [Martin *et al.,* 1997]. Las curvas DET serán las que se usen en la sección de experimentos para mostrar los resultados de una manera gráfica. La Figura 17 muestra las curvas mencionadas anteriormente, α y β son los correspondientes valores de falso rechazo y falsa aceptación (iguales en el punto EER).

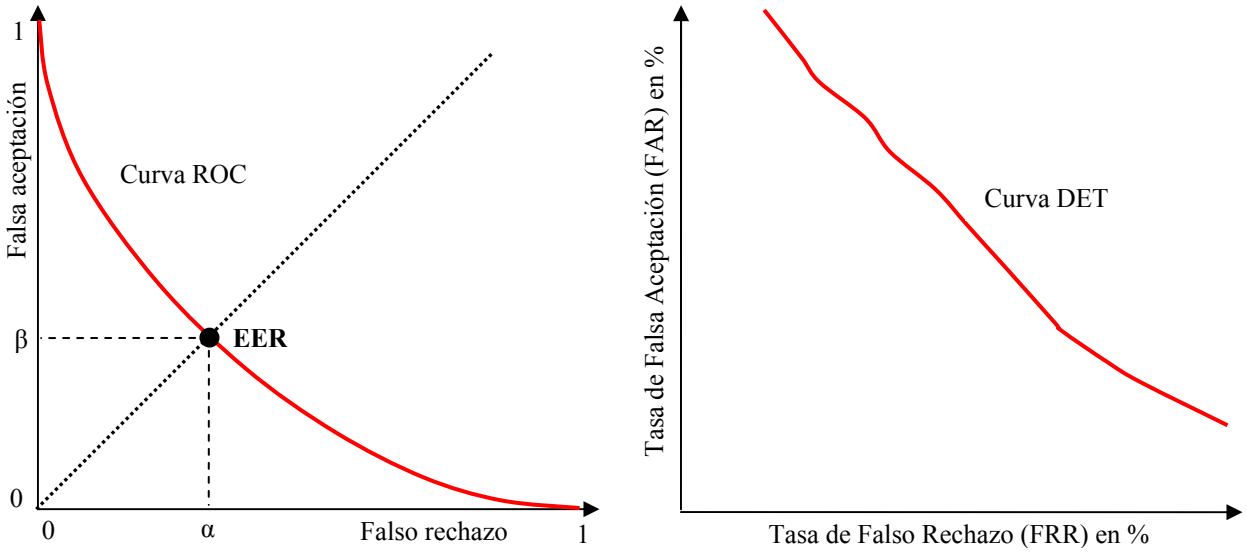


Figura 17. Curva ROC y curva DET

Como ya se ha mencionado la curva DET será la que se adopte en las secciones de experimentos para presentar los resultados de manera gráfica. Junto a cada una de estas curvas se incluirá una tabla con cuatro valores importantes a la hora de evaluar un sistema. Estos valores serán: el DCF *(Detection Cost Function)*, EER, EER por modelo y EER por fichero de test.

Una forma de medir el rendimiento se basa en la función de coste, introducida en el apartado 7.1:

$$C_{DET}(i) = C_{Miss} \cdot P_{Miss|T\arg et} \cdot P_{T\arg et} + C_{FalseAlarm|NonT\arg et} \cdot (1 - P_{T\arg et})$$

En cada evaluación de NIST se proporcionan los costes asociados a falsa aceptación y falso rechazo (C_{Miss} y $C_{FalseAlarm}$), también se establece la probabilidad de que un fichero dado pertenezca al locutor o idioma en cuestión, $P_{T\arg et}$. De esta forma, con el porcentaje de falsa aceptación y falso rechazo, $P_{Miss|T\arg et}$ y $P_{FalseAlarm|NonT\arg et}$, obtenido de nuestro sistema podremos evaluar la función de coste, obteniendo lo que se conoce como DCF.

El EER por modelo se define como el valor teórico que podríamos conseguir sino tuviéramos desalineamiento entre modelos. Este desalineamiento es debido a que no todos los ficheros puntúan en el mismo rango de puntuaciones al enfrentarlos a los distintos modelos, por lo tanto al calcular el EER global del sistema obtendremos un valor mayor. Más adelante, sección 8.9, introduciremos un tipo de normalización que tratará de corregir este desalineamiento, Z-Norm.

El EER por fichero de test es muy similar al EER por modelo, la diferencia radica en que en este caso el desalineamiento se produce por fichero de test. La normalización que mejorará este comportamiento será T-Norm (véase sección 8.9).

8. Resultados reconocimiento biométrico de locutor con SVM

8.1 Introducción

En esta sección se expondrán los resultados obtenidos de la investigación en el campo de reconocimiento biométrico de locutor mediante SVM. En primer lugar se mostrarán dos aspectos importantes del sistema de partida, la configuración y su rendimiento. A continuación se comenzará cambiando la biblioteca utilizada por el sistema, sección 8.3, para continuar con una serie de experimentos orientados mejorar el rendimiento del sistema.

Los experimentos irán dirigidos a probar distintas variables que influyen en el entrenamiento de los modelos, secciones 8.4, 8.5 y 8.7, emplear varios tipos de normalizaciones: tanto de los datos de entrada al sistema, secciones 8.6, 8.8 y 8.12, como de las puntuaciones, sección 8.9 y 8.11. De especial importancia en este informe es la sección 8.10, en la que se realizan pruebas con un tipo de SVM novedoso, épsilon-SVR, y cuyos resultados fueron objeto de una publicación [Lopez-Moreno *et al.,* 2007].

8.2 Sistema de partida

El sistema de partida utilizado para la realización de los trabajos fue el presentado por el grupo ATVS a la evaluación NIST SRE 2006 [NIST SRE]. La configuración del sistema es la siguiente:

Parametrización (véase sección 5):

- 19 coeficientes MFCC + delta con CMN-Rasta-Mapping.
- Filtrado en banda telefónica (300Hz - 3300Hz).
- Vectores extraídos cada 20ms con solapamiento del 50%.

Configuración del sistema SVM:

- Expansión polinómica de tercer grado [Wan y Campbell, 2000].
- Kernel GLDS [Campbell, 2002].

Biblioteca entrenamiento y test:

- Torch [Torch].

Normalización:

- Tnorm [Auckenthaler *et al.,* 2000].

En la Figura 18 se presenta la curva DET del sistema de partida comentado anteriormente, la Tabla 3 muestra los resultados numéricos, con EER 10.5%.

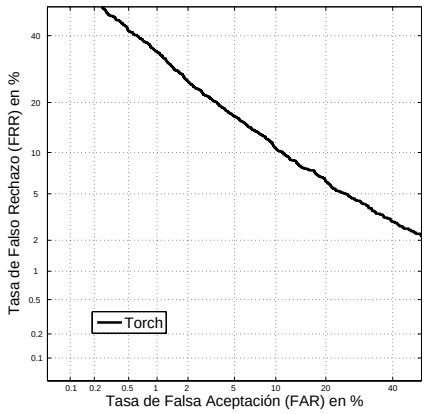


Figura 18. Curva DET del sistema de partida presentado a la evaluación NIST SRE 2006

8.3 Migración a LibSVM

Una vez estudiado el funcionamiento del sistema de partida se abordó el primer objetivo del proyecto, la migración de la biblioteca *Torch* [Torch] a *LibSVM* [LibSVM, 2001]. Estas bibliotecas juegan un papel muy importante dentro del sistema global SVM, se encargan del entrenamiento de los modelos así como de realizar los enfrentamientos entre modelos y ficheros de test.

Torch es un software desarrollado por IDIAP (*Institut Dalle Molle d'Intelligence Artificielle Perceptive*), actualmente bajo licencia BSD (*Berkeley Software Distribution)*. Este tipo de licencia pertenece al grupo de licencias de software libre, con la salvedad de que permite el uso del código fuente en software no libre.

LibSVM forma parte de un proyecto que comenzó a desarrollarse en el año 2000 en la universidad de Taiwan. El código fuente está disponible bajo licencia BSD, al igual que ocurre en la actualidad con Torch.

El motivo principal que llevo a la migración de una biblioteca a otra fue que en el momento de la realización del proyecto, Torch era un software de pago mientras que LibSVM pertenecía al grupo de software bajo licencia BSD. Además, LibSVM ofrece un amplio abanico de opciones, entre ellas podemos destacar:

- Diferentes tipos de SVMs: SVC, υ-SVC, one-class SVM, épsilon-SVR y υ-SVR [Schölkopf *et al.,* 2000].
- Distintos tipos de kernels: lineal, polinómico, radial, sigmoide, etc. [Cristianini y Shawe-Taylor, 2000]
- Posibilidad de entrenar modelos para estimación de probabilidad.

La eficiencia en tiempo y recursos utilizados por cada una de estas dos bibliotecas es bastante similar. En las pruebas de rendimiento realizadas se encontró una ligera ventaja en el uso de Torch frente a LibSVM, los resultados de dichas pruebas se muestran en la Tabla 3.

A continuación se incluye un experimento completo sobre el protocolo de referencia NIST SRE 2006, la tarea a realizar será la 1conv-1conv para género masculino. El experimento tiene por objeto comparar el funcionamiento del sistema con ambas bibliotecas. La comparativa se hará en función del EER, DCF, EER por modelo, EER por fichero de test (véase sección 7.3), tiempo empleado en entrenamiento, tiempo empleado en test y tamaño de los modelos generados.

La descripción completa del experimento es la siguiente, mientras no se diga lo contrario se seguirá usando esta misma configuración:

Evaluación	NIST SRE 2006 masculino		**Normalización**	Ninguna
Tarea	1conv-1conv		**Compensación canal**	Ninguna
Conjunto NonTargets	6500 vectores		**Tipo entrenamiento**	Clasificación (SVC)

Tabla 2. Datos descriptivos del experimento migración Torch LibSVM

Las curvas DET con los resultados se muestran en la Figura 19.

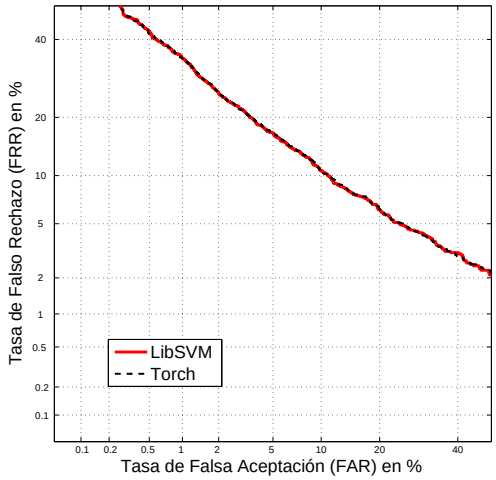


Figura 19. Curva DET del sistema SVM con Torch y con LibSVM

Color	Biblioteca	EER (%)	DCF	EER modelo (%)	EER test (%)
Rojo	LibSVM	10.49	**0.043**	**7.56**	**4.73**
Negro ---	Torch	**10.35**	0.044	7.70	4.80

Tabla 3. Comparación resultados sistema SVM, LibSVM vs. Torch

Los resultados con respecto a eficiencia computacional[1] de ambas bibliotecas se presentan en la siguiente tabla:

Biblioteca	Entrenamiento 5 modelos	Entrenamiento 1 modelo	Test 300 ficheros	Test 1 fichero
LibSVM	8min06s	1min37s	31s	0.104s
Torch	**6min34s**	**1min19s**	**12s**	**0.040s**

Tabla 4. Comparativa eficiencia computacional sistema SVM, biblioteca LibSVM vs. Torch

Como puede extraerse de la Tabla 4 el rendimiento de ambos sistemas es bastante similar, los tiempos empleados en entrenar los modelos varían en 18s (18.5%) mientras que los empleados en los tests lo hacen en 64ms. Estos resultados nos llevan a la conclusión de que el nuevo sistema es capaz de afrontar problemas de una complejidad parecida al anterior. Pongamos el caso de la tarea realizada en este experimento, NIST SRE 2006 género masculino. Se deben entrenar 353 modelos y realizar 23179 enfrentamientos (22131 procesables), el tiempo medio empleado por el sistema será:

$$t = (Num.\mathrm{mod} \times t.\mathrm{mod}) + (Num.enfrent \times t.enfrent) = (353 \times 97) + (22131 \times 0.104) \approx 10 horas$$

Otra variable que debemos tener en cuenta en la migración de una biblioteca a otra es el tamaño que ocupan los modelos en memoria. En la Tabla 5 se muestra una comparativa de dicho tamaño.

[1] La máquina donde fueron medidos los tiempos presenta las siguientes características: Pentium IV (3Ghz), 2Gigabytes de memoria RAM y 1024KB de memoria caché.

Biblioteca	Tamaño modelos
LibSVM	340Kbytes
Torch	**40Kbytes**

Tabla 5. Comparativa tamaño modelos LibSVM y Torch

De la Tabla 5 podemos extraer una diferencia mucho mayor, el tamaño de los modelos generados con LibSVM es un 8.5 veces mayor que los modelos de Torch. Esta diferencia se debe en gran medida al formato en que son guardados los datos, LibSVM lo hace en ASCII mientras que Torch utiliza un formato binario. Podemos encontrar más información acerca de los formatos de salida en [Torch; LibSVM, 2001].

Los tamaños de modelos mostrados en la Tabla 5 hacen referencia a modelos simplificados. La simplificación de los modelos se realiza mediante una función no incluida en la biblioteca. Dicha función aprovecha la característica de linealidad de los vectores soporte por los que está compuesto el modelo, gracias a esta característica podemos agrupar todos los vectores en uno reduciendo considerablemente la cantidad de información a almacenar.

8.4 Influencia de la variable coste en el entrenamiento

Una vez adaptada la nueva biblioteca al banco de pruebas existente se realizaron varios experimentos para ajustar distintas variables, en esta sección describiremos los experimentos orientados a ajustar la variable *coste*.

El coste en el entrenamiento (ver sección 6) es una variable mediante la cual controlamos la penalización aplicada a una muestra incorrectamente clasificada a la hora de establecer el hiperplano de separación entre las clases. En secciones sucesivas veremos la influencia de aplicar costes distintos a las clases de problema, la forma de llevarlo a cabo será a través del valor de las etiquetas *Target* y *NonTarget*.

La expansión polinómica de tercer orden, ilustrada en la Figura 13, aplicada a vectores de 38 coeficientes da como resultado vectores de 9880 dimensiones [Wan y Campbell, 2000]. En problemas donde el número de vectores sea inferior al número de dimensiones siempre es posible encontrar un plano lineal que separe las muestras de las dos clases, por lo tanto el algoritmo no debería tener problemas en la clasificación de las muestras (construcción del hiperplano). En nuestro caso particular, el conjunto de datos de entrenamiento no supera en número a las dimensiones (6500 < 9880), por lo que el coste no debería influir en los resultados ya que todas las muestras estarán clasificadas correctamente y no será necesario aplicarlas ninguna penalización.

Los resultados se muestran en la Figura 20 en forma de curva DET y en la Tabla 6 con valores numéricos.

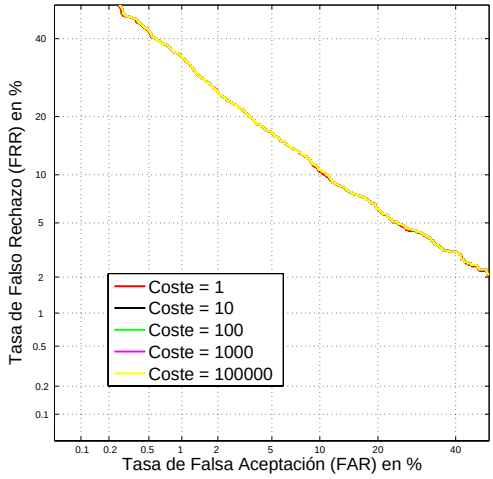


Figura 20. Curva DET del sistema SVM con distintos costes de entrenamiento

Color	Coste	EER (%)	DCF	EER modelo (%)	EER test (%)
Rojo	1	**10.29**	**0.043**	**7.56**	**4.68**
Negro	10	10.49	**0.043**	**7.56**	4.73
Verde	100	10.49	**0.043**	**7.56**	4.73
Rosa	1000	10.49	**0.043**	**7.56**	4.73
Amarillo	100000	10.49	**0.043**	**7.56**	4.73

Tabla 6. Comparación resultados sistema SVM con distintos costes de entrenamiento

Una observación importante a tener en cuanta es que aunque el coste no influye en los resultados si lo hace en el tiempo de entrenamiento. A mayor coste mayor tiempo se emplea en el entrenamiento de los modelos, podemos ver esta afirmación reflejada en la Tabla 7. La tendencia se mantiene si seguimos aumentando el coste por lo que interesará utilizar el menor posible.

Coste	Tiempo entrenamiento 1 modelo
1	**47s**
100	1.16min

Tabla 7. Comparación tiempos entrenamiento del sistema SVM con distintos costes

8.5 Coste de la clase NonTarget

En la sección anterior vimos la influencia de la variable coste en los resultados, en este apartado nos centraremos en el valor de la etiqueta de los vectores NonTargets, que es una forma de dar más o menos peso a las muestras pertenecientes a esa clase en el entrenamiento. En la sección 6 se explicó el comportamiento de los sistemas basados en clasificación, mostrando la influencia de esta variable en la construcción del hiperplano de separación. Con estos experimentos probaremos si efectivamente esta variable influye en los resultados obtenidos por el sistema.

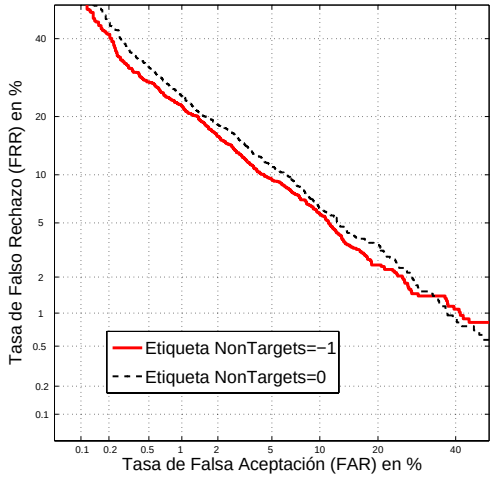


Figura 21. Curva DET del sistema con distintos valores de etiqueta NonTargets

Color	Etiqueta NonTargets	EER (%)	DCF	EER modelo (%)	EER test (%)
Rojo	-1	**7.54**	**0.031**	**4.59**	**2.33**
Negro ---	0	8.08	0.034	5.17	2.73

Tabla 8. Comparación resultados etiqueta NonTargets

A la vista de los resultados se concluye que el valor de etiqueta -1 obtiene unos resultados sustancialmente mejores que el valor 0. Esto es debido a la propia definición de la función de pérdidas del SVM.

$$f_{pérdidas}(\vec{x}_i) = \max\{0, 1 - y_i f(\vec{x}_i)\}$$

Siendo y_i el valor de la etiqueta y $f(\vec{x}_i) =< \vec{w}, \vec{x}_i > +b$ la función que mide la distancia del vector $\vec{x}_i$ al hiperplano $\vec{w}$.

Uno de los criterios del SVM es minimizar dicha función, por lo tanto con un valor de etiqueta 0 la función de pérdidas valdrá siempre 1, por el contrario con un valor de etiqueta -1 la función de pérdidas será $f_{pérdidas}(\vec{x}_i) = \max\{0, 1 + f(\vec{x}_i)\}$

8.6 Escalado de los datos de entrada

En esta sección se abordará el escalado de los datos utilizados para el entrenamiento y test de los modelos de la evaluación. El escalado de datos es una opción recomendada encarecidamente por los creadores de la biblioteca, el objetivo principal de esta técnica es expandir (en el caso de que el rango de los datos sea muy pequeño) o contraer (en el caso contrario) el margen de variación de los datos. De esta forma obtendríamos una distribución de los datos más homogénea, la cual conllevaría una mejor y más rápida construcción del hiperplano de separación, con la correspondiente mejora de los resultados.

Los resultados obtenidos no fueron los esperados, en la Figura 22 se puede observar el comportamiento del sistema tras el escalado, los datos escalados empeoran sensiblemente el rendimiento del sistema. A la vista del comportamiento del sistema se descarto la opción de escalar los datos para el resto de las pruebas.

Algo a tener en cuenta en el escalado de datos es que los vectores de test se deben escalar en el mismo rango que los datos utilizados para entrenar el modelo correspondiente. Esto trae consigo tanto un gran coste computacional como un mayor uso de memoria, ya que hay que guardar los rangos de escalado de todos y cada uno de los modelos.

Por cuestiones de eficiencia sólo se entrenaron los modelos con 100 vectores de *NonTargets*, las características del experimento se resumen en la Tabla 9.

Evaluación	NIST SRE 2006 masculino		**Normalización**	Ninguna
Tarea	1conv-1conv		**Compensación canal**	Ninguna
Conjunto NonTargets	100 vectores		**Tipo entrenamiento**	Clasificación (SVC)

Tabla 9. Datos descriptivos del experimento de escalado

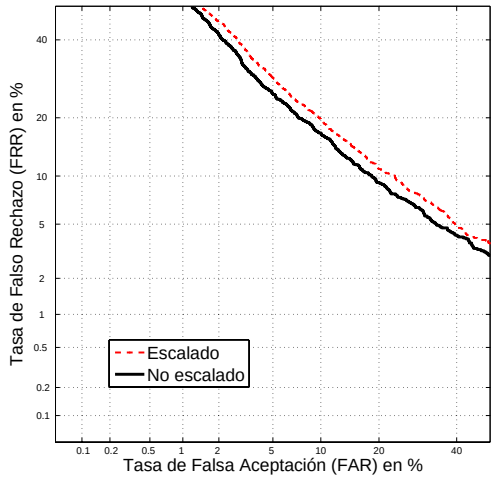


Figura 22. Curva DET del sistema escalado

Color	Datos	EER (%)	DCF	EER modelo (%)	EER test (%)
Rojo ---	Escalados	14.7	0.063	11.7	8.0
Negro	No escalados	**13.2**	**0.060**	**10.5**	**6.3**

Tabla 10. Comparación resultados escalado

El EER del sistema no escalado con respecto al escalado es un 9.8% mayor, en DCF podemos observar una tendencia similar, siendo en este caso el incremento de un 4.8%.

8.7 Conjunto de datos de impostores, NonTargets

El conjunto de *NonTargets* está formado por todos aquellos vectores correspondientes a locuciones de impostores. Consideramos como impostores a todos los usuarios distintos de los existentes en el grupo que tengamos que reconocer, es decir, podemos usar tantos impostores para entrenar el sistema como queramos siempre y cuando no consideremos como impostores a otros usuarios de dicha evaluación.

Este tipo de experimentos fueron orientados a conseguir un conjunto de datos de desarrollo eficiente. El objetivo principal del conjunto de datos de desarrollo es recoger la mayor variabilidad posible en base a diferentes criterios: locutores, bases de datos, micrófonos, idiomas…

Con una buena selección de este conjunto conseguiremos dos objetivos importantes para el sistema. Por un lado reducir el tiempo de entrenamiento de los modelos ya que cuantos menos vectores contenga el conjunto de *NonTargets* menos tiempo empleará el algoritmo en encontrar el hiperplano óptimo de separación. Por otro lado, la selección de un conjunto con una mayor variabilidad nos llevará a crear modelos más representativos de los usuarios, esto debería traducirse en una mejor identificación del usuario con un EER global menor para el sistema.

Se pasó de un conjunto de 6500 vectores *NonTargets* a uno de 3841, lo que supone una reducción del 40%. Las bases de datos utilizadas en este nuevo conjunto de datos fueron:

- Switchboard I [LDC].
- Switchboard II [LDC].
- NIST SRE 2004 [NIST SRE].

A continuación se presentan los resultados obtenidos, Figura 23 y Tabla 11:

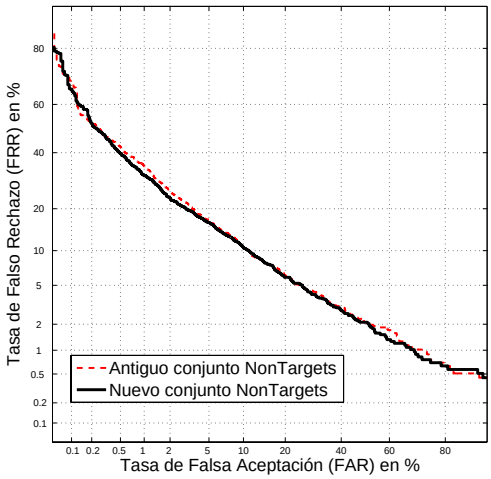


Figura 23. Curva DET del sistema con el nuevo conjunto de NonTargets

Color	Conjunto NonTargets	EER (%)	DCF	EER modelo (%)	EER test (%)
Rojo ---	Antiguo	10.49	0.043	7.56	**4.73**
Negro	Nuevo	**10.42**	**0.041**	**7.52**	4.92

Tabla 11. Comparación resultados conjunto NonTargets

Como puede verse en la Tabla 11 la eficiencia del sistema mejora ligeramente con el nuevo conjunto de *NonTargets*, donde apreciamos una mayor influencia del nuevo

conjunto es en el tiempo de entrenamiento[2], la Tabla 12 muestra una reducción del tiempo de entrenamiento por modelo de casi un 50%. Esto hace que el sistema sea mucho más eficiente y competitivo a la hora de enfrentarlo a las diversas pruebas.

Conjunto NonTargets	Tiempo entrenamiento un modelo (s)
Antiguo	93
Nuevo	**49**

Tabla 12. Comparativa tiempos entrenamiento modelos con los distintos conjuntos de NonTargets

8.8 Normalización de rango, Rank Normalization

Este tipo de normalización permite una distribución uniforme de los vectores por el espacio de características, lo que facilita el modelado de los usuarios. Su base teórica se explica en detalle en [Stolcke *et al.,* 2005].

La implementación de esta normalización se realizó de forma que resultara lo más eficiente posible, para ello se descompuso la normalización en dos tareas principales:

1. La primera de ellas consistía en construir un matriz de referencia partiendo de unos vectores dados, esta matriz se ordenaría mediante el método "QuickSort". La matriz de referencia permanece almacenada en el sistema evitando tener que volver a construir una nueva cada vez que se quiera utilizar el banco de pruebas.

2. La segunda tarea es la de normalizar los vectores. Se enfrentan los vectores uno a uno a la matriz de referencia, buscando la posición que ocuparía cada componente del vector en la matriz de referencia. Para dicha tarea se usa la técnica de búsqueda "búsqueda binaria" cuya complejidad computacional se reduce a O(log N).

La matriz de referencia se construyó partiendo de todos los vectores *NonTargets* del sistema. Posteriormente, en la segunda fase, se normalizaron todos los vectores, tanto los usados para entrenamiento como los usados para test.

Los resultados obtenidos se presentan en la Tabla 13 y la Figura 24.

Color	Normalización	EER (%)	DCF	EER modelo (%)	EER test (%)
Rojo ---	Rank-Normalization	12.13	0.047	9.1625	6.57
Negro	RAW	**10.42**	**0.041**	**7.5156**	**4.92**

Tabla 13. Comparación resultados Rank-Normalization

A la vista de los resultados se puede observar como esta nueva técnica de normalización probada no presenta un buen comportamiento, al menos con el tipo de datos que se

[2] La máquina donde fueron medidos los tiempos presenta las siguientes características: Pentium IV (3Ghz), 2Gigabytes de memoria RAM y 1024KB de memoria caché.

manejan en este proyecto. Dada su ineficiencia se descartará para el resto de pruebas que se sucedan en el presente informe.

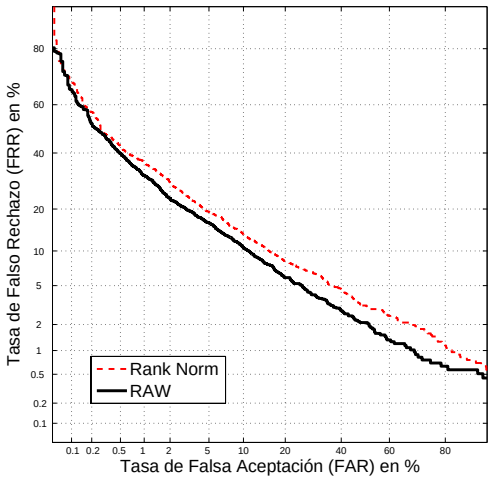


Figura 24. Curva DET del sistema con normalización de datos Rank-Normalization

8.9 Normalización de puntuaciones, T-Norm y Z-Norm.

La normalización de puntuaciones [Bimbot *et al.,* 2004] se emplea para realizar escalado de distribuciones y eliminar ciertas dependencias (como por ejemplo del canal) contribuyendo de ese modo a la robustez del sistema. La normalización de puntuaciones es además importante de cara a la fusión debido a que los sistemas a fusionar pueden no ser homogéneos (uno puede medir similitudes y otro distancias). Así pues, la normalización ayuda a transformar las puntuaciones generadas por los sistemas individuales de modo que se sitúen en un rango homogéneo. Adicionalmente situar las puntuaciones en rangos comparables permite utilizar un umbral por sistema en lugar de usar uno específico para cada locutor.

La normalización T-Norm *(Test Normalization)* [Auckenthaler *et al.,* 2000] es un procedimiento de escalado de la distribución de puntuaciones, pero en lugar de centrarse en el modelo y en cómo se comporta ante datos de otros locutores, Z-Norm *(Zero Normalization)*, se centra en el fichero de test y en su comportamiento frente a otros modelos, evitando de este modo posibles desajustes entre los ficheros para entrenamiento de la normalización y el fichero de test actual. Dado un fichero de test, éste se enfrenta al modelo del usuario bajo estudio, pero también a una cohorte de modelos de otros locutores (impostores), obteniendo un conjunto de puntuaciones con las que estimar una media y una varianza. A cada puntuación se le resta esta media y se divide entre la raíz cuadrada de la varianza.

La normalización Z-Norm [Auckenthaler *et al.,* 2000] se realiza de una manera muy similar a la explicada para el caso de T-Norm, la diferencia radica en que es el modelo el que se enfrenta a una cohorte de ficheros de test. Con el conjunto de puntuaciones obtenido se estima la media y la varianza, para posteriormente normalizar las puntuaciones siguiendo el mismo procedimiento aplicado en T-Norm.

A la hora de aplicar T-Norm la selección de la cohorte de modelos es un elemento importante y sujeto a investigación. Estos modelos han de ser lo más parecidos como

sea posible a los modelos de usuario, y su número ha de ser relativamente elevado (cuantos más mejor), ya que debemos estimar una gausiana (media y varianza) a partir de las puntuaciones obtenidas. En los resultados que se muestran en la Tabla 14 se usaron los modelos de la evaluación NIST SRE 2005 para construir la cohorte de T-Norm.

Al igual que sucedía con la cohorte de T-Norm la selección de la cohorte de ficheros necesaria para Z-Norm es también muy importante. Los ficheros seleccionados para este fin fueron los correspondientes a los datos de test de NIST SRE 2005.

Color	Normalización	EER (%)	DCF	EER modelo (%)	EER test (%)
Rojo ---	raw	10.42	0.041	7.52	4,92
Negro	T-Norm	10.42	**0.037**	**6.98**	4.92
Verde …	Z-Norm	**10.29**	0.042	7.52	**4.24**

Tabla 14. Comparación resultados normalización puntuaciones, T-Norm y Z-Norm

En la Figura 25 puede verse como la curva DET del sistema cuyas puntuaciones han sido normalizadas mediante T-Norm presenta un mejor comportamiento que el sistema sin normalizar, raw. Esta diferencia es más notable en la zona donde la falsa aceptación es menor.

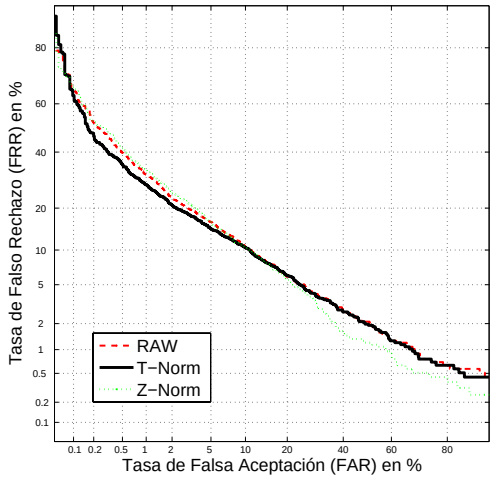


Figura 25. Curva DET del sistema con normalización de puntuaciones T-Norm y Z-Norm

8.10 SVM épsilon-SVR

Hasta ahora los resultados mostrados hacen referencia a experimentos basados en el tipo de SVM SVC (*Support Vector Classification*), en este apartado se mostrará una comparativa entre los distintos tipos de SVM posibles de entre los que destacaremos el épsilon-SVR *(ε-Support Vector Regresión)*. La explicación detallada de cada uno de estos tipos de SVM, así como la del sistema épsilon-SVR implementado puede encontrarse en la sección 6.

La verificación de locutor es básicamente un problema binario de clases, los impostores pertenecerán a una y el usuario a otra, por este motivo la mayor parte de los esquemas basados en SVM-GLDS utilizan la clasificación (SVC) en lugar de la regresión (SVR) a

la hora de entrenar los modelos. El objetivo de la clasificación es calcular la clase a la que pertenece cada una de las muestras extraídas de los datos. La regresión representa una aproximación más general, su objetivo es encontrar una aproximación a la función de las características [Schölkopf *et al.,* 2000].

En la Figura 26 se muestra una primera comparativa del mismo sistema funcionando con cuatro tipos distintos de SVM. De la curva DET y la Tabla 15 se puede extrapolar fácilmente como el sistema con regresión funciona mejor que cualquier otro. Estos resultados confirman las suposiciones iniciales acerca de la diferencia entre clasificación y regresión, con el mejor funcionamiento de esta última.

Color	Tipo SVM	EER (%)	DCF	EER modelo (%)	EER test (%)
Rojo ---	SVC	10.42	0.040	7.52	4.92
Negro	One-class	48.76	0.100	49.22	74.54
Verde	épsilon-SVR	**7.54**	**0.031**	**4.59**	**2.33**
Rosa …	nu-SVR	10.88	0.043	8.01	5.13

Tabla 15. Comparación resultados distintos tipos de SVM

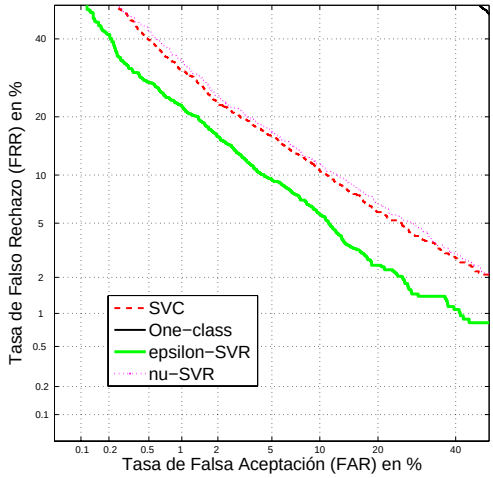


Figura 26. Curva DET del sistema con distintos tipos de SVM

Con la intención de comparar la eficiencia computacional (el ordenador donde fueron realizados los experimentos es el mismo mostrado en pruebas de eficiencia anteriores) de los tipos de SVM analizados, se adjunta la Tabla 16 que contiene los tiempos empleados en el entrenamiento de un modelo. El tiempo empleado en los tests sigue la misma tendencia.

Tipo SVM	Tiempo entrenamiento un modelo (s)
SVC	**49**
One-class	810
épsilon-SVR	120
nu-SVR	1080

Tabla 16. Comparativa tiempos entrenamiento modelos con los distintos tipos de SVM

El tipo de SVM más eficiente computacionalmente hablando es el SVC, en el caso del épsilon-SVR el tiempo de entrenamiento asciende a 2 minutos, lo que supone más del doble de tiempo empleado por SVC. Si se tienen en cuenta los resultados de la Tabla 15 puede verse como el sistema basado en épsilon-SVR presenta un resultado en EER un 27.6% mejor que el sistema basado en SVC. La afirmación anterior, junto con la observación de que los 2 minutos requeridos para el entrenamiento mediante épsilon-SVR son afrontables por el sistema nos llevan a la siguiente conclusión; el sistema SVM-GLDS basado en regresión se presenta como un buen candidato para sustituir al sistema basado en clasificación, sus resultados son sensiblemente mejores y los tiempos empleados aceptables.

Al igual que se hizo en la sección 8.4 para SVC, se mostrará en la Tabla 17 los resultados obtenidos al realizar un par de pruebas con distinto coste de entrenamiento en regresión. El objetivo de las pruebas es comprobar la influencia de esta variable en el sistema, como ya vimos era la que daba más o menos peso a las muestras penalizadas.

Coste	EER (%)	DCF	EER modelo (%)	EER test (%)
10	7.54	0.031	4.59	**2.33**
1000	**7.45**	**0.031**	**4.58**	**2.33**

Tabla 17. Comparación resultados coste entrenamiento en regresión

El tiempo empleado en entrenar los modelos sigue la misma tendencia explicada anteriormente en clasificación, la Tabla 18 muestra los tiempos empleados por el sistema en entrenar un modelo.

Coste	Tiempo entrenamiento un modelo (s)
10	**120**
1000	210

Tabla 18. Comparativa tiempos entrenamiento modelos regresión con distinto coste

Continuando con el estudio del coste de entrenamiento veremos que existe la posibilidad de aplicar un coste distinto a cada una de clases del problema. Parece lógico pensar que clasificar incorrectamente el vector correspondiente al usuario es más grave que clasificar incorrectamente un vector de impostor, de usuario tenemos tan sólo un vector ya que las pruebas realizadas son de la tarea 1conv-1conv de NIST [NIST SRE], mientras que de impostor tenemos los 3841 vectores del conjunto de *NonTargets*.

Basándonos en este hecho se realizó una prueba en la que el coste para la clase NonTargets era de 10 mientras que para la clase de Targets era de 1000, 100 veces superior. Los resultados (con T-Norm) se muestran en la Tabla 19.

Coste clase NonTargets	Coste clase Target	EER (%)	DCF	EER modelo (%)	EER test (%)
10	1000	**6.78**	**0.029**	**4.17**	**2.33**
10	10	6.89	**0.029**	**4.17**	**2.33**

Tabla 19. Comparación resultados coste entrenamiento en regresión

Podemos apreciar una mínima mejora en el comportamiento del sistema en EER y DCF, la mejora es de tan sólo el 1.5% en EER y el 0.4% en DCF. Estos resultados nos llevan a la misma conclusión que los de la Tabla 17, la mayor parte de los modelos se entrenan correctamente, por lo tanto la penalización a las muestras incorrectamente clasificadas no tiene una gran influencia en los resultados.

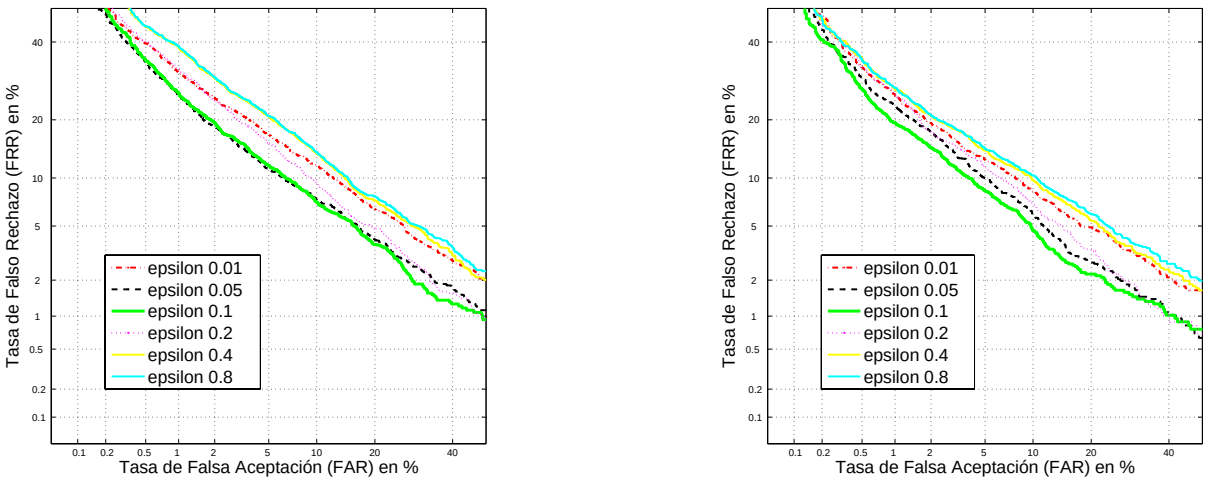
Para concluir el estudio del tipo de SVM épsilon-SVR se mostrará la influencia del parámetro ε (véase sección 6) en el comportamiento del sistema [Smola y Schoelkopf, 1998]. La comparativa se hará en base a su comportamiento en EER, DCF_{min}, EER por modelo y EER por fichero de test para ambos géneros, masculino Tabla 20 y femenino Tabla 21. Los resultados presentados proceden de sistemas en los que se ha aplicado una normalización de las puntuaciones T-Norm.

Color	Valor de ε	EER (%)	DCF	EER modelo (%)	EER test (%)
Rojo -.-	0.01	9.08	0.035	6.27	3.74
Negro ---	0.05	7.78	0.032	4.93	**2.25**
Verde	0.1	**6.89**	**0.029**	**4.17**	2.33
Rosa …	0.2	8.39	0.035	5.34	3.07
Amarillo	0.4	9.88	0.037	6.93	4.50
Azul	0.8	10.29	0.037	6.98	4.86

Tabla 20. Comparación resultados distintos valores ε, male

Color	Valor de ε	EER(%)	DCF	EER modelo (%)	EER test (%)
Rojo -.-	0.01	11.02	0.041	7.62	5.69
Negro ---	0.05	8.56	**0.035**	**5.51**	**3.49**
Verde	0.1	**8.47**	0.035	5.72	**3.49**
Rosa …	0.2	9.74	0.042	7.05	3.95
Amarillo	0.4	11.94	0.048	8.99	6.10
Azul	0.8	12.00	0.048	9.05	6.26

Tabla 21. Comparación resultados distintos valores ε, female



a) **b)**

Figura 27. Curvas DET del sistema SVM épsilon-SVR, para género masculino a) y femenino b)

Se puede ver como el valor de ε influye en gran medida en los resultados, tanto para el género masculino como para el femenino. Los mejores resultados se alcanzan con el valor de ε de 0.1, a medida que nos alejamos de este valor los resultados empeoran. Otra de las cosas a tener en cuenta es el tiempo empleado en el entrenamiento, a medida que el valor de ε se hace más pequeño los tiempos se incrementan sensiblemente. La Tabla 22 muestra una comparativa de los tiempos de entrenamiento para los valores de épsilon seleccionados.

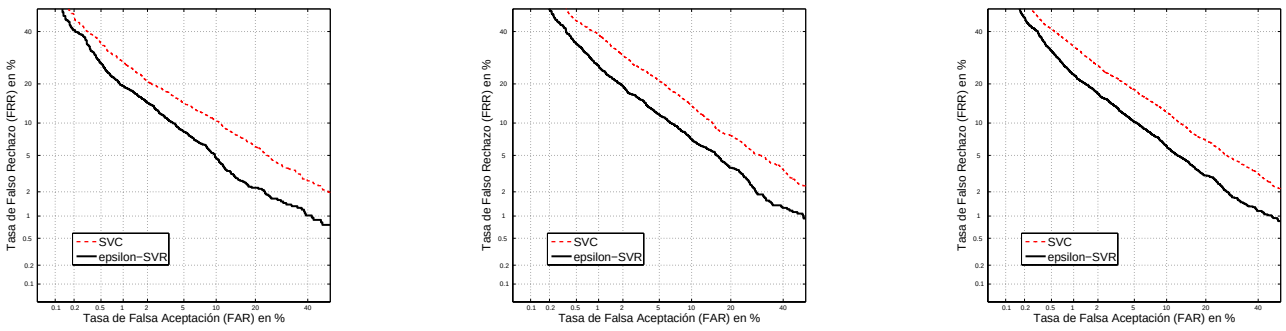
Valor de ε	Tiempo entrenamiento un modelo
0.01	20m40s
0.05	7m30s
0.1	2m
0.2	1m50s
0.4	1m
0.8	**42s**

Tabla 22. Comparativa tiempos entrenamiento modelos regresión con distinto valor de ε

En la Tabla 22 queda probado lo anteriormente citado, cuanto más restrictivo sea el valor de ε más tiempo le lleva al sistema entrenar los modelos. Si nos fijamos en el valor óptimo ε=0.1 vemos que los modelos tardan unos 2 minutos en entrenarse, pero para el valor de ε de 0.01 el tiempo asciende a casi 21 minutos. Este valor de tiempo es muy elevado y hace que el sistema completo sea muy lento lo que da lugar a que la realización de pruebas no se complete con la fluidez necesaria.

Esta diferencia de tiempos es debida a que cuanto menor sea el valor de ε más muestras tendrá en cuenta el sistema a la hora de entrenar los modelos, estas muestras se penalizarán con un valor proporcional a la distancia respecto al hiperplano. Uno de los objetivos de está técnica a la hora de buscar el hiperplano óptimo es minimizar estas distancias, por tanto cuantas más tengamos en cuenta más tardará el algoritmo.

Para concluir este apartado se presentará una comparativa entre el sistema de base, SVM-GLDS con SVM del tipo SVC, y el sistema mejorado, SVM-GLDS con SVM del tipo ε-SVR. La comparación se hará para ambos géneros por separado así como para la fusión de ambos.



a) **b)** **c)**

Figura 28. Curvas DET del sistema SVM épsilon-SVR y SVM SVC, para género masculino a), femenino b) y fusión de géneros c)

Figura	Color	Género	Tipo SVM	EER (%)	DCF	EER modelo (%)	EER test (%)
a	Rojo ---	Masculino	SVC	10.42	0.037	6.98	4.92
a	Negro	Masculino	épsilon-SVR	**6.89**	**0.029**	**4.17**	**2.33**
b	Rojo ---	Femenino	SVC	11.98	0.048	9.05	6.25
b	Negro	Femenino	épsilon-SVR	**8.47**	**0.035**	**5.72**	**3.31**
c	Rojo ---	Fusión	SVC	11.28	0.043	9.91	7.29
c	Negro	Fusión	épsilon-SVR	**7.80**	**0.033**	**6.31**	**3.93**

Tabla 23. Comparación resultados sistema SVM-GLDS SVC y sistema SVM-GLDS épsilon-SVR, para género masculino, femenino y fusión de ambos

Como ya se afirmó al comienzo de este apartado el sistema basado en regresión presenta unos resultados sensiblemente mejores al sistema basado en clasificación. En la Figura 28 podemos apreciar una mejora significativa en todos los puntos de operación de la curva DET para cualquier género. De la Tabla 23 podemos extrapolar unas mejoras del 34% para el género masculino y del 29% para el género femenino en términos de EER, en el caso del DCF las mejoras suponen un 22% y 25% para los casos masculino y femenino respectivamente.

Parte de este trabajo ha sido recogido en un artículo presentado a Interspeech 2007 bajo el título "Support Vector Regresión for Speaker Verification", [Lopez-Moreno *et al.,* 2007]. El artículo mencionado se incluye en el apéndice.

8.11 Estimación de probabilidad

En los apartados anteriores a este se presentaban resultados obtenidos con modelos creados para obtención de una puntuación. La función de test de la biblioteca LibSVM devolvía en un principio el valor de la etiqueta a la que pertenecía el fichero testeado. Suponiendo que las etiquetas fueran 1 y -1 para la clase de *Targets* y *NonTargets* respectivamente, la biblioteca devolvía un 1 en caso de que el vector enfrentado al modelo de usuario perteneciera a ese usuario y -1 en caso contrario. Este comportamiento se modificó a nivel de código para que devolviera una puntuación, de esta manera cabía la posibilidad de establecer umbrales de decisión y controlar de una manera más exhaustiva el comportamiento del sistema.

En este apartado se probará una opción de la biblioteca con la cual los modelos son entrenados para devolver una probabilidad. En cierto sentido se podría decir que esta probabilidad es como la puntuación que devolvía anteriormente, lo que distingue esta probabilidad de la puntuación es que la probabilidad está normalizada, es decir, estará entre 0 y 1. Gracias a esta normalización todas las puntuaciones se situarán en un rango homogéneo, por lo tanto el sistema será capaz de distinguir usuarios de impostores con una mayor facilidad.

A continuación se presenta el comportamiento del sistema usando un tipo de SVM ε-SVR. Con el fin de realizar las pruebas de la manera más general posible, los resultados aparecerán normalizados mediante T-Norm y sin normalizar *(Raw)*.

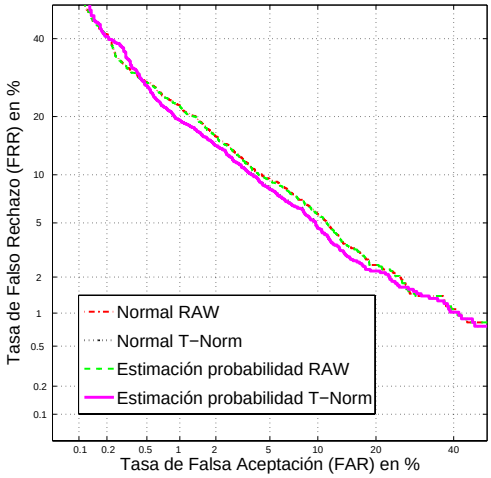


Figura 29. Curvas DET del sistema con modelos para estimación de probabilidad

Modelos	Normalización puntuaciones	EER (%)	DCF	EER modelo (%)	EER test (%)
Normal	RAW	7.54	0.031	4.59	**2.33**
Estimación probabilidad	RAW	7.45	0.031	4.58	**2.33**
Normal	T-Norm	6.89	**0.029**	**4.17**	**2.33**
Estimación probabilidad	T-Norm	**6.78**	**0.029**	**4.17**	**2.33**

Tabla 24. Comparación resultados del sistema con modelos creados para estimación de probabilidad

De la Tabla 24 se pueden extraer las diferencias entre el entrenamiento normal de los modelos y el entrenamiento para la obtención de la probabilidad. Si nos fijamos en los valores más bajos de la tabla, en rojo y negrita, nos daremos cuenta de que el EER para el modelo con puntuaciones T-normalizadas está en 6.78%, lo que supone una mejora del 1.5%.

Se debe tener en cuenta que el entrenamiento se ralentiza, llegando a cuadriplicarse, es decir la media de 2 minutos por modelo pasa a ser ahora de 8 minutos.

8.12 Compensación de variabilidad intersesión: NAP

La compensación de la variabilidad intersesión es un punto clave, dada la alta variación que la señal de voz sufre debido a múltiples factores (canal, entorno, etc.). Esquemas de compensación [Solomonoff *et al.,* 2004; Solomonoff *et al.,* 2005] intentan compensar esta variabilidad y resultan muy interesantes en el campo de los SVMs.

En la compensación de canal podemos distinguir dos pasos importantes, por un lado la creación de una matriz de referencia que usaremos para compensar los vectores y por otro lado la compensación de los vectores que emplearemos en los experimentos. La matriz de referencia deberá contener la mayor cantidad de datos posible, cuantos más usuarios y más locuciones por usuario mejor se recogerá la variabilidad intersesión.

Para la creación de la matriz de referencia se usaron ficheros de las bases de datos de NIST SRE 2004 y NIST SRE 2005. Estos ficheros son distintos de los usados en los conjuntos de desarrollo y prueba. Con la intención de comprobar la influencia de la matriz de referencia en la compensación se crearon dos matrices de referencia distintas:

- Matriz de referencia 1: compuesta por 220 locutores, 110 procedentes de la base de datos NIST SRE 2004 y 110 de NIST SRE 2005 y un total de 3449 locuciones. En la base de datos NIST SRE 2005 tenemos locutores con un máximo de 38 locuciones y con un mínimo de 11 locuciones. En NIST SRE 2004 los locutores varían entre 24 y 8 locuciones.

- Matriz de referencia 2: para la construcción de esta matriz se usaron todos los datos disponibles de las evaluaciones NIST SRE 2004 y 2005. El número total de locutores asciende a 360, 136 procedentes de la base de datos NIST SRE 2004 y 224 de NIST SRE 2005. Existen 4277 locuciones en total, llegando a tener locutores con un máximo de 38 locuciones para NIST SRE 2005 y 24 para NIST SRE 2004, el número mínimo de locuciones fue 1 para ambas bases de datos.

Los ficheros a compensar fueron los correspondientes a los datos de desarrollo y test. Los datos de desarrollo son los mencionados en el punto 8.7, página 51, los datos de test son los correspondientes a la evaluación NIST SRE 2006, cuyo protocolo de evaluación está siendo usado para las pruebas.

En los experimentos además de probar distintas configuraciones de la matriz de referencia se probaron distinto número de dimensiones a eliminar (compensar). Con ello se intentaba buscar la configuración óptima para el reconocimiento de locutor partiendo del diseño de experimentos que se tenía en aquel momento.

La Tabla 25 muestra los resultados del sistema utilizando para la compensación la primera matriz de referencia y distinto número de dimensiones a compensar. De esta tabla se puede extraer que los mejores resultados en EER y DCF se obtienen con 40 y 60 dimensiones respectivamente. En la Tabla 26 se compararán los resultados de compensar 40 y 60 dimensiones con las dos posibles matrices de referencia, además se añadirán los resultados del sistema sin compensación de variabilidad intersesión, de esta forma podremos apreciar la ganancia.

A la vista de los resultados mostrados en la Tabla 25 podemos concluir que el número de dimensiones compensadas influye mínimamente en los resultados. Se podría decir que existe un valor óptimo de dimensiones a compensar, 40, un valor mucho más bajo, 20, obtiene peores resultados debido a que no elimina todo el ruido posible. Por el contrario un valor mucho más alto, 64, obtiene unos resultados peores debido a que elimina el ruido y parte de la información necesaria para la identificación del locutor.

La Figura 30 muestra la curva DET del sistema original, sin NAP, y del sistema compensado con las dos matrices de referencia mencionadas (40 dimensiones compensadas). La mejora aportada por NAP es más notable en la zona donde FAR = FRR (EER) de la curva, si se compara el sistema con y sin compensación observaremos una mejora del 12.6% en EER.

Dimensiones eliminadas	EER (%)	DCF	EER modelo (%)	EER test (%)
20	6.54	0.029	3.93	2.08
25	6.49	0.029	4.11	2.05
30	6.40	0.029	4.12	2.00
35	6.08	0.029	4.07	2.03
40	**6.03**	0.029	3.88	1.91
45	6.21	0.029	3.97	1.90
50	6.08	0.028	3.90	**1.85**
55	6.15	0.028	4.00	2.00
60	6.14	**0.028**	**3.86**	2.02
64	6.27	0.028	3.92	2.00

Tabla 25. Comparación resultados compensación de variabilidad intersesión, NAP, matriz referencia 1 con distintas dimensiones a compensar

Sistema	Dimensiones eliminadas	EER (%)	DCF	EER modelo (%)	EER test (%)
Sistema sin compensar	-	6.89	0.029	4.17	2.33
Matriz de referencia 1	40	6.03	0.029	3.88	**1.91**
	60	6.14	**0.028**	**3.86**	2.02
Matriz de referencia 2	40	**6.02**	0.029	3.97	1.92
	60	6.22	0.028	3.94	1.96

Tabla 26. Comparación resultados compensación de variabilidad intersesión, NAP, matriz referencia 2 con distintas 40 y 60 dimensiones

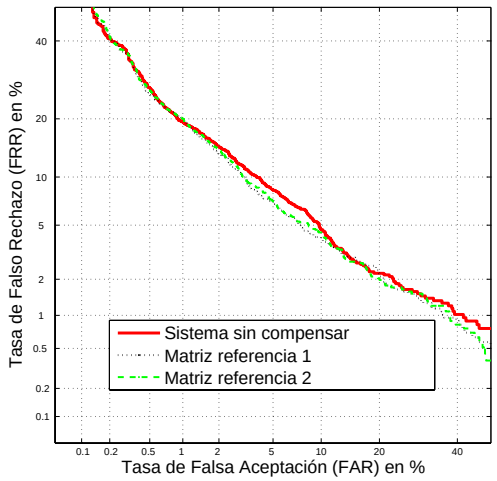


Figura 30. Curvas DET del sistema con compensación de variabilidad intersesión, se muestran los resultados con las dos matrices de referencia y con 40 dimensiones compensadas

9. Resultados reconocimiento biométrico de locutor con SuperVectors (GMM-SVM)

9.1 Introducción

En esta sección seguimos abordando el mismo problema que en la anterior, el reconocimiento biométrico de locutor, la diferencia radica en la técnica empleada para llevar a cabo tal fin. Mientras que en la sección 8 eran máquinas de vectores soporte, en esta sección se utilizan los SuperVectors, técnica que fue explicada en la sección 3.3 de del presente informe.

El número de experimentos en este caso será mucho menor, tan sólo se cambiará la biblioteca Torch por LibSVM, sección 9.2 y se probará la influencia de la variable coste en los resultados 9.3. La última sección, 9.4, consta de una especial importancia, en ella se muestran los resultados de la fusión suma del sistema SVM y el sistema SuperVectors, obteniendo unos resultados altamente competitivos.

9.2 Migración a LibSVM

Al igual que se hizo con el sistema basado en SVM, (véase apartado 8.3), se realizó la migración de la biblioteca empleada para el entrenamiento y test de modelos utilizada por el sistema de SuperVectors.

En este apartado se mostrará una comparativa entre ambas bibliotecas, las variables medidas en dicha comparativa serán las habituales (EER, DCF, EER por modelo y EER por fichero de test), Tabla 27, curvas DETs, Figura 31, y eficiencia computacional (en la máquina habitual), Tabla 28.

La prueba realiza se corresponde con la tarea 1conv-1conv, para género masculino de NIST SRE 2006, los resultados incluyen normalización de puntuaciones mediante T-Norm y compensación de variabilidad intersesión, NAP.

Color	Biblioteca	EER (%)	DCF	EER modelo (%)	EER test (%)
Rojo	LibSVM	**5.02**	**0.024**	**3.11**	1.35
Negro ---	Torch	5.03	**0.024**	3.13	**1.35**

Tabla 27. Comparación resultados sistema SuperVectors, LibSVM vs. Torch

Biblioteca	Tiempo entrenamiento 20 modelos	Tiempo entrenamiento 1 modelo
LibSVM	1h38min	4min54s
Torch	**13min13s**	**39s**

Tabla 28. Comparativa eficiencia computacional sistema SuperVector, biblioteca LibSVM vs. Torch

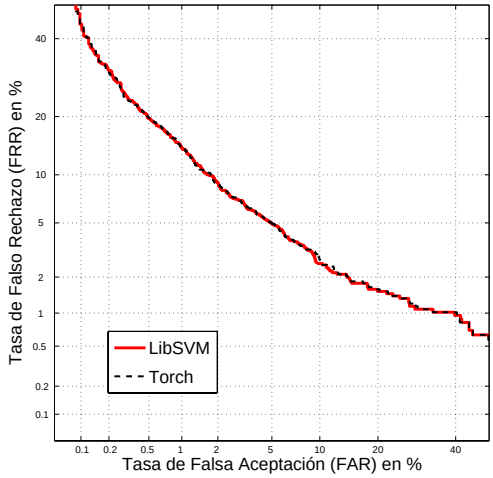


Figura 31. Curva DET del sistema SuperVector con Torch y con LibSVM

La tendencia de los resultados es muy similar a la vista con el sistema SVM, ambas bibliotecas presentan un comportamiento prácticamente idéntico en los resultados. Donde se diferencian es en la eficiencia computacional, en la Tabla 28 puede observarse una clara ventaja de Torch frente a LibSVM.

En la Tabla 4 se puso de manifiesto una ligera diferencia en los tiempos empleados en el entrenamiento de los modelos, en ese caso la diferencia era de tan sólo unos segundos. Con el sistema de SuperVectors la diferencia asciende hasta varios minutos, concretamente 4. La causa de esta diferencia está en el volumen de datos necesario para el entrenamiento, en el sistema SVM los vectores tenían 9880 dimensiones y el fichero de entrenamiento alcanzaba los 170Mb, el sistema de SuperVectors está compuesto por vectores de 38912 dimensiones lo que hace que el fichero de entrenamiento tenga un tamaño de 571Mb. La diferencia en el número de dimensiones da lugar a un entrenamiento más laborioso, este hecho unido una menor capacidad de LibSVM para trabajar con ficheros de gran tamaño hace que los modelos requieran una mayor cantidad de tiempo para completar su entrenamiento.

9.3 Distintos costes entrenamiento

Siguiendo la línea de experimentos del sistema SVM se comprobará la influencia de la variable coste en el comportamiento del sistema. Para las pruebas se eligieron dos valores de coste, 10 y 1000, los resultados se muestran en la Tabla 29 y la Figura 32.

Color	Coste	EER (%)	DCF	EER modelo (%)	EER test (%)
Rojo	10	**5.02**	**0.024**	**3.11**	**1.35**
Negro --	1000	5.12	0.026	3.37	1.56

Tabla 29. Comparación resultados sistema SuperVectors, coste entrenamiento

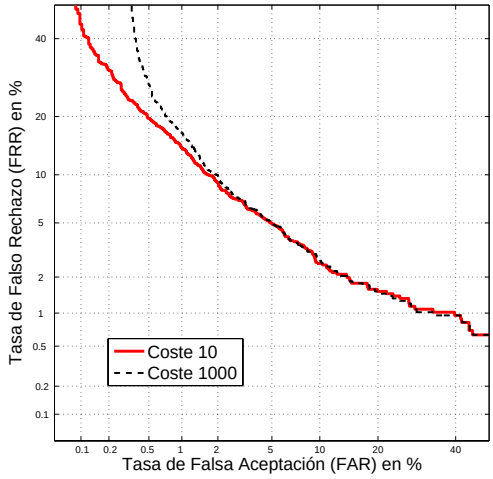


Figura 32. Curva DET del sistema SuperVector con distintos costes de entrenamiento

Como se desprende de la Figura 32 el sistema con un coste menor tienen un comportamiento muy similar al sistema con el coste de 1000, las curvas difieren sobre todo en la zona donde la falsa aceptación (FAR) es menor, siendo en esta zona mejor el sistema con coste menor, $C = 10$.

Si nos fijamos en la eficiencia computacional, Tabla 30, observaremos la misma tendencia observada con el sistema SVM, Tabla 7 apartado 8.4.

Coste	Tiempo entrenamiento 1 modelo
10	**4min54s**
1000	7min27s

Tabla 30. Comparación tiempos entrenamiento del sistema SuperVectors con distintos costes

9.4 Fusión SVM-GLDS y SuperVectors

La fusión de sistemas permite generar una única decisión final a partir de varios sistemas individuales, con un resultado final mejor que el de cada uno de los subsistemas por separado [Reynolds *et al.,* 2003b]. Existen varios tipos de fusiones, (véase sección 3.2) en este caso se mostrará la fusión suma del sistema SVM-GLDS basado en regresión y el sistema SuperVectors.

El funcionamiento de esta fusión suma es realmente muy sencillo basta con sumar las puntuaciones generadas por cada uno de los sistemas individuales en cada uno de los enfrentamientos, obteniéndose de este modo una única puntuación final.

Los experimentos mostrados a continuación hacen referencia a la evaluación NIST SRE 2006 para género masculino, en todos los resultados se han incluido normalización de puntuaciones, T-Norm, y compensación de variabilidad intersesión mediante NAP.

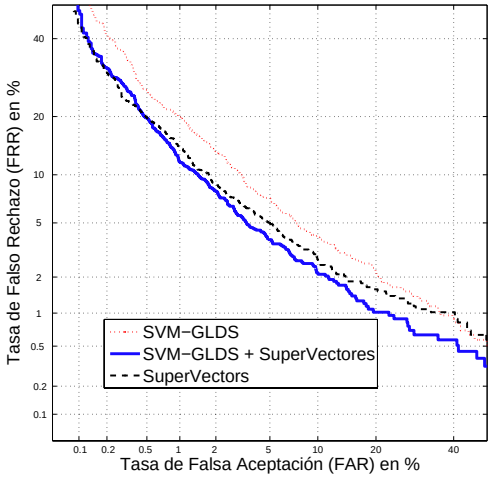


Figura 33. Curva DET de la fusión suma SVM-GLDS y SuperVectors

En la Figura 33 se pueden comprobar las afirmaciones hechas en el párrafo anterior y en la sección 3.2, la fusión suma da como resultado un sistema con un comportamiento mejor que el de los sistemas individuales. En este caso hemos conseguido un sistema con un 4.43% de EER y un 0.0216 de DCF.

Color	Sistema	EER (%)	DCF
Rojo …	SVM-GLDS	6.03	0.029
Negro ---	SuperVectors	5.03	0.024
Azul	Fusión suma	**4.43**	**0.022**

Tabla 31. Comparación resultados fusión suma SVM-GLDS y SuperVectors

10. Resultados reconocimiento de idioma con SVM

10.1 Introducción

En esta sección se presentan los resultados de la investigación en el campo de reconocimiento de idioma mediante SVM. Al igual que se hizo en las secciones de experimentos anteriores, se irán presentando por orden todos los experimentos llevados a cabo. Desde los relacionados con el tipo de parametrización, secciones 10.2, 10.10 y 10.11, los orientados al ajuste de variables en el entrenamiento, secciones 10.4, 10.5 y 10.7, hasta los dirigidos a normalizar las puntuaciones, sección 10.3, o la variabilidad intersesión, sección 10.5.

La secciones 10.8 y 10.9 gozan de un especial interés, en la primera de ellas se mostrará un estudio completo del sistema épsilon-SVR aplicado a reconocimiento de idioma, como veremos no presenta el mismo comportamiento observado en reconocimiento de locutor. Por último la sección 10.9 reflejará los mejores resultados obtenidos en este campo con SVM, resultados incluidos como colaboración en un artículo del grupo ATVS [Toledano *et al.,* 2007].

El rendimiento de los sistemas se expresará de la manera habitual, curva DET y tabla con resultados numéricos de EER, DCF, EER por modelo y EER por fichero de test. Debemos tener en cuenta que tanto el EER como el EER por modelo y fichero de test irán expresados en tanto por ciento (%), por tanto, se omitirá en las cabeceras de las tablas.

10.2 Parametrizaciones

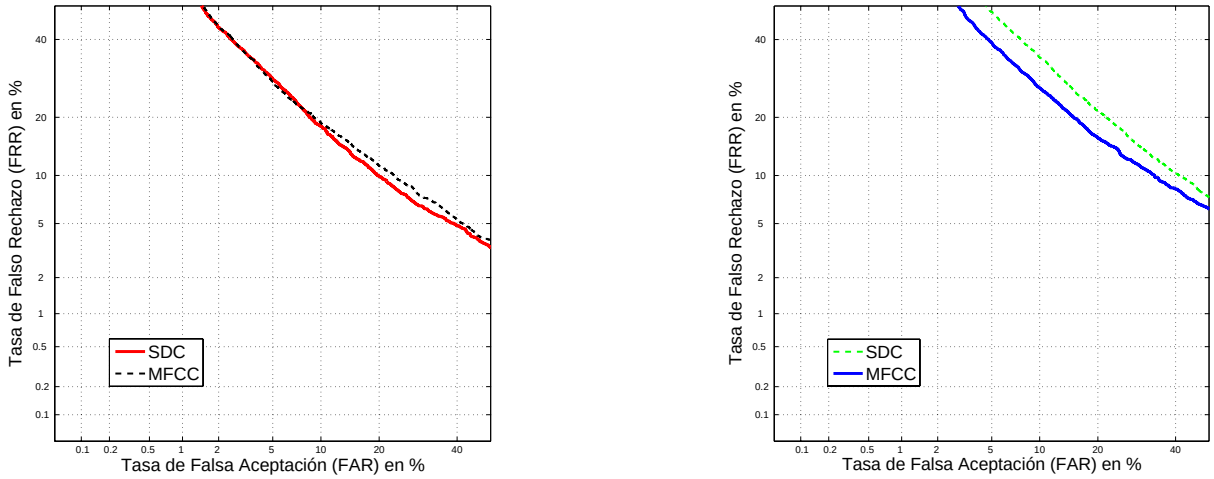
En la sección 5, se explicaron dos de las parametrizaciones más importantes utilizadas a la hora de extraer las características de la señal de voz, MFCC [Deller *et al.*, 1999] y SDC [Torres-Carrasquillo, 2002]. La tarea de reconocimiento de locutor se llevó a cabo mediante la parametrización MFCC, sin embargo, diversos estudios han mostrado mejoras significativas en el campo del reconocimiento de idioma al aplicar la parametrización SDC.

Esta sección de los experimentos irá orientada a comprobar las bondades de esta nueva parametrización, para ello se realizará una serie de experimentos en los que se comparará el rendimiento del sistema utilizando ambos tipos de parametrizaciones. Con el fin de que las conclusiones sean lo más generales posible, se emplearán los tipos de SVM empleados en reconocimiento de locutor, SVC y épsilon-SVR.

La configuración de los experimentos que se llevarán a cabo se resume en la Tabla 32.

Evaluación	NIST LRE 2005 30s		**Normalización**	T-Norm
Datos entrenamiento	Callfriend (30s y 30m) NIST LRE 1996 NIST LRE 2003		**Compensación canal**	Ninguna
Parametrización	MFCC + delta = 38coef SDC 7-2-3-7		**Tipo entrenamiento**	SVC épsilon-SVR

Tabla 32. Datos descriptivos de los experimentos con distinta parametrización: MFCC vs. SDC



a) **b)**

Figura 34. Comparación curvas DET parametrización MFCC y SDC con dos posibles tipos de SVM: a) SVC y b) SVR

Color	Tipo SVM	Parametrización	EER	DCF	EER modelo	EER test
Rojo	SVC	SDC	**13.98**	**0.063**	**15.19**	**10.57**
Negro --	SVC	MFCC	14.74	0.063	16.78	11.11
Azul	épsilon-SVR	SDC	20.89	0.083	22.16	17.46
Verde --	épsilon-SVR	MFCC	17.79	0.075	19.65	14.23

Tabla 33. Comparación resultados parametrización MFCC y SDC con dos posibles tipos de SVM: a) SVC y b) SVR

De los resultados podemos obtener dos observaciones de gran relevancia. En primer lugar, el comportamiento de la parametrización SDC comparado con la MFCC varía según el tipo de SVM que usemos. Si empleamos el tipo SVC, la parametrización SDC obtiene un valor de EER un 5.2% mejor que la MFCC. Por el contrario, si el tipo de SVM empleado es épsilon-SVR, la parametrización MFCC presenta un EER un 14.8% mejor que la SDC.

En segundo lugar, a diferencia de lo que sucedía en reconocimiento de locutor, sección 8.10, el entrenamiento de los modelos mediante épsilon-SVR obtiene resultados significativamente peores que SVC.

10.3 Normalización de puntuaciones, T-Norm, Z-Norm, ZT-Norm.

Al igual que se hizo en la sección 8.9 para locutor, en esta sección se aplicarán las técnicas de normalización de puntuaciones al sistema de reconocimiento de idioma. Los procedimientos explicados en dicha sección para T-Norm y Z-Norm, siguen siendo válidos, por lo que no se repetirán. Sin embargo, el caso de ZT-Norm requiere un tratamiento especial, ya que no llegó a aplicarse en reconocimiento de locutor.

ZT-Norm [Auckenthaler *et al.,* 2000] es una técnica de normalización de puntuaciones que intenta fusionar las virtudes de Z-Norm y T-Norm. Para ello, en primer lugar aplicará Z-Norm a las puntuaciones, después se utilizarán esas puntuaciones Z-normalizadas en el proceso de T-normalización.

Este tipo de normalizaciones consta de una fase muy importante, la selección de la cohorte de modelos de T-Norm y la selección de la cohorte de ficheros de Z-Norm. En el caso de reconocimiento de locutor la cohorte de T-Norm no debía incluir los modelos de otros usuarios de la evaluación, en reconocimiento de idioma, por el contrario, la cohorte de modelos suele ser precisamente el resto de idiomas de la evaluación. La selección de dicha cohorte es un elemento importante y sujeto a estudio, más adelante se mostrará una pequeña investigación sobre este hecho.

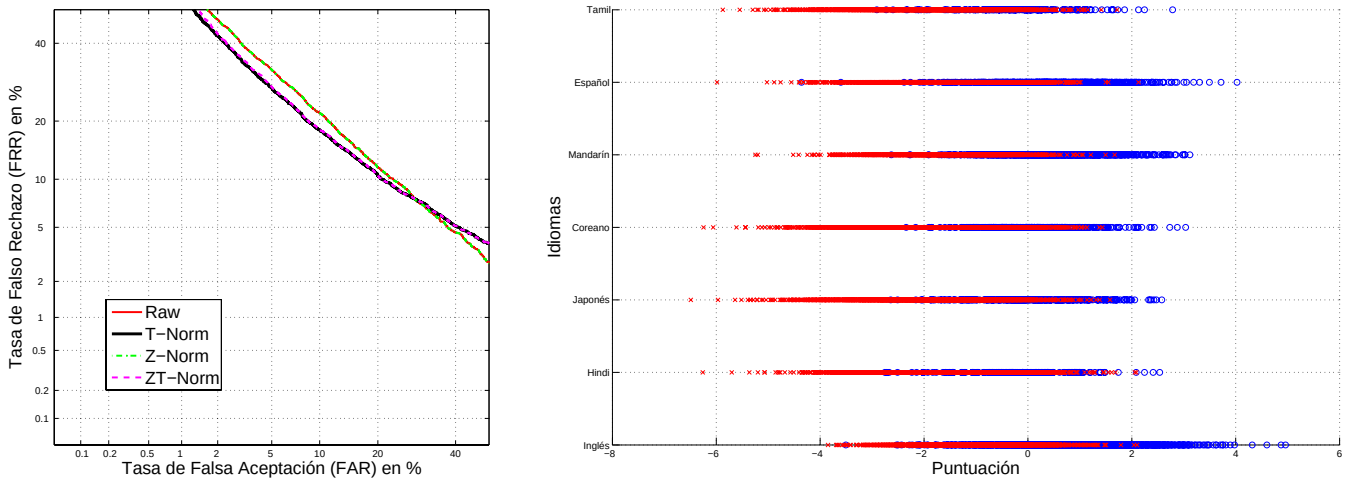
En la Tabla 34 se presenta la información relativa al experimento realizado para comprobar la bondad de las normalizaciones. Como cohorte de modelos de T-Norm se han usado los de la propia evaluación, como cohorte de ficheros de Z-Norm se emplearon 280 ficheros (40 por idioma) de la base de datos NIST LRE 1996.

Evaluación	NIST LRE 2005 30s	**Normalización**	T-Norm, Z-Norm y ZT-Norm
Datos entrenamiento	Callfriend (30s y 30m) NIST LRE 1996[3] NIST LRE 2003	**Compensación canal**	Ninguna
Parametrización	MFCC + delta = 38coef	**Tipo entrenamiento**	SVC

Tabla 34. Datos descriptivos del experimento de normalización de puntuaciones

La Figura 35 a) muestra las curvas DET del sistema sin ningún tipo de normalización de puntuaciones (Raw), y del sistema con las normalizaciones T-Norm, Z-Norm y ZT-Norm.

Los resultados numéricos de EER, DCF, EER por modelo y EER por fichero de test se presentan en la Tabla 35.



a) **b)**

Figura 35. a) Comparación curvas DET normalizaciones T-Norm, Z-Norm y ZT-Norm. b) Distribución de puntuaciones frente a los modelos

[3] Excluyendo los ficheros utilizados para la cohorte de Z-Norm.

Color	Normalización	EER	DCF	EER modelo	EER test
Rojo	Raw	15.33	0.066	16.86	**10.66**
Negro	T-Norm	**14.32**	**0.061**	**16.29**	**10.66**
Verde -.-	Z-Norm	15.33	0.066	16.86	**10.66**
Rosa ---	ZT-Norm	14.36	0.062	**16.29**	10.69

Tabla 35. Comparación resultados normalizaciones T-Norm, Z-Norm y ZT-Norm

De la tabla anterior podemos extraer varias informaciones. En primer lugar, tanto en EER como en DCF la normalización de puntuaciones que obtiene una mejora mayor es T-Norm. Por otro lado, los resultados del sistema raw y del sistema con normalización Z-Norm son idénticos. Este hecho puede ser debido a dos causas, bien la cohorte de ficheros elegida para Z-Norm es demasiado pequeña o bien las puntuaciones por modelo están alineadas en el sistema raw.

En la Figura 35 b) están representadas las puntuaciones, sin normalizar *(Raw)*, de los ficheros de test frente a los 7 modelos de la evaluación. Los círculos azules muestran las puntuaciones de usuario y las cruces rojas las de impostor, como se puede observar este sistema tiene las puntuaciones bastante alineadas, por lo que es normal que la normalización Z-Norm no aporte nada.

En el campo de la normalización T-Norm se han realizado diversas investigaciones orientadas a incrementar la mejora que supone dicha normalización. Se llevaron a cabo pruebas ampliando la cohorte de T-Norm, es decir añadiendo más modelos a los que enfrentar cada uno de los ficheros de test, estos modelos fueron 5 más, obtenidos de la base de datos Callfriend. Al ampliar la cohorte de T-Norm cada fichero de test es enfrentado a más modelos, de esta forma conseguimos más puntuaciones que nos ayuden a determinar mejor la media y la varianza con las que normalizamos.

La configuración del experimento orientado a incrementar la cohorte de T-Norm es la misma que la mostrada en la Tabla 34, con la salvedad de que no se excluyó ningún fichero de NIST LRE 1996. Las curvas DET del sistema se muestran en la Figura 36 y los resultados numéricos en la Tabla 36.

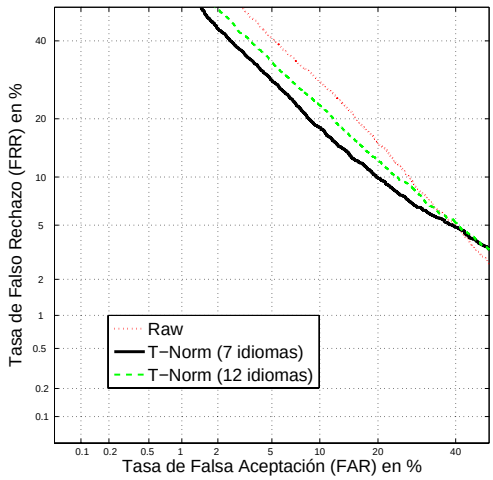


Figura 36. Curva DET del sistema con la nueva cohorte de T-Norm ampliada

Color	Normalización	Cohorte T-Norm	EER	DCF	EER modelo	EER test
Rojo …	Raw	-	17.68	0.075	18.70	**10.57**
Negro	T-Norm	7 idiomas	**13.98**	**0.063**	**15.19**	**10.57**
Verde ---	T-Norm	12 idiomas	15.69	0.069	16.62	11.84

Tabla 36. Comparación resultados del sistema con la nueva cohorte de T-Norm ampliada

Los resultados no fueron los esperados, el rendimiento del sistema empeoró y como se puede obtener de la Tabla 36, el EER se incrementó en algo más de un punto. Sin embargo, podemos observar mejora en EER del 21%, desde el sistema raw al sistema con la cohorte de 7 idiomas.

10.4 Influencia del conjunto de datos de entrenamiento

A la hora de entrenar cada uno de los modelos de idioma necesitaremos dos tipos de datos, los que hacen referencia a ficheros del idioma para el que construimos el modelo *(Targets)*, y los que hacen referencia al resto de los idiomas *(NonTargets)*.

Los experimentos presentados a continuación ilustran como el aumento de datos en el entrenamiento, influye en los resultados. Intuitivamente, puede pensarse que cuanto mayor sea el conjunto de datos que usemos para entrenar, mejor podremos construir el modelo correspondiente a cada uno de los idiomas. Además, cuanto mejor sean los modelos más exactas serán las puntuaciones producidas en los enfrentamientos con los ficheros de test, lo que nos llevará a una identificación más precisa.

Con la intención de mostrar la influencia del conjunto de datos de entrenamiento, se seguirá una metodología de pruebas simple. Bajo un escenario común se realizarán una serie de experimentos variando la cantidad de datos de entrenamiento por idioma. La descripción del escenario común puede verse en la Tabla 37.

Evaluación	NIST LRE 2005 30s		**Normalización**	T-Norm
Datos entrenamiento	Variable		**Compensación canal**	Ninguna
Parametrización	SDC 7-2-3-7		**Tipo entrenamiento**	épsilon-SVR

Tabla 37. Datos descriptivos del experimento de influencia del conjunto de datos de entrenamiento

La Figura 37 muestra los resultados obtenidos de cuatro experimentos distintos, en ella podemos ver como la curva DET baja a medida que el conjunto de entrenamiento contiene más datos. Estos mismos resultados son analizados con más detalle en la Tabla 39.

El conjunto de datos de entrenamiento que se ha usado en cada uno de ellos puede verse en la Tabla 38, la información sobre las bases de datos puede encontrarse en [NIST LRE; LDC].

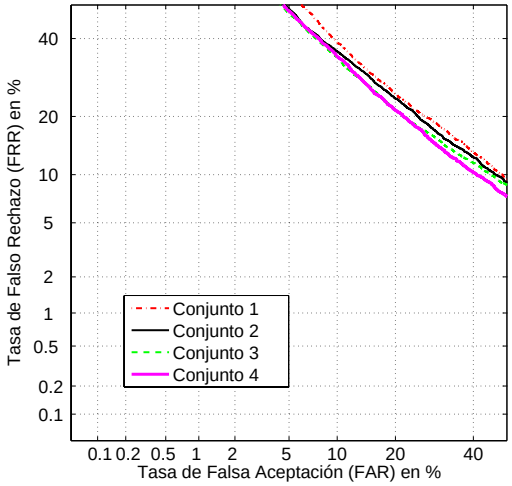


Figura 37. Curva DET del sistema con distintos conjuntos de entrenamiento

Bases de datos	Número ficheros	Duración fichero	Duración base datos	Conjunto 1	Conjunto 2	Conjunto 3	Conjunto 4
Callfriend	280	30m	8400m	X	X	X	X
	315	30s	157m30s				
NIST LRE 1996 entrenamiento	768	30s	384m		X	X	X
NIST LRE 1996 evaluación	1093	30s	546m30s			X	X
NIST LRE 2003	800	30s	400m				X
			Duración total	143h	149h	158h	165h

Tabla 38. Composición de los conjuntos de datos de entrenamiento. h, m, s denotan horas, minutos y segundos respectivamente. La duración total se ha redondeado

Color	Datos entrenamiento	EER	DCF	EER modelo	EER test
Rojo -.-	Conjunto 1	22.90	0.088	22.43	19.49
Negro	Conjunto 2	22.29	**0.083**	23.62	18.76
Verde ---	Conjunto 3	**20.86**	0.084	**21.32**	18.01
Rosa	Conjunto 4	20.89	0.083	22.16	**17.46**

Tabla 39. Comparación resultados del sistema con distintos conjuntos de entrenamiento

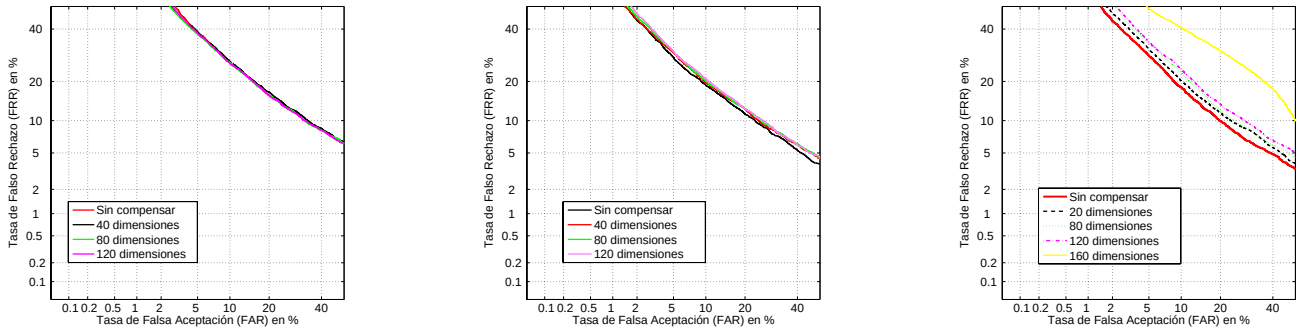
10.5 Compensación de variabilidad intersesión: NAP

Al igual que sucedía en reconocimiento automático de locutor, el reconocimiento de idioma presenta problemas de variabilidad intersesión, ya que las grabaciones están expuestas a ruidos o distorsiones propias de micrófonos, canal telefónico, etc. Como ya se vio en los experimentos realizados para reconocimiento de locutor, sección 8.8, NAP *(Nuisance Attribute Projection)* mejora los resultados eliminando esta variabilidad [Solomonoff *et al.,* 2004; Solomonoff *et al.,* 2005]. En esta sección de los experimentos se aplicará esa misma técnica de normalización para comprobar sus efectos en el rendimiento del sistema.

Con el fin de generalizar los resultados el máximo posible se mostrarán dos escenarios distintos de pruebas, uno para la parametrización MFCC y otro para la SDC. En los dos casos se realizarán diversas pruebas con distintos valores de dimensiones a compensar. La parametrización MFCC se probará tanto con el tipo de SVM SVC, como con épsilon-SVR. Las características del escenario de los experimentos se resumen en la Tabla 40.

Evaluación	NIST LRE 2005 30s		**Normalización**	T-Norm
Datos entrenamiento	Conjunto 4		**Compensación canal**	NAP
Parametrización	SDC 7-2-3-7 MFCC + delta = 38coef		**Tipo entrenamiento**	épsilon-SVR SVC

Tabla 40. Datos descriptivos del experimento de compensación de variabilidad intersesión



a) **b)** **c)**

Figura 38. Curvas DET del sistema con compensación de variabilidad intersesión, NAP: a) parametrización MFFC tipo SVM SVR, b) MFCC y SVC c) SDC y SVC

En la Figura 38 podemos ver las curvas DET del sistema, si nos fijamos en las gráficas a) y b), con parametrización MFCC, veremos como el comportamiento del sistema apenas varía. En la gráfica c), parametrización SDC, puede verse como los resultados empeoran a medida que eliminamos un mayor número de dimensiones.

Como se explicó en la sección 8.12, la matriz de referencia usada para la compensación consta de gran relevancia. En las gráficas mostradas esta matriz de referencia estaba compuesta por los datos de entrenamiento, conjunto 4 (Tabla 38), por lo que la compensación iba orientada a eliminar la variabilidad del idioma. En la Tabla 41, además de mostrar los resultados numéricos de las pruebas anteriormente citadas, se

muestran los resultados de realizar la compensación con una matriz de referencia compuesta por locutores y sus locuciones, la misma empleada en la sección 8.12.

Figura	Color	Tipo SVM	Parame-trización	Compensación	EER	DCF	EER modelo	EER test
a	Rojo	épsilon-SVR	MFCC	Ninguna	**17.79**	0.075	19.65	14.23
a	Negro	épsilon-SVR	MFCC	Idioma 40 dim.	18.12	0.074	20.02	14.50
a	Verde	épsilon-SVR	MFCC	Idioma 80 dim.	17.82	0.073	19.72	14.29
a	Rosa	épsilon-SVR	MFCC	Idioma 120 dim.	**17.79**	**0.074**	**19.58**	**14.21**
b	Negro	SVC	MFCC	Ninguna	**14.74**	**0.063**	**16.78**	**11.11**
b	Rojo	SVC	MFCC	Idioma 40 dim.	15.17	0.064	17.03	11.78
b	Verde	SVC	MFCC	Idioma 80 dim.	15.30	0.065	17.61	11.62
b	Rosa	SVC	MFCC	Idioma 120 dim.	15.57	0.066	17.76	11.84
c	Rojo	SVC	SDC	Ninguna	**13.98**	**0.063**	**15.19**	**10.57**
c	Negro ---	SVC	SDC	Idioma 20 dim.	14.97	0.067	16.12	11.49
c	Verde …	SVC	SDC	Idioma 80 dim.	15.83	0.069	16.72	12.33
c	Rosa -.-	SVC	SDC	Idioma 120 dim.	16.17	0.071	17.78	13.04
c	Amarillo	SVC	SDC	Idioma 160 dim.	26.36	0.077	28.46	12.84
-	-	SVC	MFCC	Ninguna	**14.18**	**0.062**	**16.42**	**10.59**
-	-	SVC	MFCC	Locutor 60 dim.	14.74	0.064	17.42	11.50

Tabla 41. Comparación resultados del sistema con distintas compensaciones de variabilidad intersesión

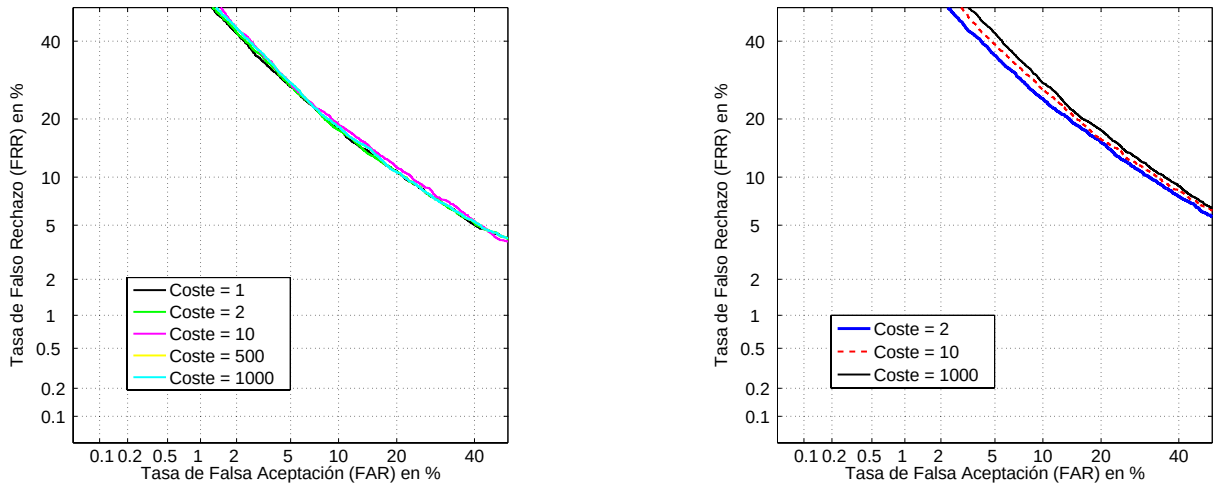
Tanto de los resultados gráficos como de los numéricos se extrae una misma idea, la compensación no está aportando nada al sistema. Por el contrario, se puede observar una clara tendencia de aumento de EER al incrementar el número de dimensiones compensadas. Esto nos lleva a la conclusión de que las dimensiones que eliminamos son importantes y representativas de cada idioma, en lugar de ser distorsiones molestas.

10.6 Distintos costes entrenamiento

La variable coste fue una de las variables sobre la que se realizaron varias pruebas en la sección de reconocimiento de locutor. En esa sección 8.4, se llegó a la conclusión de que su valor no influía en los resultados, sin embargo, cuanto mayor era el coste aplicado al entrenamiento más tiempo requería el sistema para entrenar los modelos.

En esta sección, bajo el marco común de la parametrización MFCC, se llevarán a cabo dos clases de experimentos. Por un lado se empleará el tipo de SVM SVC, por otro el tipo épsilon-SVR, los dos tipos de entrenamiento del SVM utilizados hasta ahora.

Al igual que sucedía en la sección de locutor, la parametrización MFCC de 19 coeficientes más deltas da como resultado vectores de 38 coeficientes. Una vez expandidos mediante la expansión polinómica de grado 3 [Wan y Campbell, 2000], la dimensión de los vectores resultante alcanza las 9880 dimensiones. Por lo tanto nos encontramos en una situación similar a la anterior, el número de vectores introducidos en el sistema es menor que las dimensiones de dichos vectores.



a) **b)**

Figura 39. Curvas DET del sistema con distintos costes de entrenamiento: a) tipo SVM SVC, b) tipos de SVM épsilon-SVR

Figura	Color	Tipo SVM	Coste	EER	DCF	EER modelo	EER test
a	Negro	SVC	1	14.18	**0.062**	**16.42**	**10.59**
a	Verde	SVC	2	**13.99**	**0.062**	16.47	10.71
a	Rosa	SVC	10	14.74	0.063	16.78	11.11
a	Amarillo	SVC	500	14.57	0.063	16.70	10.75
a	Azul	SVC	1000	14.57	0.063	16.70	10.75
b	Azul	épsilon-SVR	2	**17.34**	**0.071**	**19.34**	**13.72**
b	Rojo ---	épsilon-SVR	10	17.79	0.075	19.65	14.23
b	Negro	épsilon-SVR	1000	18.66	0.078	20.51	15.05

Tabla 42. Comparación resultados distintos costes de entrenamiento, SVC y épsilon-SVR

Tanto de la Figura 39 como de la Tabla 42 podemos extraer una conclusión, la variable coste no influye en gran medida en los resultados, pero cuanto menor es su valor mejores son estos. Si bien es verdad que los experimentos hechos con esta variable en reconocimiento de locutor, Tabla 6, mostraban unos valores aun más similares. Este hecho puede explicarse desde el punto de vista de los vectores utilizados como *Targets*, mientras que en reconocimiento de locutor tan sólo teníamos un vector que identificada al locutor en reconocimiento de idioma tenemos bastante más. La Tabla 43 muestra el número de estos vectores *Targets* por idioma. Debemos tener en cuenta que al conjunto de datos 4 presentado en la Tabla 38 se han añadido otros 280 ficheros de 30 minutos de la base de datos Callfriend [LDC].

Mientras no se diga lo contrario este será el conjunto de datos utilizado en las pruebas para el resto de los experimentos, al que denominaremos *Conjunto Total.*

Idioma	Vectores Targets
Inglés	1002
Hindi	358
Japonés	442
Coreano	358
Mandarín	515
Español	509
Tamil	350

Tabla 43. Número de vectores Targets por idioma

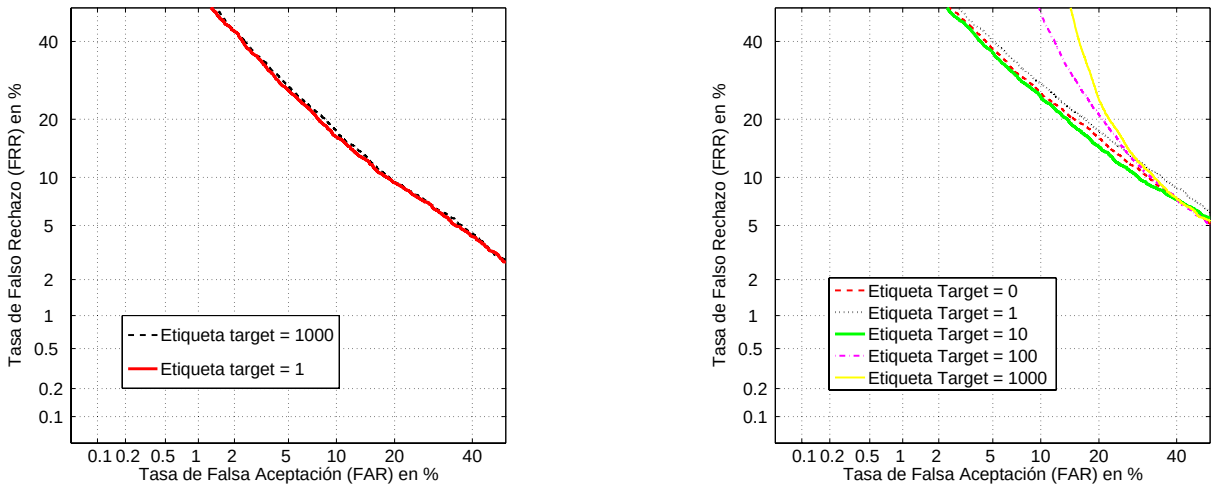
Como se desprende de la tabla anterior, el número de vectores Targets por idioma no es constante, cada idioma tendrá más o menos dependiendo de la información que exista sobre ellos en la base de datos. El inglés, al ser el más popular y extendido es con diferencia el idioma del que más información existe en las bases de datos.

Más adelante, en la sección 10.8, veremos como el elevado número de vectores *Targets* parece ser el principal culpable de que el tipo de SVM épsilon-SVR no alcance el rendimiento esperado.

10.7 Coste de la clase Target

En la sección 8.5 se comprobó la influencia del valor de la etiqueta de los vectores *NonTargets* en el resultado. Siguiendo la misma línea de experimentos llevada a cabo en reconocimiento de locutor, probaremos en esta sección la influencia de las etiquetas de los vectores en el reconocimiento de idioma.

El valor de etiqueta de *NonTargets* con mejores resultados en reconocimiento de locutor era el -1, véase Tabla 8. En este caso tomaremos este valor como óptimo y variaremos el valor de la etiqueta de los vectores *Targets*. Este cambio en la forma de proceder viene propiciado por la gran cantidad de vectores *Targets* que componen el entrenamiento de un modelo de idioma. Al tener una mayor cantidad de datos del idioma en si, será más probable que alguno de estos datos (puntos) sea incorrectamente clasificado.
La parametrización usada en este caso será SDC, con los dos tipos de SVM posibles.



a) **b)**

Figura 40. Curvas DET sistema con distintos valores de etiqueta Target: a) SVC, b) épsilon-SVR

Figura	Color	Tipo SVM	Etiqueta Target	EER	DCF	EER modelo	EER test
a	Negro ---	SVC	1000	13.79	**0.062**	15.16	10.14
a	Rojo	SVC	1	**13.32**	**0.062**	**14.99**	**9.88**
b	Rojo ---	épsilon-SVR	0	18.09	0.073	**18.80**	13.92
b	Negro …	épsilon-SVR	1	18.71	0.074	20.42	14.99
b	Verde	épsilon-SVR	10	**17.01**	**0.072**	19.14	**13.67**
b	Rosa -.-	épsilon-SVR	100	20.50	0.089	20.70	17.32
b	Amarillo	épsilon-SVR	1000	21.39	0.092	20.60	20.06

Tabla 44. Comparación resultados distintos valores etiqueta Target, SVC y épsilon-SVR

De los resultados obtenidos podemos obtener dos conclusiones de gran relevancia, en primer lugar la influencia del valor de la etiqueta en los resultados. Los mejores valores se obtienen con valores de etiqueta 1 y 10, para el tipo de entrenamiento SVC y épsilon-SVR respectivamente. En segundo lugar, el valor de la etiqueta no influye de la misma manera en los dos tipos de SVM. Mientras que la diferencia en EER entre los sistemas SVC con etiqueta 1 y 1000 es de un 3.5%, los sistemas épsilon-SVR con estos mismos valores de etiqueta presentan una diferencia de un 12.5%.

Basándonos en los valores presentados en la Tabla 42 y la Tabla 44, podemos concluir que el entrenamiento basado en regresión es más sensible a la influencia de las variables coste y valor de la etiqueta de Target. Por tanto, a la hora trabajar con un sistema SVM basado en épsilon-SVR deberemos realizar una serie de pruebas que nos lleven a encontrar la configuración óptima del sistema.

10.8 SVM épsilon-SVR

En esta sección se presentarán distintas investigaciones, todas ellas fueron llevadas a cabo con el fin de explicar los resultados vistos en la Tabla 33. Esta tabla mostraba como el tipo de entrenamiento SVC era más eficiente que el épsilon-SVR, todo lo contrario a lo visto en reconocimiento automático de locutor. La Tabla 23 y la Figura 28 de la sección de locutor reflejaban una diferencia sustancial entre ambos tipos de entrenamiento, siendo claramente más eficiente el tipo de SVM épsilon-SVR.

Influencia del valor de la variable épsilon

En primer lugar se realizará un estudio sobre la influencia del valor de la variable épsilon, igual que se hizo en la Tabla 20 y la Tabla 21. Este estudio se llevará a cabo con la base de datos Callfriend, y sin ningún tipo de normalización de puntuaciones. La Tabla 37 resume el escenario del experimento.

Evaluación	Callfriend 30s	**Normalización**	Raw
Datos entrenamiento	Callfriend 30m	**Compensación canal**	Ninguna
Parametrización	SDC 7-2-3-7	**Tipo entrenamiento**	épsilon-SVR

Tabla 45. Datos descriptivos del experimento de influencia del valor de épsilon

Los resultados se presentan de manera gráfica en la Figura 41 y numéricamente en la Tabla 46. Se ha añadido como valor de referencia los resultados del sistema con SVC, de esta forma podremos ver si alcanzamos en algún momento ese valor.

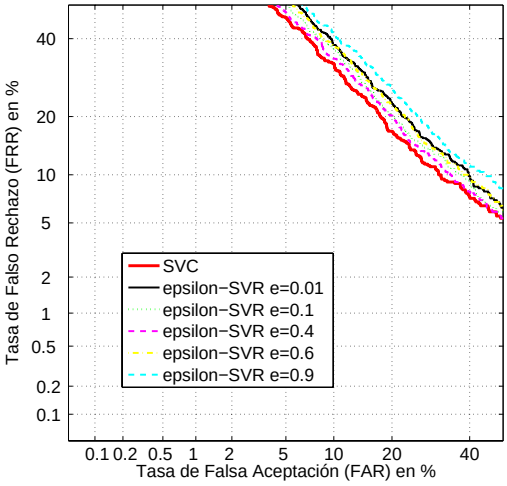


Figura 41. Curvas DET evaluación influencia valor épsilon en épsilon-SVR

Color	Sistema	Valor de ε	EER	DCF	EER modelo	EER test
Rojo	SVC	-	**18.70**	**0.081**	**18.72**	10.22
Negro	épsilon-SVR	0.01	21.38	0.086	21.60	10.98
Verde …	épsilon-SVR	0.1	20.12	0.083	20.42	10.15
Rosa ---	épsilon-SVR	0.4	20.13	0.082	20.24	**9.76**
Amarillo -.-	épsilon-SVR	0.6	21.19	0.083	21.35	10.87
Azul ---	épsilon-SVR	0.9	23.05	0.086	22.78	11.45

Tabla 46. Comparación resultados distintos valores de épsilon

De los resultados podemos extraer una conclusión clara, el sistema con el tipo de SVM SVC presenta mejores prestaciones que cualquiera de los sistemas basados en épsilon-SVR. El valor de épsilon con mejor comportamiento en EER es 0.1, por tanto, desde este punto en adelante será este valor el que se use para todos los experimentos restantes.

Cantidad de datos Targets

El siguiente paso en nuestra investigación irá orientado a comprobar si la gran cantidad de datos Targets, mostrados en la Tabla 43, es la causante de que los modelos con regresión no presenten el comportamiento esperado. Para realizar esta comprobación llevaremos a cabo una demostración similar a la realizada en el apartado 10.4. Utilizaremos por un lado el conjunto de datos 4 (véase Tabla 38), por otro lado el conjunto de datos total mostrado en la Tabla 43. Ambos conjuntos se parametrizarán mediante MFCC y se probarán en sistemas de clasificación y regresión.
La tabla descriptiva del experimento se presenta a continuación:

Evaluación	NIST LRE 2005 30s	**Normalización**	T-Norm
Datos entrenamiento	Conjunto 4 Conjunto total	**Compensación canal**	Ninguna
Parametrización	MFCC	**Tipo entrenamiento**	SVC épsilon-SVR

Tabla 47. Datos descriptivos de la investigación sobre el número de datos Target

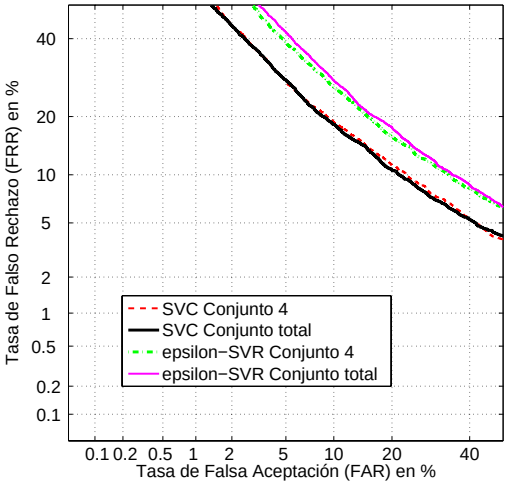


Figura 42. Curvas DET evaluación influencia de la cantidad de datos Target en épsilon-SVR

Color	Sistema	Conjunto datos	EER	DCF	EER modelo	EER test
Rojo ---	SVC	Conjunto 4	14.74	0.063	16.78	11.11
Negro	SVC	Conjunto total	**14.57**	**0.062**	**16.70**	**10.75**
Verde -.-	épsilon-SVR	Conjunto 4	**17.79**	**0.075**	**19.65**	**14.22**
Rosa	épsilon-SVR	Conjunto total	18.68	0.077	20.48	14.95

Tabla 48. Comparación resultados cantidad de datos Target en el entrenamiento

La Tabla 48 confirmar las sospechas, la cantidad de vectores usados como Target en el entrenamiento es la causante del mal comportamiento del sistema basado en regresión. Mientras que el sistema SVC mejora en todos los aspectos al incrementar los datos de entrenamiento, el sistema épsilon-SVR empeora. Es decir, lo que parecía obvio en la sección 10.4, cuanta más información se tenía para entrenar los modelos mejor se comportaba el sistema, deja de cumplirse al pasar de un cierto volumen de datos.

Estas pruebas nos llevan a afirmar que los sistemas basados en regresión son mucho más sensibles al número de datos a la hora de entrenar. Una línea importante de investigación será la de encontrar un conjunto de datos, con un tamaño moderado, que recoja la máxima variabilidad posible de los idiomas.

El siguiente paso en la investigación será comparar los sistemas SVC y épsilon-SVR con un conjunto de datos pequeño.

Comparación SVC y épsilon-SVR con 40 puntos Target por idioma

Todas las pruebas llevadas a cabo en este apartado utilizan como datos de entrenamiento 40 ficheros por idioma, de 30 minutos cada uno, obtenidos de la base de datos Callfriend, lo que se traduce en 40 puntos Targets por idioma. Se realizaron varios experimentos con regresión en los que se varió el valor de la etiqueta de la clase Targets, y uno con clasificación (con la mejor configuración posible) que serviría de referencia para la comparación.

Los resultados obtenidos en la sección 10.7 motivaron las pruebas con distintos valores de esta etiqueta para la clase Target, en dicha sección se vio como este valor influía en

gran medida en los resultados, por lo tanto debía pasar un periodo de ajuste. La Tabla 49 resume el escenario de los experimentos.

Evaluación	NIST LRE 2005 30s		**Normalización**	T-Norm
Datos entrenamiento	Callfriend 30m		**Compensación canal**	Ninguna
Parametrización	SDC (7-2-3-7)		**Tipo entrenamiento**	SVC épsilon-SVR

Tabla 49. Datos descriptivos de los experimentos con 40 puntos Targets por idioma

Los resultados de estas pruebas se presentarán en el formato habitual, gráfica con las curvas DET, Figura 43, y tabla resumen de resultados, Tabla 50 .

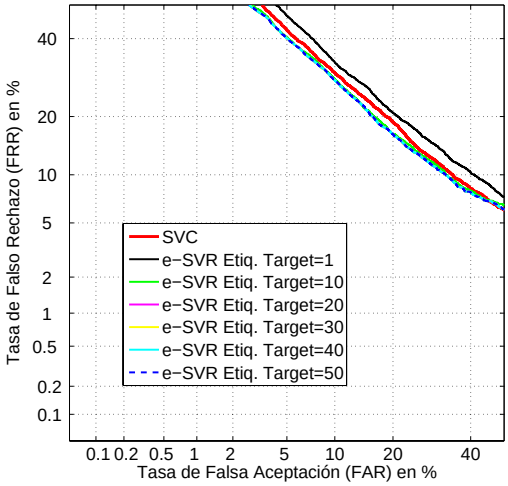


Figura 43. Curvas DET de los sistemas SVC y épsilon-SVR con 40 puntos Target por idioma

Color	Sistema	Etiqueta Target	EER	DCF	EER modelo	EER test
Rojo	SVC	-	19.46	0.078	19.10	15.25
Negro	épsilon-SVR	1	20.53	0.082	20.05	16.79
Verde	épsilon-SVR	10	18.32	0.073	18.64	15.27
Rosa	épsilon-SVR	20	**17.99**	**0.072**	**18.46**	**15.18**
Amarillo	épsilon-SVR	30	**17.99**	**0.072**	**18.46**	15.19
Azul	épsilon-SVR	40	**17.99**	**0.072**	**18.46**	15.19
Azul ---	épsilon-SVR	50	**17.99**	**0.072**	**18.46**	**15.18**

Tabla 50. Comparación resultados sistemas con 40 puntos Target por idioma

Los resultados obtenidos tienen una gran importancia, es la primera vez que en reconocimiento de idioma el tipo de entrenamiento épsilon-SVR supera las prestaciones del tipo SVC. La Tabla 50 confirma las suposiciones realizadas anteriormente, el volumen de datos de entrenamiento elevado trae consigo un empeoramiento del comportamiento del sistema en regresión.

La diferencia en EER entre el sistema SVC y el épsilon-SVR con etiqueta para la clase Target mayor que 20 es casi un 8%. Motivados por estos resultados y los conseguidos en reconocimiento automático de locutor, se comenzó a investigar en una nueva línea de trabajo dirigida a conseguir un sistema basado en épsilon-SVR eficiente.

Agrupación de vectores Target

La nueva línea de trabajo fue orientada a reducir el número de puntos Target de cada idioma. Para ello se implementaron diversas soluciones que se detallan a lo largo de este apartado.

En primer lugar se pensó en agrupar todos los vectores (puntos) Target en uno sólo. De esta manera conseguiríamos un sistema similar al implementado en reconocimiento automático de locutor. Para llevar a cabo esta agrupación se siguió un procedimiento sencillo, calcular el punto medio de la distribución de puntos. Esta técnica de agrupación no requiere grandes algoritmos ni tiempo computacional excesivo, tan sólo es necesario sumar los vectores y dividir el vector resultante entre el total de vectores.

El escenario de los experimentos es el mostrado en la Tabla 51, aunque el objetivo es mejorar el sistema basado en épsilon-SVR, se probará con ambos tipos de SVM. De esta forma podremos examinar y comprender mejor el comportamiento de los dos tipos de entrenamiento de modelos.

Evaluación	NIST LRE 2005[4] 30s	**Normalización**	T-Norm
Datos entrenamiento	Conjunto total	**Compensación canal**	Ninguna
Parametrización	SDC (7-2-3-7)	**Tipo entrenamiento**	SVC épsilon-SVR

Tabla 51. Datos descriptivos de los experimentos de agrupación de vectores Target

Como podemos extraer de la Figura 44 los resultados empeoran de manera drástica. Al agrupar todos los vectores en uno mediante el cálculo del vector medio perdemos mucha información relevante para el sistema. En la Tabla 52 podemos apreciar que tanto para el caso SVC como para épsilon-SVR el EER casi se duplica.

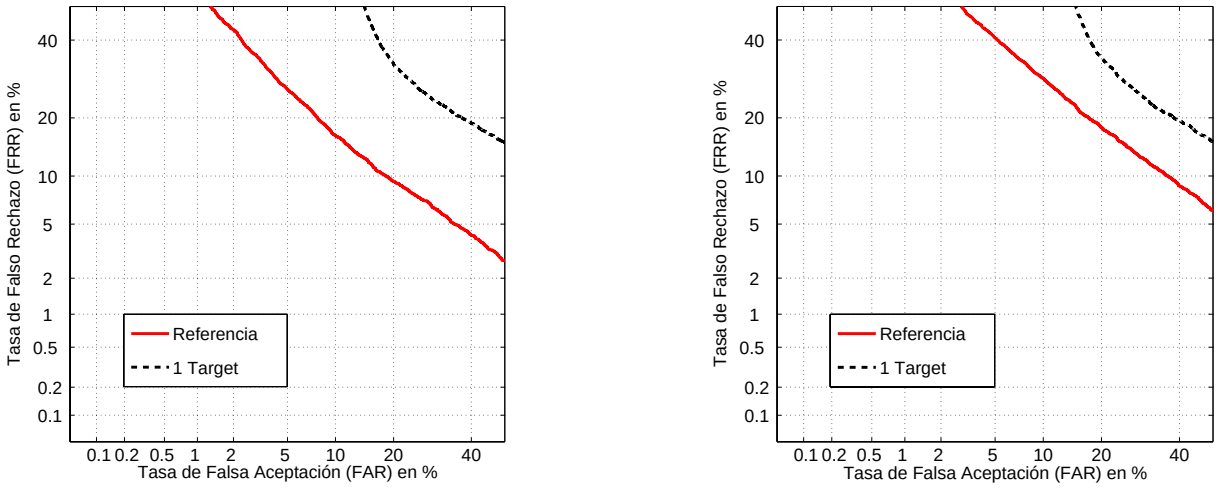


Figura 44. Curvas DET del sistema con agrupación de los vectores Target en un solo punto: a) SVC, b) épsilon-SVR

[4] En este experimento se añaden a la evaluación NIST LRE 2005 84 archivos de alemán de 30s de test. El resultado varía mínimamente con lo obtenido hasta ahora, ya que son 84 archivos de 3662 que conforman el total. A partir de este momento se empleará la evaluación completa.

Figura	Color	Sistema	Target	EER	DCF	EER modelo	EER test
a	Rojo	SVC	Todos Target	**13.32**	**0.062**	**14.99**	**9.88**
a	Negro ---	SVC	1 sólo Target	26.34	0.100	31.43	24.48
b	Rojo	épsilon-SVR	Todos Target	**19.01**	**0.076**	**20.70**	**15.30**
b	Negro ---	épsilon-SVR	1 sólo Target	27.09	0.100	31.66	24.85

Tabla 52. Comparación resultados sistemas con agrupación de vectores Target en un solo punto

Visto que esta forma de reducción del número de vectores Target no obtenía buenos resultados, se pensó en otra forma de agrupación de vectores más elaborada. La nueva técnica elegida fue K-means [Duda *et al.,* 2001].

K-means es un algoritmo de agrupación de objetos *(clustering)*, que clasifica a los objetos en particiones basándose en sus atributos. El algoritmo tratará de encontrar los centros de estas particiones, a los que llamaremos centroides *(codeword)*, basándose en un criterio de distancia. El mejor conjunto de centroides, al que llamaremos *codebook*, será aquel cuya suma de las distancias de los vectores a su centroide correspondiente sea mínima. En la
Figura 45 se muestra un ejemplo gráfico en dos dimensiones de un codebook con dos centroides.

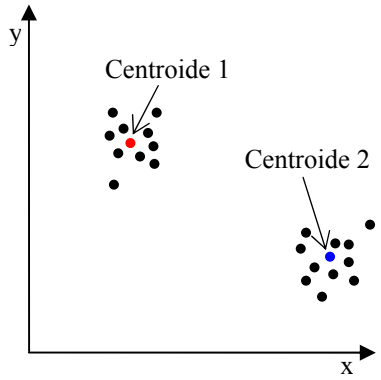


Figura 45. Ejemplo K-means, codebook de dos centroides en dos dimensiones

Las cuatro etapas del algoritmo son las siguientes:

1. Inicialización: selección de N vectores como los centroides iniciales del codebook. Hay varias posibilidades: selección aleatoria, segmentación inicial en N tramos iguales con selección aleatoria, ídem con selección promedio, etc.
2. Búsqueda NN *(nearest-neighbour)*: para cada vector de entrenamiento, buscaremos el centroide más próximo, y asociaremos dicho vector a la clase correspondiente representada por el centroide.
3. Actualización de centroides: con los vectores asociados a cada clase, calculamos el nuevo centroide que represente mejora a los elementos de esa clase.
4. Iteración: repetimos los pasos 2. y 3. hasta que la distancia promedio caiga por debajo de un umbral predeterminado, o hasta llegar a un número de iteraciones dado.

Una vez explicado el funcionamiento del algoritmo que usaremos para agrupar los vectores Target pasamos a los experimentos. Las dos variables más importantes del

algoritmo, y que por tanto deberemos ajustar, son el número de iteraciones y el número de centroides que compondrá el codebook.
Empezaremos viendo la influencia del número de iteraciones del algoritmo en los resultados, para ello fijaremos el número de centroides a 40, número de puntos Target con el que vimos en la Tabla 50 que épsilon-SVR superaba a SVC. La descripción del marco experimental se detalla en la Tabla 53, este marco experimental será común para el resto de los experimentos de esta misma sección.

Evaluación	NIST LRE 2005 30s		**Normalización**	Raw
Datos entrenamiento	Conjunto total		**Compensación canal**	Ninguna
Parametrización	SDC (7-2-3-7)		**Tipo entrenamiento**	SVC épsilon-SVR

Tabla 53. Datos descriptivos de los experimentos con K-Means

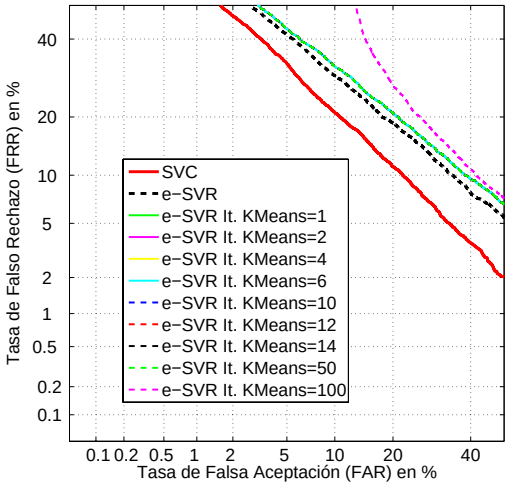


Figura 46. Curvas DET con distintas iteraciones para el algoritmo K-means

Color	Sistema	Iteraciones K-means	EER	DCF	EER modelo	EER test
Rojo	SVC	-	**15.22**	**0.065**	**16.41**	**10.28**
Negro ---	épsilon-SVR	-	19.30	0.072	21.63	13.67
Verde	épsilon-SVR	1	20.47	0.075	22.64	14.50
Rosa	épsilon-SVR	2	20.47	0.075	22.64	14.50
Amarillo	épsilon-SVR	4	20.47	0.075	22.64	14.50
Azul	épsilon-SVR	6	20.47	0.075	22.64	14.50
Azul ---	épsilon-SVR	10	20.47	0.075	22.64	14.50
Rojo ---	épsilon-SVR	12	20.47	0.075	22.64	14.50
Negro ---	épsilon-SVR	14	20.47	0.075	22.64	14.50
Verde ---	épsilon-SVR	50	20.47	0.075	22.64	14.50
Rosa ---	épsilon-SVR	100	23.29	0.099	24.93	23.09

Tabla 54. Comparación resultados distintas iteraciones algoritmo K-means

La Figura 43 y la Tabla 54 muestran como el comportamiento del sistema es el mismo con la mayor parte de las iteraciones probadas. Sólo en el caso de realizar 100 iteraciones el algoritmo obtiene un codebook que empeorará el comportamiento del

sistema. En lo sucesivo se utilizarán dos iteraciones del algoritmo K-means para generar el codebook, de esta forma el proceso será más rápido.
El siguiente conjunto de experimentos sigue el mismo escenario mostrado en la Tabla 53, en esta ocasión la variable a ajustar será el número de centroides que componen el codebook. Por un lado un codebook mayor representará mejor la nube de puntos Target del sistema, por otro lado un número elevado de vectores hará que los modelos de regresión no se entrenen bien, como vimos al comienzo de esta sección en la Tabla 48.

La Figura 47 muestra las curvas DET del sistema con los tamaños de codebook seleccionados, desde 10 hasta 100 vectores. Se realizaron dos tipos de pruebas, unas con valor de etiqueta para la clase Target 10 y otras con valor 20. Estos valores de etiqueta fueron seleccionados en base a que fueron los que mostraron mejores prestaciones en los experimentos de ajuste de dicho valor, Tabla 44 y Tabla 50.

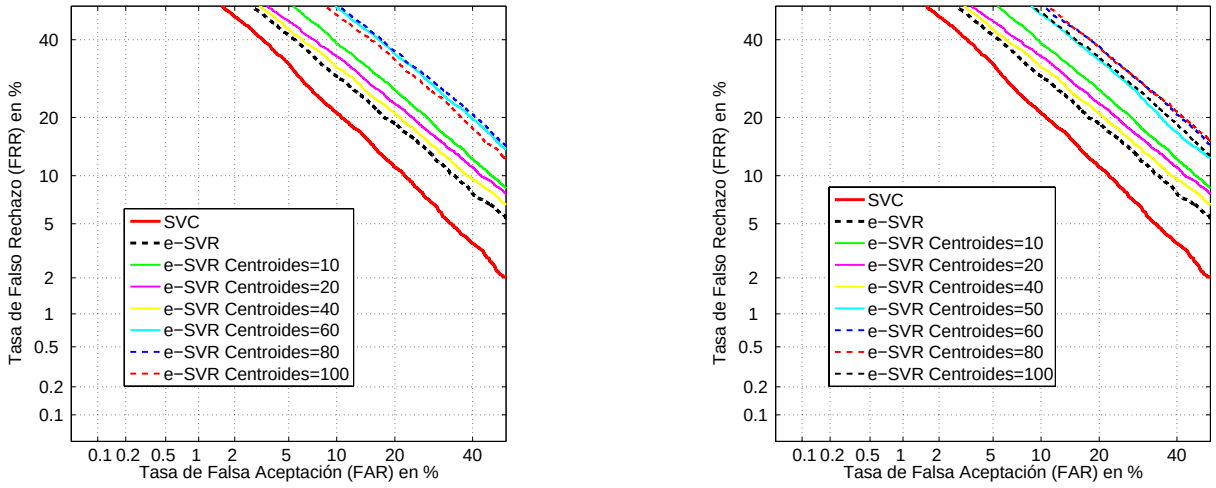


Figura 47. Curvas DET del sistema épsilon-SVR con distintos tamaños de codebook: a) Etiqueta Target = 10, b) etiqueta Target = 20

Figura	Color	Tipo SVM	Etiqueta Target	Número centroides	EER	DCF	EER modelo	EER test
a y b	Rojo	SVC	-	-	**15.22**	**0.065**	**16.41**	**10.28**
a y b	Negro ---	épsilon-SVR	1	-	19.30	0.072	21.63	13.67
a	Verde	épsilon-SVR	10	10	23.27	0.079	24.43	17.55
a	Rosa	épsilon-SVR	10	20	21.83	0.076	23.03	15.73
a	Amarillo	épsilon-SVR	10	40	20.47	0.075	22.64	14.50
a	Azul	épsilon-SVR	10	60	28.25	0.088	30.45	22.75
a	Azul ---	épsilon-SVR	10	80	28.77	0.087	30.56	23.13
a	Rojo ---	épsilon-SVR	10	100	27.26	0.086	29.42	21.63
b	Verde	épsilon-SVR	20	10	23.27	0.079	24.43	17.55
b	Rosa	épsilon-SVR	20	20	21.83	0.076	23.03	15.73
b	Amarillo	épsilon-SVR	20	40	20.47	0.075	22.64	14.50
b	Azul	épsilon-SVR	20	50	27.23	0.086	28.96	21.52
b	Azul ---	épsilon-SVR	20	60	28.96	0.089	31.30	23.62
b	Rojo ---	épsilon-SVR	20	80	29.08	0.088	30.82	23.66
b	Negro ---	épsilon-SVR	20	100	27.66	0.086	30.03	21.91

Tabla 55. Comparación resultados sistema épsilon-SVR con distintos tamaños de codebook

A la vista de los resultados presentados en la Tabla 55, podemos concluir que el comportamiento del sistema épsilon-SVR no mejora al reducir el número de vectores Targets mediante K-means. Los mejores resultados se obtiene con 40 centroides, dejando el EER del sistema en un 20.5%, algo más de un punto por encima de los resultados del sistema entrenado con los datos sin agrupar, y 5 puntos por encima del sistema basado en SVC. Como era de esperar la agrupación de los vectores mediante K-means es mucho más eficiente que el cálculo del vector medio explicado anteriormente.

Para concluir esta sección de los experimentos realizaremos un par de pruebas más, esta vez el tipo de SVM utilizado será SVC. Las pruebas trataran de comprobar la influencia de reducir el número de vectores Target introducidos al sistema SVC mediante K-means. Se probarán los dos tipos de parametrizaciones vista hasta ahora, MFCC y SDC.

El marco de los experimentos sigue siendo el mostrado en la Tabla 53.

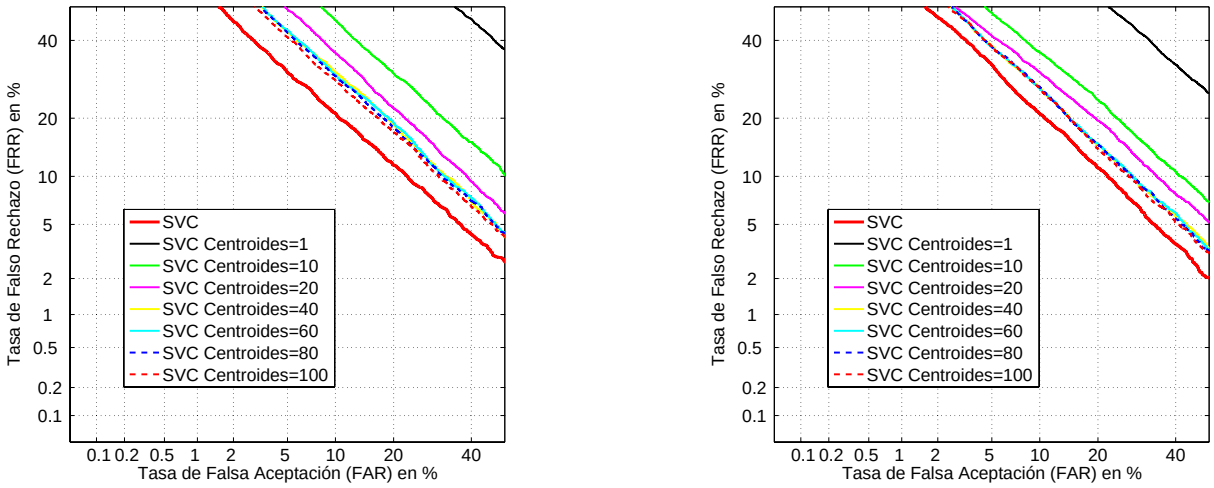


Figura 48. Curvas DET del sistema SVC con distintos tamaños de codebook: a) parametrización MFCC, b) parametrización SDC

Color	Número centroides	Parametrización	Figura	EER	DCF	EER modelo	EER test
Rojo	-	MFCC	a	15.27	0.065	17.06	10.33
		SDC	b	**15.22**	**0.065**	**16.41**	**10.28**
Negro	1	MFCC	a	43.33	0.100	43.13	42.21
		SDC	b	36.21	0.097	36.27	33.92
Verde	10	MFCC	a	25.62	0.091	28.22	20.08
		SDC	b	22.17	0.079	23.07	16.52
Rosa	20	MFCC	a	21.31	0.082	23.27	15.66
		SDC	b	19.78	0.073	20.28	14.43
Amarillo	40	MFCC	a	19.38	0.078	20.90	13.39
		SDC	b	17.23	0.072	18.40	21.25
Azul	60	MFCC	a	19.61	0.078	20.73	13.42
		SDC	b	17.31	0.072	18.39	12.15
Azul ---	80	MFCC	a	19.05	0.077	20.40	13.41
		SDC	b	17.11	0.071	18.16	12.07
Rojo ---	100	MFCC	a	18.49	0.075	19.41	12.85
		SDC	b	17.15	0.071	17.99	12.16

Tabla 56. Comparación resultados sistema SVC con distintos tamaños de codebook

La información de la Tabla 56 apunta a que el sistema basado en clasificación tiene un comportamiento similar al sistema basado en regresión. En ningún caso la agrupación de vectores mediante K-means supera las prestaciones del sistema entrenado con todo el conjunto de datos. Por otro lado, a medida que aumentamos el número de centroides, es decir, el tamaño del codebook, los resultados se acercan más al comportamiento del sistema de referencia.

10.9 Fusión parametrizaciones MFCC y SDC

Como se explicó en la sección 3.2, la fusión de subsistemas complementarios trata de sacar el máximo partido a la información aportada por cada uno de los sistemas. Los resultados obtenidos en el campo del reconocimiento de locutor, mediante la fusión suma del sistema SVM-GLDS y SuperVectors, sección 9.4, dieron por probada la bondad de esta técnica.

Los experimentos incluidos en esta sección irán orientados a comprobar el comportamiento de un sistema global que fusione dos subsistemas, por un lado el subsistema con parametrización MFCC y por otro el subsistema con parametrización SDC.

La Figura 49 y la Tabla 57 resumen los resultados obtenidos de fusionar mediante la regla de la suma los dos subsistemas mencionados anteriormente.

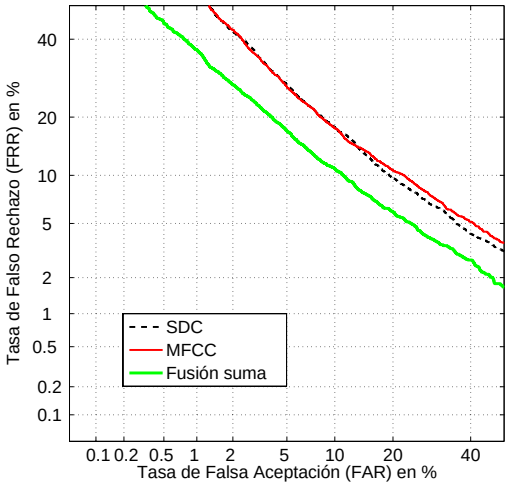


Figura 49. Curvas DET parametrización MFCC, SDC y fusión de ambas parametrizaciones

Color	Sistema	EER	DCF	EER modelo	EER test
Negro ---	SDC	13.84	0.061	14.86	10.28
Rojo	MFCC	14.07	0.061	16.18	10.33
Verde	Fusión suma	**10.55**	**0.045**	**11.60**	**6.82**

Tabla 57. Comparación resultados fusión suma MFCC y SDC, con sistemas individuales

La fusión deja el EER en un 10.55%, lo que supone una mejora del 33.4% con respecto a los subsistemas individuales. Los valores de DCF siguen una tendencia similar, siendo en este caso la mejora de un 26.8%. Este resultado hace que el sistema SVM-GLDS dedicado al reconocimiento de idioma mejore significativamente sus prestaciones, acercándose a los resultados de otro tipo de sistemas fonéticos que conforman el estado

del arte. Los resultados mostrados en esta sección constan de especial relevancia, ya que fueron incluidos como colaboración en un artículo presentado a Interspeech 2007 [Toledano *et al.,* 2007], dicho artículo se incluye en el apéndice.

10.10 Cálculo de SDC con mapping y warping

Como se explicó en la sección 5, los parámetros SDC son unos parámetros derivados de la parametrización MFCC. Por lo tanto primero deberemos obtener estos parámetros y a continuación calcular los SDC.

Hasta ahora, todos los experimentos realizados con la parametrización SDC partían de la misma base MFCC, un vector de 7 coeficientes, sin deltas, calculados con las normalizaciones CMN, rasta y mapping. En esta sección de los experimentos se probará el efecto de cambiar la normalización *mapping* por *warping*.

Para el entrenamiento de los modelos se emplearon 80 ficheros por idioma, de 30 minutos cada uno, procedentes de la base de datos Callfriend. Esta y otras características de los experimentos se resumen en la Tabla 58. La Figura 50 y la Tabla 59 muestran los resultados obtenidos por el sistema.

Evaluación	NIST LRE 2005 30s	**Normalización**	T-Norm
Datos entrenamiento	Callfriend 30m	**Compensación canal**	Ninguna
Parametrización	SDC (7-2-3-7)	**Tipo entrenamiento**	SVC

Tabla 58. Datos descriptivos experimentos SDC con mapping y warping

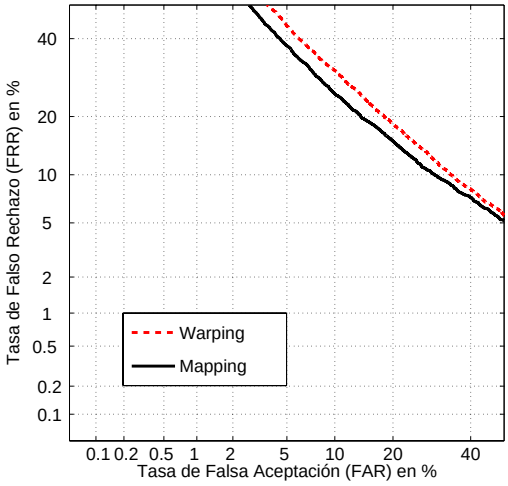


Figura 50. Curvas DET del sistema con parametrización SDC obtenida con mapping y warping

Color	Normalización	EER	DCF	EER modelo	EER test
Rojo ---	Warping	19.27	0.080	19.14	15.32
Negro	Mapping	**17.40**	**0.075**	**17.38**	**13.30**

Tabla 59. Comparación resultados sistema con parametrización SDC mapping y warping

Los resultados muestran una clara ventaja de la normalización *mapping* sobre la normalización *warping*. El EER con normalización *mapping* es un 9.8% más bajo que con normalización *warping*, el DCF sigue la misma tendencia, siendo en este caso la ventaja de *mapping* frente a *warping* de un 6.4%.

A la vista de los resultados obtenidos, podemos concluir que la normalización *warping* elimina parte de la información importante para el reconocimiento de idioma, por lo tanto será conveniente seguir con la normalización anterior.

10.11 Inclusión del vector MFCC en el vector SDC

Como colofón a la sección experimental de reconocimiento de idioma se realizará una última serie de experimentos relacionados con la parametrización de los datos. Hasta ahora se ha probado la parametrización SDC y la parametrización MFCC por separado, véase sección 10.2, además, en la sección 10.9 vimos los resultados de un sistema que fusionaba dos subsistemas basados en estas parametrizaciones.

En esta sección, se investigará el comportamiento de una parametrización ligeramente diferente a las anteriores. Esta nueva parametrización consiste en concatenar el vector de MFCC junto con el vector SDC [Castaldo *et al.,* 2007]. De esta manera tendremos un vector de 56 parámetros, 7 de vector MFCC y 49 del vector SDC, lo que tras el proceso de expansión se convierte en un vector de 30856 dimensiones.

La Tabla 58 muestra el escenario donde probaremos este nuevo tipo de parametrización.

Evaluación	NIST LRE 2005 30s	**Normalización**	T-Norm
Datos entrenamiento	Conjunto total	**Compensación canal**	Ninguna
Parametrización	MFCC + SDC	**Tipo entrenamiento**	SVC

Tabla 60. Datos descriptivos experimentos concatenación MFCC y SDC

Los resultados se comparan con la parametrización SDC, la que mejor resultado nos ha dado hasta ahora. La Figura 51 muestra las curvas DET de los sistemas y la Tabla 61 los valores numéricos.

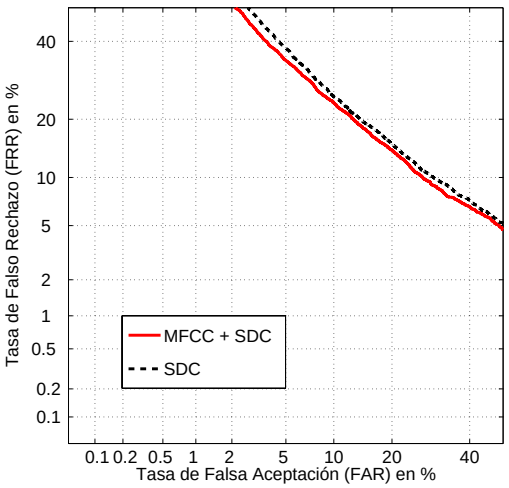


Figura 51. Curvas DET del sistema con el vector MFCC concatenado al SDC

Color	Parametrización	EER	DCF	EER modelo	EER test
Rojo	MFCC + SDC	**16.48**	**0.071**	**15.75**	**12.60**
Negro ---	SDC	17.40	0.075	17.37	13.30

Tabla 61. Comparación resultados sistema con vector MFCC concatenado al SDC

A la vista de los resultados podemos afirmar que el comportamiento del sistema mejora en todos las aspectos. Puede apreciarse una ganancia a lo largo de toda la curva DET, en EER esta ganancia equivale al 5%, en el caso del DCF la ganancia es del 6%.

11. Conclusiones y trabajo futuro

El presente proyecto se ha centrado en la investigación y mejora de los sistemas de reconocimiento biométrico y sistemas de reconocimiento de idioma basados en máquinas de vectores soporte. Uno de los objetivos principales era conseguir sistemas competitivos al nivel del estado del arte, con la restricción de que fuesen lo menos pesados computacionalmente como fuese posible.

En el campo del reconocimiento biométrico de locutor, el primer hito alcanzado fue la migración de la biblioteca empleada para el entrenamiento y test de los modelos. El cambio de *Torch* a *LibSVM* permitió obtener un sistema con rendimiento y prestaciones similares, además esta nueva biblioteca incluía la posibilidad de entrenar los modelos con regresión, uno de los resultados de investigación fundamentales del proyecto.

A parte de esta investigación, se realizaron experimentos en los que se examinó desde la influencia de ciertas variables en el comportamiento del sistema, hasta su rendimiento tras la implementación de distintos tipos de normalizaciones y compensaciones de variabilidad intersesión.

Las variables probadas fueron por un lado el coste del entrenamiento, llegando a la conclusión de que su única influencia en el sistema era sobre el tiempo empleado en el entrenamiento de los modelos. Por otro lado, se realizaron experimentos con los valores de las etiquetas de las clases del SVM, *Target y NonTarget.* Los resultados obtenidos mostraron la influencia de esa variable en el rendimiento del sistema, influencia que sería posteriormente observada en reconocimiento de idioma.

Las normalizaciones implementadas a nivel de puntuaciones fueron *T-Norm* y *Z-Norm*, las cuales mostraron una leve mejora en el comportamiento del sistema. También se realizaron otro tipo de normalizaciones orientadas a compensar la variabilidad intersesión, la técnica empleada para tal fin fue NAP, obteniendo una mejora de casi un 13% en el EER global.

A parte de estos experimentos se realizaron otros dirigidos a escalar los datos de entrada al sistema, para ello se emplearon funciones propias de la biblioteca y normalizaciones de rango *(Rank Normalization)* implementadas durante la realización de este proyecto. El rendimiento del sistema empeoró un 10% con la función de la propia biblioteca y un 16% con la normalización de rango. Estos resultados nos llevaron a pensar que el tipo de datos utilizado por nuestro sistema se veía fuertemente afectado por el escalado, por tanto no volvió a emplearse en ninguno de los experimentos sucesivos.

El entrenamiento de los modelos basado en regresión, *épsilon-SVR*, fue una de las investigaciones más importantes llevadas a cabo en el campo de reconocimiento de locutor en este proyecto. La estimación de una función que se ajustara a los datos en lugar de simplemente clasificarlos en base a su distancia al hiperplano, como hacía *SVC*, obtuvo unos resultados sensiblemente mejores. Además, se mostró como mediante el ajuste del parámetro ε éramos capaces de adaptarnos a la variabilidad de los datos (variabilidad intersesión, efectos del canal, etc.). La tarea realizada en las pruebas fue la *1conv-1conv* de NIST SRE 2006, obtenido mejoras en términos de EER del 34% y 29% para género masculino y femenino respectivamente.

Como colofón a los avances realizados durante la elaboración de este proyecto, destacaremos el resultado alcanzado mediante la fusión del sistema SVM y el sistema híbrido GMM-SVM. El sistema global conseguido presenta un EER del 4.4%, para el género masculino de la tarea mencionada anteriormente. Este resultado coloca al sistema global en una muy buena posición en el estado del arte actual.

En la otra línea de investigación llevada a cabo, reconocimiento automático de idioma basado en SVM, se siguió una política de experimentos similar a la mostrada hasta el momento. Sin embargo, los resultados no fueron tan satisfactorios como los logrados en reconocimiento de locutor. Hemos de tener en cuenta que este campo de investigación es relativamente nuevo en el grupo ATVS, si lo comparamos con el de reconocimiento de locutor, por lo que todavía queda mucho por avanzar en lo que a prestaciones del sistema se refiere.

La normalización de puntuaciones mediante *T-Norm* mejoró el comportamiento del sistema en mayor medida que como lo hiciera en reconocimiento de locutor. Por el contrario, la compensación de variabilidad intersesión y el entrenamiento de los modelos basados en regresión no presentaron el comportamiento esperado.

A raíz de los resultados resulta evidente que la técnica de compensación empleada, *NAP*, elimina información relevante del propio idioma a la vez que trata de compensar efectos de canal, ruidos, etc. Por otra parte, el principal problema de la regresión parece ser el número de vectores empleados en el entrenamiento, un número demasiado elevado hace que el sistema no sea capaz de ajustar la correspondiente función a los datos. Para tratar de solventar este problema se implementaron técnicas de agrupación de vectores, *cálculo del vector medio* y *K-means*, cuyo objetivo era disminuir el número de datos de entrada sin que ello llevara consigo una disminución de la información suministrada al sistema. Como se vio en las distintas pruebas las técnicas de agrupación implementadas reducían la información suministrada al sistema, haciendo que su rendimiento decreciera.

Uno de los hitos importantes en este campo se consiguió a través de la fusión del sistema trabajando con dos parametrizaciones distintas, *MFCC* y *SDC*. El sistema global obtuvo un EER del 10.5%, sobre la tarea de 30 segundos del protocolo de evaluación NIST LRE 2005, lo que supone una mejora del 33% con respecto a los subsistemas individuales. El resultado por si sólo no se encuentra en el estado del arte, pero fusionado con otros sistemas de alto nivel del grupo aporta información complementaria que da lugar a un sistema más robusto.

La regresión se presenta como la línea principal de trabajo en el campo de reconocimiento de locutor: uso de distintas técnicas basadas en regresión como por ejemplo υ-SVR [Schölkopf *et al.,* 2000], funciones de coste no lineales, diferentes tipos de *kernels*, etc. También resulta interesante la aplicación de las técnicas de regresión en otros sistemas basados en SVM, como por ejemplo el sistema de SuperVectors [Campbell *et al.*, 2006a]. Por último, siguiendo esta misma línea de trabajo, la investigación en sistemas con distintos valores de etiquetas, en función de la distancia al hiperplano de separación de las muestras sería muy interesante.

Una línea futura de trabajo en ambos campos sería probar nuevas técnicas de compensación, una de las más populares en este momento es la compensación de canal

mediante *Joint Factor Análisis* [Kenny y Dumouchel, 2004; Vogt y Sridharan, 2006; Vair *et al.,* 2006]. Esta técnica consiste básicamente en detección y compensación de direcciones de máxima variabilidad en un espacio de características de muy altas dimensiones.

Centrándonos en el campo de reconocimiento de idioma, se está trabajando en desarrollar sistemas *PhoneSVM* [Campbell *et al.,* 2004a], sistemas que aprovechan la capacidad discriminativa de los SVM para la separación de características del idioma, basándose en n-gramas. También se está implementando un sistema basado en GMM y SVM, como ya se hizo para reconocimiento de locutor, de esta forma aprovecharíamos las características del modelado generativo de los sistemas GMM, así como el modelado discriminativo de los SVM.

Por último, la implementación de sistemas dependientes de género reduce la variabilidad del idioma y hace que el modelado sea más sencillo, por lo que se implementará en un futuro próximo. Un problema que se deberá tener en cuenta es que el género del locutor puede ser desconocido a priori, como ocurre en las evaluaciones de NIST. La identificación de género lleva consigo un determinado porcentaje de error que podría degradar el rendimiento del sistema.

Parte de estos trabajos han sido recogidos y publicados en [Lopez-Moreno *et al.,* 2007] y [Toledano *et al.,* 2007].

12. Referencias

R. Auckenthaler, M. Carey y H. Lloyd-Thomas. Score normalization for text-independent speaker verification systems. *Digital Signal Processing*, V10, pp. 42-54, 2000.

F. Bimbot et al. A tutorial on text-independent speaker verification. *Journal on Applied Signal Processing*, N. 4, pp. 430-451, 2004.

C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, V2, pp. 121-167, 1998.

W. M. Campbell. Generalized linear discriminate sequence kernels for speaker recognition. *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Vol. 1, pp. 161-164, 2002.

W. M. Campbell, J. R. Campbell, D. A. Reynolds, D. A. Jones y T. R. Leek. High-level speaker verification with support vector machines. *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP),* Vol. 1, N. 17-21, pp. 73-76, Mayo 2004a.

J. P. Campbell, H. Nakasone, C. Cieri, D. Miller, K. Walker, A. F. Martin, y M. A. Przybocki. The MMSR bilingual and crosschannel corpora for speaker recognition research and evaluation. *Proc. IEEE Speaker and Language Recognition Workshop (Odyssey),* pp. 29-32, 2004b.

W. M. Campbell, D. E. Sturim y D. A. Reynolds. Support vector machines using GMM supervectors for speaker verification. *Signal Processing Letters, IEEE,* Vol. 13, N. 5, pp. 308-311, Mayo 2006a.

W. M. Campbell, J. P. Campbell, D. A. Reynolds, E. Singer, y P. A. Torres-Carrasquillo. Support vector machines for speaker and language recognition. *Computer Speech and Language*, Vol. 20, N. 2-3, pp. 210-229, 2006b.

P. Carr. English Phonetics and Phonology: An Introduction. *Blackwell Publishing*, 1999.

F. Castaldo, E. Dalmasso, P. Laface, D. Colibro y C. Vair. Language identification using acoustic models and speaker compensated cepstral-time matrices. *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Vol. 4, pp. 1013-1016, Abril 2007.

N. Cristianini y J. Shawe-Taylor. An introduction to support vector machines and other kenerl-based learning methods. *Cambridge University Press*, 2000.

J. R. Deller, J. H. L. Hansen y J. G. Proakis. Discrete-time processing of speech signals. *Wiley-IEEE Press*, Septiembre 1999.

G. Doddington. Speaker recognition based on idiolectal differences between speakers, *Proc. Eurospeech,* pp. 2512-2524, 2001.

R. O. Duda, P. E. Hart y D. G. Store. Patter Classification. *Wiley*. 2001.

J. Fierrez-Aguilar, J. Ortega-García, D. García-Romero y J. González-Rodríguez. A comparative evaluation of fusion strategies for multimodal biometric verification. *Proc. 4th IAPR Intl. Conf. on Audio and Video Based Person Authentication AVBPA*, pp. 830-837, Junio 2003.

J. Fierrez-Aguilar, D. Garcia-Romero, J. Ortega-Garcia y J. Gonzalez-Rodriguez. Speaker verification using adapted user-dependent multilevel fusion. *Proc. 6th IAPR Intl. Workshop on Multiple Classifier Systems*, MCS, Springer LNCS-3541, pp. 356-365, 2005.

S. Furui. Cepstral Analysis technique for automatic speaker verification. *IEEE Transactions on acoustics, speech and signal processing,* Vol. ASSP-29, N. 2, Abril 1981.

A. A. Garcia y R. J. Mammone. Channel-robust speaker identification using modified-mean cepstral mean normalization with frecuency warping. *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Vol. 1, pp. 325-328, 1999.

J. Gonzalez-Rodriguez, D. Ramos-Castro, D. Torre-Toledano, A. Montero-Asenjo, J. Gonzalez-Dominguez, I. Lopez-Moreno, J. Fierrez-Aguilar, D. Garcia-Romero y J. Ortega-Garcia. On the use of high-level information for speaker recognition: the ATVS-UAM system at NIST SRE 2005. *IEEE Aerospace and Electronic Systems Magazine*, pp. 15-21, Enero 2007.

A. O. Hatch, B. Peskin y A. Stolcke. Improved phonetic speaker recognition using lattice decoding. *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Vol. 1, pp. 169-172, Marzo 2005.

H. Hermansky y N. Morgan. Rasta processing of speech. *IEEE Transactions on Speech and Audio Processing 2*, pp. 578-589, Octubre 1994.

A. K. Jain, A. Ross y S. Prabhakar. An introduction to biometric recognition. *IEEE Transactions on Circuits Systems for Video Technology*, Vol. 14, N.1, pp. 4-20, 2004.

P. Kenny y P. Dumouchel. Disentangling speaker and channel effect in speaker verification. *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Vol. 1, pp. 37-40, 2004.

N. Krause y R. Gazit. SVM-based speaker classification in the GMM model space. *Proc. IEEE Speaker and Language Recognition Workshop (Odyssey)*, pp. 1-5, 2006.

LDC, *(Linguistic Data Consortium)*. Descripción de distintas bases de datos. www.ldc.upenn.edu

LibSVM: C. C. Chang y C. J. Lin. A library for support vector machines. *Software disponible en http://www.csie.ntu.edu.tw/~cjlin/libsvm*, 2001.

I. Lopez-Moreno, I. Mateos-García, D. Ramos y J. Gonzalez-Rodriguez. Support vector regression for speaker verification. *Proc. Interspeech*, Agosto 2007.

D. Maltoni, D. Maio, A. K. Jain y S. Prabhakar. Handbook of Fingerprint Recognition, *Springer* 2003.

A. Martin, G. Doddington, T. Kamm, M. Ordowski y M. Przybocki. The DET curve in assessment of decision task performance. *Proc. EuroSpeech*, pp. 1895-1898, 1997.

K. Muller, A. J. Smola, G. Ratsch, B. Scholkopf, J. Kohlmorgen, y V. Vapnik. Predicting time series with support vector machines. *Proc. of the 7th International Conference on Artificial Neural Networks*, Vol. 1327 of *Lecture Notes In Computer Science*, pp. 999-1004, 1997.

NIST LRE. Descripciones de las distintas evaluaciones NIST de idioma http://www.nist.gov/speech/tests/lang

NIST SRE. Descripciones de las distintas evaluaciones NIST de locutor http://www.nist.gov/speech/tests/spk

Y. Obuchi y N. Sato. Language identification using phonetic and prosodic HMMs with feature normalization. *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Marzo 2005.

J. Pelecanos y S. Sridharan. Feature warping for robust speaker verification. *Proc. IEEE Speaker and Language Recognition Workshop (Odyssey)*, pp. 213-218, 2001.

L. R. Rabiner, A. E. Rosemberg y S. E. Levinson. Considerations in dynamic time warping algorithms for discrete word recognition. *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Vol. ASSP-26, pp. 575-582, Diciembre 1978.

L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition, *Proc. IEEE*, Vol. 77, N. 2, pp. 257-286, Febrero 1989.

D. A. Reynolds, T. F. Quatieri, y R. B. Dunn. Speaker verification using adapted gaussian mixture models. *Digital Signal Processing 10,* pp. 19-41, 2000.

D. A. Reynolds et al. Supersid project: Exploiting high-level information for high accuracy speaker recognition. *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Vol. 4, pp 784-787, Abril 2003.

D. A. Reynolds. An overview of speaker recognition technology. *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4072–4075, 2003a.

D. A. Reynolds. Channel robust speaker verification via feature mapping. *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Vol. 2, pp. 53-56, Abril 2003b.

B. Schölkopf, C. J. C. Burges y A. J. Smola. Advances in kernel methods and support vector learning. *MIT Press*, 2000.

A. J. Smola y B. Schoelkopf. A tutorial on support vector regression. *Tech. Rep. NeuroCOLT2 Technical Report NC2-TR-1998-030*, Royal Holloway College, University of London, UK, 1998.

A. Solomonoff, C. Quillen, y W.M. Campbell. Channel compensations for SVM speaker recognition. *Proc. IEEE Speaker and Language Recognition Workshop (Odyssey),* pp. 57-62, 2004.

A. Solomonoff, W.M. Campbell y I. Boardman. Advances channel compensations for SVM speaker recognition. *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP),* Vol. 1, pp. 629-632, 2005.

P. Sollich. Probabilistic methods for support vector machines. *Advances in Neural Information Processing Systems*, S. A. Solla, T. K. Leen, and K. Müller, Eds., Vol. 12, pp. 349-355. MIT Press, 1999.

A. Stolcke, L. Ferrer, S. Kajarekar, E. Shriberg y A. Venkataraman. MLLR transforms as features in speaker recognition. *Proc. Eurospeech,* pp. 2425-2428, Septiembre 2005.

D. T. Toledano, J. Gonzalez-Dominguez, A. Abejón-Gonzalez, D. Spada, I. Mateos-García y J. Gonzalez-Rodriguez. Improved language recognition using better phonetic decoders and fusion with MFCC and SDC features. *Proc. Interspeech*. Agosto 2007.

Torch: R. Collobert, S. Bengio y J. Mariéthoz, IDIAP. *Software disponible en* http://www.torch.ch

P. A. Torres-Carrasquillo, E. Singer, M. A. Kohler, R. J. Greene, D. A. Reynolds y J. R. Deller Jr. Approaches to language identification using gaussian mixture models and shifted delta cepstrum. *Proc. International Conference on Spoken Language (ICSLP)*, 2002.

C. Vair, D. Colibro, F. Castaldo, E. Dalmasso y P. Laface. Channel factors compensation in model and feature domain for speaker recognition. *Proc. IEEE Speaker and Language Recognition Workshop (Odyssey)*, pp. 1-6, Junio 2006.

V. N. Vapnik. The nature of statistical learning theory. *Springer, second edition, 1995.*

R. Vogt y S. Sridharan. Experiments in session variability modelling for speaker verification. *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Vol. 1, pp. 897-900, 2006.

M. Wagner, C. Summerfield, T. Dunstone, R. Summerfield y J. Moss. An evaluation of "commercial off-the-shelf" speaker verification systems. *Proc. IEEE Speaker and Language Recognition Workshop (Odyssey)*, 2006.

V. Wan y W. Campbell. Support vector machines for speaker verification and identification. *Proc. IEEE Workshop on Neural Networks for Signal Processing*, Vol. 2, pp. 775-784, 2000.

M. A. Zissman. Comparison of four approaches to automatic language identification of telephone speech. *IEEE Transactions on Speech and Audio Processing*, pp. 31-44, Enero 1996.

13. Apéndice

A continuación se incluyen los artículos mencionados anteriormente en los que se recogen parte de los resultados obtenidos en el presente proyecto:

- I. Lopez-Moreno, I. Mateos-García, D. Ramos y J. Gonzalez-Rodriguez. Support vector regresión for speaker verification. *Proc. Interspeech*, Agosto 2007.

- D. T. Toledano, J. Gonzalez-Dominguez, A. Abejón-Gonzalez, D. Spada, I. Mateos-García y J. Gonzalez-Rodriguez. Improved language recognition using better phonetic decoders and fusion with MFCC and SDC features. *Proc. Interspeech*. Agosto 2007.

Support Vector Regression for Speaker Verification

Ignacio Lopez-Moreno, Ismael Mateos-Garcia,
Daniel Ramos and Joaquin Gonzalez-Rodriguez

ATVS (Biometric Recognition Group), C/ Francisco Tomas y Valiente 11,
Universidad Autonoma de Madrid, E28049 Madrid, Spain
{ignacio.lopez, ismael.mateos, daniel.ramos, joaquin.gonzalez}@uam.es

Abstract

This paper explores Support Vector Regression (SVR) as an alternative to the widely-used Support Vector Classification (SVC) in GLDS (Generalized Linear Discriminative Sequence)-based speaker verification. SVR allows the use of a ε-insensitive loss function which presents many advantages. First, the optimization of the ε parameter adapts the system to the variability of the features extracted from the speech. Second, the approach is robust to outliers when training the speaker models. Finally, SVR training is related to the optimization of the probability of the speaker model given the data. Results are presented using the NIST SRE 2006 protocol, showing that SVR-GLDS yields a relative improvement of 31% in EER compared to SVC-GLDS.

Index Terms: speaker verification, GLDS, SVM classification, SVM regression

1. Introduction

Speaker verification has been dominated in the last decade by systems working at the spectral level of the speaker identity [1]. Techniques like Gaussian Mixture Models (GMM) [2] or Support Vector Machines (SVM) using Generalized Linear Discriminant Sequence (GLDS) kernels [3] have demonstrated its superiority to higher level approaches [1, 4]. In recent years, hybrid approaches such as GMM-SVM systems [5] and channel compensation techniques like factor analysis [6] or nuisance attribute projection [7] have led to a significant improvement of the state-of-the-art performance.

One of the techniques which have yielded a good performance at the spectral level is SVM-GLDS speaker verification [3]. Using this technique, parameters are mapped to a high-dimensional space via a GLDS kernel function. Then, a SVM classifier is used in order to discriminate genuine users from impostors at that high dimensional space. The performance of SVM-GLDS speaker verification systems has demonstrated to be similar to the GMM modelling. Also, the fusion of SVM-GLDS classification with other approaches at the spectral level significantly improves performance [3].

SVMs have demonstrated their efficiency and accuracy in solving two main problems: *i)* discriminating among classes (classification) and *ii)* function estimation (regression). In the former the objective is to compute a class for every feature extracted from the data. In the latter the aim is finding a good approximation to a function of the features. In this sense, regression is a more general approach than classification, as a class label is indeed a function of the features. As speaker verification is essentially a binary class problem, most popular schemes are based on SVM classifiers (SVC). However, as we will show, a more general and robust approach can be adopted by using SVM regression (SVR). In this paper we propose the use of SVR for speaker verification using a GLDS kernel. Reported results using NIST SRE 2006 experimental protocol show a significant improvement of SVR-GLDS versus SVC-GLDS.

This work is organized as follows. SVM classification and regression is introduced in Section 2, highlighting their main differences. SVM regression for GLDS speaker verification (SVR-GLDS) is presented in Section 3. In Section 4, Experiments showing the adequacy of the proposed technique are presented. Finally, conclusions are drawn in Section 5.

2. Support Vector Machine Classification and Regression

SVM derive from the Vapnik's statistical learning theory [8], and since 1994 they have been largely used for pattern recognition due to its excellent generalization properties. For instance, a well known effect of SVM is that the number of observations and its dimensionality do not affect to SVM generalization [9]. These properties, added to the efficiency and elegance of kernel methods [10], make SVM giving an excellent performance in many different tasks. The good discrimination of SVM-based speaker verification systems [3, 5] supports this fact.

In this section we describe the use of Support Vector Machines for both classification and regression. We compare both methods and we highlight the main differences between them.

2.1. Support Vector Machine Classification (SVC)

Suppose we have l vectors $x_i \in \mathbb{R}^n$ from two different classes. Each class is labelled as $y_i \in [+1, -1]$. The classification problem consist in assigning each x_i to its corresponding class y_i. The SVC approach finds an optimal hyperplane $\mathbf{w}$ which separates $\mathbb{R}^n$ in two regions: vectors in one of the regions will be assigned to the class $+1$ and the rest to the class -1. We define the scoring function $f(\cdot) : \mathbb{R}^n \rightarrow \mathbb{R}$, which measures the distance of each vector to the separating hyperplane $\mathbf{w}$:

$$f(x) = \langle \mathbf{w}, x \rangle + b \tag{1}$$

where b is a learned offset parameter. If the data set $D = \{(x_1, y_1), (x_2, y_2) \cdots (x_l, y_l)\}$ is linearly separable, $f(\cdot)$ will be positive for al values of x_i where $y_i = +1$ and negative otherwise.

However, there are many effects which may cause overlapping between classes, e. g. noise, channel effects, intra- and

This work was partially funded by the Spanish Ministry of Education under project TEC2006-13170-C02-01.

inter- class variability, etc. Therefore, some vectors will be incorrectly classified. In this case, we will have two different criteria for finding $\mathbf{w}$: *i)* maximizing the margin between classes and *ii)* minimising a loss function proportional to missclassificated vectors. A weighting factor C controls the relevance of one criteria against the other, as it can be seen in the following formula:

$$\begin{array}{l} \mathbf{w} = \arg\min_{\mathbf{w}} \left(\frac{1}{2}\mathbf{w}^{\mathrm{T}} \cdot \mathbf{w} + C\frac{1}{m}\sum \xi_{c,i}\right) \\ \text{subject to} \quad 0 \leq \xi_{c,i} \leq 1 - y_i f\left(x_i\right) \end{array} \tag{2}$$

Here, $\xi_{c,i}$ is a slack variable associated to the non-optimally classificated vector i $(i \in \{1, ...m\})$ in a classification problem, and it will only be non-zero for those x_i which make $y_i \cdot f(x_i) < 1$. Notice that if $0 < yi \cdot f(x_i) < 1$, x_i will be correctly classified but its associated $\xi_{c,i}$ value will be different to 0. Thus, for classification problems the loss function is defined as:

$$f_{loss}(x_i) = max\{0, 1 - y_i \cdot f(x_i)\} \tag{3}$$

Non-linear classification can be solved by using $\phi(x_i)$ instead of x_i. The function $\phi(\cdot)$ maps each vector to a higher dimensional feature space where vectors are linearly separable. As SVM only require the inner product of the vectors in the features space $\langle\phi(x_i), \phi(x_j)\rangle$, we define the kernel function as:

$$k(x_i, x_j) = \langle\phi(x_i), \phi(x_j)\rangle \tag{4}$$

The kernel function $k(x_i, x_j) \in \mathbb{R}$ allows us to compute $\langle\phi(x_i), \phi(x_j)\rangle$ without explicitly mapping each vector into the high dimensionality space. This is known as the kernel trick.

2.2. Support Vector Machine Regression

In the regression problem, y_i is not a class but any other function of x_i. Therefore SVR can be used to learn $n-$dimensional functions $g_n(\cdot)$ such as

$$g_n(x_i) = y_i \tag{5}$$

The goal in the regression problem is to approximate $f(\cdot) \simeq g_n(\cdot)$. Notice that, although $g_n(\cdot)$ can take either continuous or discrete values, the SVR approximation will always be a continuous function. In the SVR case, the C parameter is used to control how much we need to approximate $f(\cdot)$ to $g_n(\cdot)$.

Regarding the loss function, the main difference of regression with respect to classification is that errors are penalized not only when $f(\cdot) < g_n(\cdot)$ but also when $f(\cdot) > g_n(\cdot)$. Therefore, the loss function has to be modified in order to take a different behavior than in the classification case because, for classification errors, this penalty was only applied when $y_i \cdot f(x_i) - 1 < 0$.

A popular loss function for regression is the ε-insensitive loss function [11]. This function tolerates some degree of mismatch by the use of an margin controlled by the ε parameter. As errors only occur when $|f(\cdot) - g_n(\cdot)| > \varepsilon$, the SVR training goal is to find $\mathbf{w}$ such as:

$$\begin{array}{l} \min\left(\frac{1}{2}\mathbf{w}^{\mathrm{T}} \cdot \mathbf{w} + C\frac{1}{m}\sum\left(\xi_{r,i} + \xi'_{r,i}\right)\right) \\ \text{subject to} \quad \left\{\begin{array}{l} 0 \leq f\left(x_i\right) - y_i - \varepsilon \leq \xi_{r,i} \\ 0 \leq y_i - f\left(x_i\right) - \varepsilon \leq \xi'_{r,i} \end{array}\right. \end{array} \tag{6}$$

As it can be seen, two different slack variables are introduced for regression: $\xi_{r,i}$ for those vectors for which $f(x_i) > g_n(x_i) + \varepsilon$, and $\xi'_{r,i}$ for those ones that $f(x_i) < g_n(x_i) - \varepsilon$. The loss function is now defined as:

$$f'_{loss}(x_i) = max\left\{0, |y_i - f(x_i)| - \varepsilon\right\} \tag{7}$$

Figure 1 illustrates $f'_{loss}(\cdot)$ and its differences with $f_{loss}(\cdot)$.

An interesting property of SVR which does not apply for SVC is that the ε-insensitive loss function leads to a maximum-a-posteriori (MAP) estimation of $\mathbf{w}$ [12]. It can be shown that $e^{-f'_{loss}(\cdot)}$ is proportional to $p(\mathbf{w}|D, \varepsilon)$, i. e. the posterior probability of $\mathbf{w}$ given the data and the value of the ε margin. Therefore, by minimizing $f'_{loss}(\cdot)$ we will maximize the log-probability that $f(\cdot) = g_n(\cdot)$.

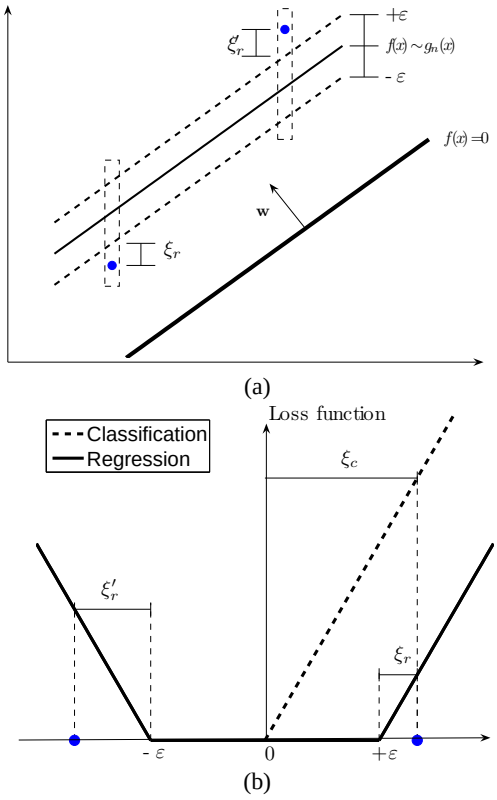


Figure 1: SVR versus SVC. Boundaries (a) and loss function (b). The loss functions are centered at $f(x_i) = y_i$ for SVC (f_{loss}) and at $f(x_i) = g_n(x_i)$ for SVR (f'_{loss}). f_{loss} penalizes x_i such as $y_i \cdot f(x_i) - 1 < 0$, while f'_{loss} penalizes x_i such as $|f(x_i) - g_n(x_i)| > \varepsilon_i$.

3. SVR-GLDS Speaker Verification

Speaker verification is a two-class classification problem. The objective is to take a decision about if a testing utterance corresponds to a claimed identity or not. In widely used SVC-GLDS speaker verification, for each SVM speaker model, the class label will take the value 1 for the target vectors belonging to the speaker and -1 for nontarget vectors from anyone else.

Our proposal is to use a SVR with an ε-insensitive loss function for classification. Thus, the SVR goal function $g_n(\cdot)$ is discrete and it only takes two different values, namely $g_n(\cdot) \in [+1, -1]$ for target and nontarget speakers respectively. Note that for this problem, the support vectors will not be the nearest ones to $\mathbf{w}$, as in classification, because in such case they

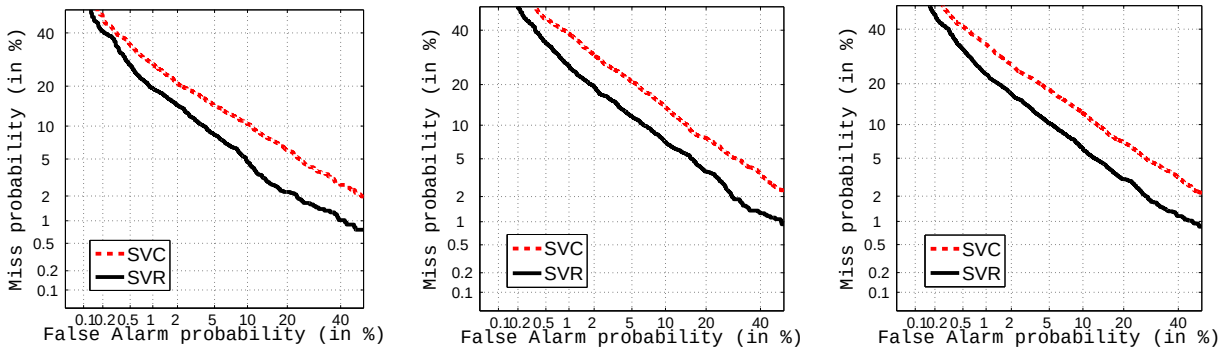


Figure 2: Comparison of SVC-GLDS and SVR-GLDS in NIST SRE 2006 ($\varepsilon = 0.1$) 1conv4w-1conv4w task for male (a), female (b) and pooled gender (c) data.

would minimize $f_{loss}(\cdot)$, but they would not minimize $f'_{loss}(\cdot)$. This difference from the standard SVC makes SVR more robust against outliers or noisy vectors being used for obtaining $\mathbf{w}$, because in SVR the support vectors are selected form regions in the feature space where vectors of each class are more concentrated. On the other hand, SVC uses a set of support vectors which are nearer the frontier between classes, where vectors of each class use to be scarce. Thus, SVC hyperplane may be more sentsitive than SVR to outliers in the support vectors.

Finally, an optimal training ε-insensitive SVR requires adequate tuning of C and ε parameters. Some works in the literature [13] relate the ε parameter to the noise or variability of the function to estimate. Therefore, the optimal value of ε allows us to obtain a quantitative measure of the feature variability in speaker verification problems. Moreover, optimizing the ε parameter adapts the SVR training process to the observed variability in the data.

4. Experiments

4.1. Baseline system

Our baseline system is a SVM-GLDS speaker recognition system as described in [3]. Feature extraction obtains 19 MFCC coefficients plus deltas. In order to avoid channel mismatch effects, cepstral mean normalization is applied, followed by RASTA filtering and feature mapping (see [4] for details). The similarity computation is based on SVC [3]. A GLDS kernel expansion is performed on the whole observation sequence, and a separating hyperplane is computed between the speaker features and the background model. The system uses a polynomial expansion of degree three [14] prior to the application of the GLDS kernel. We have used the LibSVM library [15] for both SVM classification and regression. Finally, Tnorm [16] score normalization technique is performed in order to scale the scores distribution.

4.2. Database and experimental protocol

Experiments have been performed using the evaluation protocol proposed by NIST in its 2006 Speaker Recognition Evaluation (SRE) [17]. The database used in this evaluation consists of: *i)* a subcorpus of the MIXER database [18] and *ii)* a significant amount of additional multi-channel and multi-language data acquired in order to complete the corpus for the evaluation. The acquisition conditions include different communication channels (landline, GSM, CDMA, etc.), different handsets and microphones (carbon button, electret, earphones, cordless, etc.) and different languages (American English, Arabic, Spanish, Mandarin, etc.). The evaluation protocol defines the following training conditions: 10 seconds, 1, 3 and 8 conversation sides; and the following test conditions: 10 seconds, 1 conversation side, 3 full conversations in a mixed channel and multichannel microphone data. Each conversation side has an average duration of 5 minutes, with 2.5 minutes of speech on average after silence removal. Although there are speakers of both genders in the corpus, no cross-gender trials are defined. Details can be found in the NIST webpage (www.nist.gov/speech). In our case the experiments followed the 1 conversation side training conditions, and 1 conversation side test condition (1conv4w-1conv4w). The background set for system tuning is a subset of databases from previous NIST SREs. Trials performed using this development set follow the corresponding NIST SRE protocol. The Tnorm cohorts were extracted from the NIST 2005 SRE targets models for each training condition.

4.3. Results

First of all, we have investigated the variation of the performance of the proposed SVR-GLDS system with respect to the parameter ε as defined in Section 2.2 (Equation 6). Tables 1 and 2 show the performance for different values of ε. Results are presented both as Equal Error Rate (EER) and DCF_{min} as defined by NIST [17]. It is observed that the performance of the system significantly improves for values around $\varepsilon = 0.1$, both for EER and DCF values. Therefore, the value $\varepsilon = 0.1$ will be used for SVR for the experiments presented below.

ε	0.01	0.05	0.1	0.2	0.4	0.8
EER(%)	9.1	7.8	**6.9**	8.4	9.9	10.3
$DCF_{min} \cdot 10^2$	3.5	3.2	**2.9**	3.5	3.7	3.7

Table 1: EER and DCF_{min} in NIST SRE 2006 male 1conv4w-1conv4w, for different values of ε.

We have also evaluated the performance of SVR-GLDS versus the SVC-GLDS baseline system. Table 3 shows the differences between them in terms of EER and DCF_{min}. It is

ε	0.01	0.05	0.1	0.2	0.4	0.8
EER(%)	11	8.6	**8.5**	9.7	11.9	12
$DCF_{min} \cdot 10^2$	4.1	**3.5**	3.6	4.2	4.7	4.8

Table 2: EER and DCF_{min} in NIST SRE 2006 female 1conv4w-1conv4w, for different values of ε.

shown that SVR obtains a relative improvement in EER of 34% for male, and 29% for female, whereas the relative improvement of the DCF_{min} value is 22% and 25% in the male and female cases respectively. Finally, Figure 2 shows the discrimination performance of SVR-GLDS versus SVC-GLDS for the male, female and pooled gender cases. We can observe a significant performance improvement at all operating points in the DET curve for all gender conditions.

	Male		Female		Pooled	
	SVC	SVR	SVC	SVR	SVC	SVR
EER(%)	10.4	**6.9**	12	**8.5**	11.3	**7.8**
DCF	3.7	**2.9**	4.8	**3.6**	4.3	**3.3**

Table 3: SVC-GLDS and SVR-GLDS systems in NIST SRE 2006 1conv4w-1con4w task. It shows the EER(%) and $DCF_{min} \cdot 10^2$ for male and female genders.

5. Conclusions

In this paper we have presented a Support Vector Machine Regression (SVR) approach for speaker verification in the GLDS kernel space. This technique presents advantages with respect to Support Vector Machine Calssification (SVC). First, the loss function used is related to the variability present in the feature space. Thus, varying the parameter ε we can adapt to such variation, which may be due to inter-session variability (e. g., channel mismatch) or intra-speaker variability. Second, the technique is more robust against outliers. Finally, the regression technique optimizes the posterior probability of the model $\mathbf{w}$ given the data and the ε parameter. Reported results have demonstrated the adequacy of support vector regression (SVR) for speaker verification with a GLDS kernel function, as significant improvements are shown both in EER and DCF_{min}. Future work includes the use of different SVR approaches for the GLDS space, such as ν-SVR [10], non-linear loss functions and different kernels. Also, the application of SVR to other SVM-based speaker recognition systems as GMM Supervectors [5], and non singular class labelling will be considered. Finally, the proposed technique will be tested in different databases in order to explore its robustness to environmental changes.

6. References

[1] D. A. Reynolds, "An overview of speaker recognition technology," in *Proc. of ICASSP*, 2003, pp. 4072–4075.

[2] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, "Speaker verification using adapted gaussian mixture models," *Digital Signal Processing*, vol. 10, pp. 19–41, 2000.

[3] W. M. Campbell, J. P. Campbell, D. A. Reynolds, E. Singer, and P. A. Torres-Carrasquillo, "Support vector machines for speaker and language recognition," *Computer Speech and Language*, vol. 20, no. 2-3, pp. 210–229, 2006.

[4] J. Gonzalez-Rodriguez, D. Ramos-Castro, D. Torre-Toledano, A. Montero-Asenjo, J. Gonzalez-Dominguez, I. Lopez-Moreno, J. Fierrez-Aguilar, D. Garcia-Romero, and J. Ortega-Garcia, "On the use of high-level information for speaker recognition: the ATVS-UAM system at NIST SRE 2005," *IEEE Aerospace and Electronic Systems Magazine*, pp. 15–21, January, 2007.

[5] W. M. Campbell, D. E. Sturim, and D. A. Reynolds, "Support vector machines using GMM supervectors for speaker verification," *Signal Processing Letters*, vol. 13(5), pp. 308–311, 2006.

[6] P. Kenny and P. Dumouchel, "Disentangling speaker and channel effects in speaker verification," in *Proc. of ICASSP*, 2004, vol. 1, pp. 37–40.

[7] A. Solomonoff, W. M. Campbell, and I. Boardman, "Advances in channel compensation for svm speaker recognition," in *Proc. of ICASSP*, 2005, pp. 629–632.

[8] V. Vapnik, *The Nature of Statistical Learning Theory*, Springer-Verlag, 1999.

[9] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[10] B. Schölkopf, C. J. C. Burges, and A. J. Smola, *Advances in Kernel Methods and Support Vector Learning*, MIT Press, 2000.

[11] K. Muller, A. J. Smola, G. Ratsch, B. Scholkopf, J. Kohlmorgen, and V. Vapnik, "Predicting time series with support vector machines," in *Proc. of the 7th International Conference on Artificial Neural Networks*, 1997, vol. 1327 of *Lecture Notes In Computer Science*, pp. 999–1004.

[12] P. Sollich, "Probabilistic methods for support vector machines," in *Advances in Neural Information Processing Systems*, S. A. Solla, T. K. Leen, and K. Müller, Eds., vol. 12, pp. 349–355. MIT Press, 1999.

[13] A.J. Smola and B. Schoelkopf, "A tutorial on support vector regression," Tech. Rep. NeuroCOLT2 Technical Report NC2-TR-1998-030, Royal Holloway College, University of London, UK, 1998.

[14] W. Wan and W. Campbell, "Support vector machines for speaker verification and identification," in *Proc. of IEEE International Workshop on Neural Networks for Signal Processing*, 2000, pp. 775–784.

[15] Chih-Chung Chang and Chih-Jen Lin, *LIBSVM: a library for support vector machines*, 2001, Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[16] R. Auckenthaler, M. Carey, and H. Lloyd-Tomas, "Score normalization for text-independent speaker verification systems," *Digital Signal Processing*, vol. 10, pp. 42–54, 2000.

[17] NIST, "2006 speaker recognition evaluation plan: http://www.nist.gov/speech/tests/spk/2006/index.htm," 2006.

[18] J. P. Campbell, H. Nakasone, C. Cieri, D. Miller, K. Walker, A. F. Martin, and M. A. Przybocki, "The MMSR bilingual and crosschannel corpora for speaker recognition research and evaluation," in *Proc. of Odyssey*, 2004, pp. 29–32.

IMPROVED LANGUAGE RECOGNITION USING BETTER PHONETIC DECODERS AND FUSION WITH MFCC AND SDC FEATURES

Doroteo T. Toledano, Javier Gonzalez-Dominguez, Alejandro Abejon-Gonzalez, Danilo Spada, Ismael Mateos-Garcia and Joaquin Gonzalez-Rodriguez

ATVS Biometric Recognition Group, Universidad Autónoma de Madrid, Spain

{javier.gonzalez, doroteo.torre, joaquin.gonzalez}@uam.es

Abstract

One of the most popular and better performing approaches to language recognition (LR) is Parallel Phonetic Recognition followed by Language Modeling (PPRLM). In this paper we report several improvements in our PPRLM system that allowed us to move from an Equal Error Rate (EER) of over 15% to less than 8% on NIST LR Evaluation 2005 data still using a standard PPRLM system. The most successful improvement was the retraining of the phonetic decoders on larger and more appropriate corpora. We have also developed a new system based on Support Vector Machines (SVMs) that uses as features both Mel Frequency Cepstral Coefficients (MFCCs) and Shifted Delta Cepstra (SDC). This new SVM system alone gives an EER of 10.5% on NIST LRE 2005 data. Fusing our PPRLM system and the new SVM system we achieve an EER of 5.43% on NIST LRE 2005 data, a relative reduction of almost 66% from our baseline system.

Index Terms: Language recognition, PPRLM, SVM.

1. Introduction

Automatic Language Recognition (LR) tries to recognize the language of a particular speech segment and is usually a first step for further processing the speech segment either manually (sending the speech segment to an operator proficient in the language) or automatically (sending it to an adequate automatic dialogue manager). The last years have shown an important growth in the field, resulting in a rise in the number of sites participating in the LR evaluations organized by NIST [1].

Along the evolution of automatic LR the most widely used and successful approach to LR has been Phone Recognition followed by Language Modeling (PRLM) and Parallel PRLM (PPRLM) [2, 3]. More recently PPRLM systems have been improved further by processing the whole lattice instead of just the 1-best solution produced by the phonetic decoders [4, 5] and substituting the statistical language modeling scoring by Support Vector Machines (SVMs) taking as input vectors the n-grams [6]. In this paper we will not take into account these possibilities for improvement. Rather we will concentrate on *classical* PPRLM systems and try to improve their performance as much as possible as a first step to then make further improvements using lattice decoding and SVMs. In the process we will analyze the influence on LR results of several improvements over the baseline system [7] we submitted to NIST LRE 2005.

PPRLM systems can be complemented with other types of systems possibly operating on different features. In this paper we complement our improved PPRLM system with an SVM system operating on MFCC and SDC acoustic features.

In section 2 we describe our baseline system presenting results on data taken from NIST LRE 2003. The following sections (3, 4 and 5) will analyze the influence on PPRLM performance of the use of a different parameterization, an explicit Voice Activity Detector (VAD) and phonetic models trained on larger and more appropriate corpora. Section 6 briefly describes our new acoustic SVM system and section 7 presents results of the fusion of our PPRLM and SVM systems. Finally, section 7 presents conclusions.

2. Baseline System

Our starting point for this paper is the two PPRLM systems we submitted to NIST LRE 2005. These systems used 6 (ATVS2) or 12 (ATVS1) phonetic decoders trained on the OGI Multi-Language Telephone Speech Corpus [8] which contains roughly 1-2 hours of speech by language. These decoders are based on Hidden Markov Models (HMMs) and implemented using HTK [9]. The phonetic HMMs are three-state left-to-right models with no skips, being the output *pdf* of each state modeled as a weighted mixture of Gaussians. In ATVS2 we used 10 Gaussians per state, while in ATVS1 we used 10 and 20 Gaussians per state to have two phonetic decoders with different complexities for each language. The acoustic processing uses the Advanced Distributed Speech Recognition Standard Front-End [10], based on 12 Mel Frequency Cepstral Coefficients (MFCCs) plus a combination of energy and C0 and velocities and accelerations for a total of 39 components, computing a feature vector each 10ms. It also includes mechanisms for robustness against channel distortion (blind equalization) and additive noise (double Wiener filter).

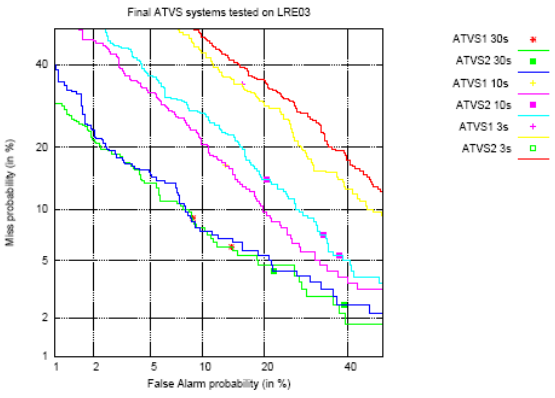


Figure 1: *Baseline system: results on a subset of NIST LRE'03 data using only the 7 languages considered in NIST LRE'05.*

The n-grams used as models for the different languages were trigrams without cut-off factor adapted from a UBM using data from one of the CallFriend (devset) database languages. The UBM n-gram was trained using transcriptions of speech segments (CallFriend devset) from the 12 CallFriend database languages. The adaptation coefficient was determined empirically and set to 0.6 for the UBM a 0.4 for the model only from the language.

Results from these two systems are shown on figure 1 (for a subset of NIST LRE 2003 data containing only test segments of the 7 languages considered in NIST LRE 2005). For NIST LRE 2003 data we attained a 9.14% EER for the 30s condition with the ATVS1 system and virtually the same with the ATVS2 system.

3. Robust vs. standard parameterization

Our baseline system used a robust front-end standardized by ETSI [10]. This front-end includes channel and noise effects compensation and has proved to produce better speech recognition results in noisy conditions. However, this front-end was less efficient than standard front-ends and was difficult to integrate with our systems. For that reason, we compared in a LR task the ETSI front-end to other simpler and more efficient. Our new front-end uses 12 MFCCs plus C0 and their velocities and accelerations for a total of 39 components, computing a feature vector each 10ms and performing Cepstral Mean Normalization (CMN).

Figures 2 shows results on NIST LRE 2003 data of 3 systems identical to the baseline systems, but with the new parameterization. The first one uses 6 phonetic decoders with 10 Gaussians/state, the second 6 with 20 Gaussians/state, and the last one all the 12 phonetic decoders. By comparing figures 1 and 2 we can conclude that the use of a robust front-end has very little influence in language recognition performance – with both front-ends results are virtually the same. By comparing the different systems in Figure 3 we can also conclude that the difference in performance achieved by using the 12 phonetic decoders (at least for the 30sec condition) does not justify the increase in computational cost required.

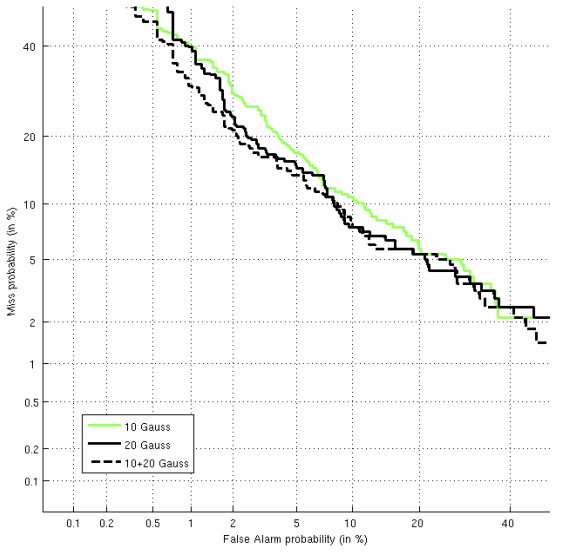


Figure 2: *Baseline system with new front-end: results on a subset of NIST LRE'03 data using only the 7 languages considered in NIST LRE'05.*

4. Adding Voice Activity Detection

One of the main differences between the NIST LR evaluations in 2003 and 2005 is that in 2003 a Voice Activity Detector (VAD) was used by NIST to remove silence areas from the recordings, while in the 2005 evaluation silence segments were kept in the recording to make conditions more realistic. Our baseline system did not include an explicit VAD. It tried to remove the effect of silence segments by removing repetitions of the silence label before training the n-grams and computing the scores. We suspected that the lack of a prior VAD to remove silences could be one of the reasons for the difference in performance between NIST LRE 03 data and NIST LRE 05 data. In order to explore this issue we have included a VAD based on energy levels and temporal restrictions and have obtained results on NIST LRE 03 data and NIST LRE 05 data, using in both cases the new parameterization and only 6 phonetic decoders with 20 Gaussians per state.

The comparison of results obtained for the systems with and without VAD on NIST LRE 03 data (figure 3) and NIST LRE 05 data (figure 4) shows that results are almost the same with and without VAD. This means that the removal of repetitions of the silence label seems to be an adequate way of removing the influence of the silent segments. Computational efficiency, however, is higher with the inclusion of an external VAD that avoids further processing of silences.

5. Using better phonetic decoders

Quality of the phonetic decoders has been recently proposed as a crucial factor in PPRLM performance for language recognition [11]. However, the experiments in [11] were performed using a very special phonetic decoder using artificial neural networks. Here we will extend the work in [11] by checking whether the same conclusions stand for more conventional HMM-based phonetic decoders. Towards this end, we have substituted the phonetic decoders trained on OGI Multi-Language Telephone Speech Corpus, which contained around 1-2 hours of speech by language, by new phonetic decoders trained on SpeechDat-like corpora, all of which contain over 10 hours of training material covering hundreds of different speakers. In particular, we have trained 6 new phonetic decoders in English, German, French, Arabic, Basque and Russian using SpeechDat-like corpora. We have also included a 7th phonetic decoder in Spanish trained on Albayzin [12] downsampled to 8 kHz, which contains about 4 hours of speech for training, but we report results separately for the system with the 6 and 7 recognizers. All the phonetic decoders share the same HMM structure – identical to the baseline systems, with 20 Gaussians/state. Also, the front-end is the same used in former sections and the systems include the external VAD.

With the new phonetic decoders important improvements are obtained. For the NIST LRE 2003 data (figure 3) just by changing the 6 phonetic decoders trained on OGI by 6 phonetic decoders trained on SpeechDat-like corpora language recognition results improve very significantly moving from 10.04% EER to 6.45% EER. Adding the Spanish recognizer the EER reduces to only 5.08%. This improvement is even more noticeable on NIST LRE 2005 data (figure 4). Here we move from a 16.38% EER to an 8.37% EER – a relative reduction of almost 50%. Adding the phonetic decoder for Spanish we get a 7.94% EER. These results stress the importance of having good quality phonetic decoders for language recognition based on PPRLM.

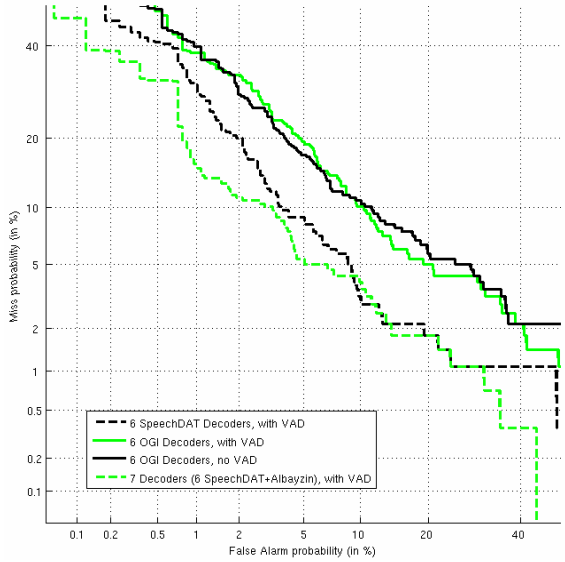


Figure 3: *The effect of VAD and better phonetic models: Comparison of results using models trained with OGI (with and without VAD) and models trained on SpeechDAT-like corpora on a NIST LRE'03 subset using only the 7 languages of NIST LRE'05.*

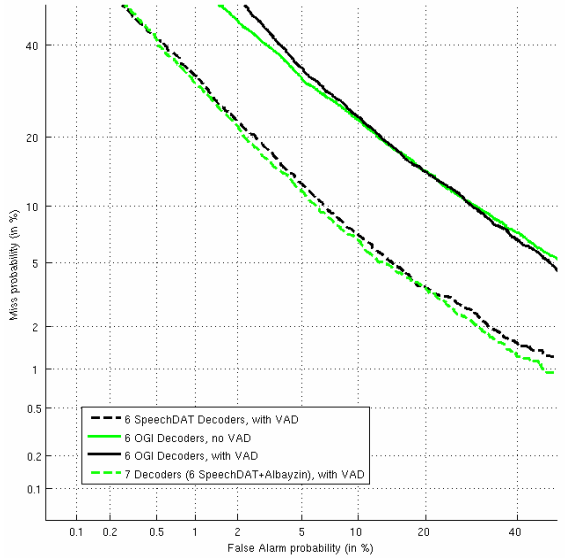


Figure 4: *The effect of VAD and better phonetic models: Comparison of results using models trained with OGI (with and without VAD) and models trained on SpeechDAT-like corpora on NIST LRE'05 data.*

6. SVM Sytems with MFCC and SDC-MFCC features

Besides PPRLM systems, which tend to be the best performing individual systems for LR [5], other systems very used for LR are acoustic systems that model the acoustic features for each particular language, typically using Shifted Delta Cepstra (SDC) features.

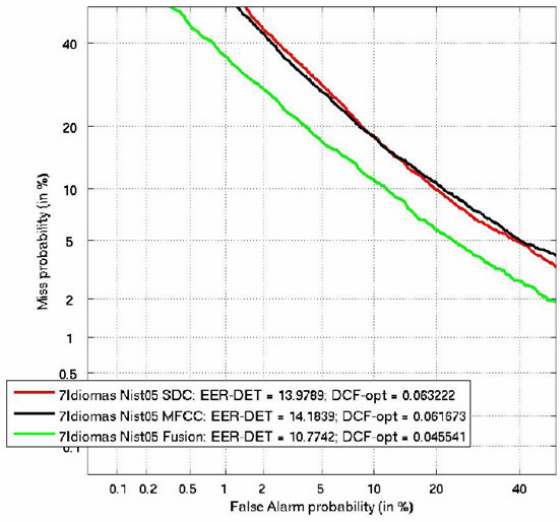


Figure 4: *Acoustic SVM systems using MFCC and SDC features, and the fusion of both. Results on NIST LRE'05 data.*

We have developed an acoustic system based on Support Vector Machines (SVM) [13]. Actually the system is the sum fusion of two SVM systems, one using 19 MFCC coefficients plus deltas and the other using SDC-MFCCs (7-2-3-7) [14]. In order to avoid channel mismatch effects, Cepstral Mean Normalization is applied, followed by RASTA filtering and feature mapping [15]. Both systems use a kernel expansion on the whole observation sequence, and a separating hyperplane is computed between the target language features and the background model. ATVS acoustic SVM-GLDS system uses a polynomial expansion of degree three [16] followed by a Generalized Linear Discriminant Sequence kernel (GLDS) as described in [17]. Finally, Tnorm score normalization technique is performed in order to scale the scores distribution.

The system has been trained using data from CallFriend, NIST LRE 1996, NIST LRE 2003 and has been evaluated on NIST LRE 2005 data (figure 4). The SVM system using MFCC features achieved a 14% EER and the SVM system using SDC-MFCC features achieved a 13.2% EER on NIST LRE 2005 data. When these two SVM systems were fused together with sum fusion we achieved an EER (figure 5) of only 10.5%.

7. Fusion with acoustic systems

Systems submitted to NIST LR Evaluations are rarely based on a single methodology. Rather they are usually the fusion of several systems using different approaches to the problem of LR. Even if the other systems are worse in terms of LR performance than the PPRLM system, the fusion of different systems tend to improve overall LR performance.

We have fused the results of our improved PPRLM system and our new SVM acoustic system with a simple sum fusion followed by Tnorm. This fusion has produced the best result we have achieved so far on NIST LRE 2005 data (figure 5), a 5.43% EER, which implies a relative reduction of almost 66% from our baseline system.

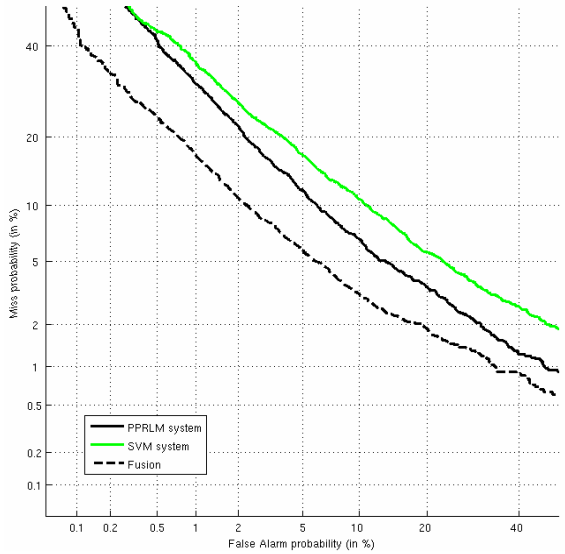


Figure 5: *Fusion of PPRLM system and acoustic SVM system on NIST LRE'05 data.*

8. Conclusions

In this paper we have improved our baseline PPRLM system achieving an EER reduction of almost 50% (from 16.38 to 8.37%). This improvement was mainly achieved by changing the phonetic decoders by other better trained (on more and more adequate data). We have also improved our PPRLM system by adding an explicit Voice Activity Detector (VAD) and a simpler front-end. While the influence of these changes on LR performance is very limited, they improve substantially the computational efficiency of the PPRLM system.

We have also developed a new acoustic system based on the fusion of two SVM systems, on using standard MFCC features and other using SDC features. Each of these systems achieves a LR performance of 13-14% EER by itself, but the fusion of both achieves an EER of only 10.5%.

By fusing our improved PPRLM system with our new acoustic SVM system we obtain a remarkable 5.43% EER on NIST LRE 05 data, which represents an EER relative reduction of around 66% from our baseline system.

9. Acknowledgements

This work was funded by the Spanish Ministry of Science and Technology under project TEC2006-13170-C02-01.

10. References

[1] "National institute of standard and technology. Language Recognition Evaluation Main Page," http://www.nist.gov/speech/tests/lang/index.htm.

[2] M.A. Zissman, "Comparison of four approaches to automatic language identification of telephone speech.," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 4, no. 1, pp. 31–44, 1996.

[3] Gleason T.P. Campbell W.M. Reynolds D.A. Singer E., Torres-Carrasquillo P.A., "Acoustic, phonetic, and discriminative approaches to automatic language identification," in *Proc. Eurospeech 2003*, Sept. 2003, pp. 1345–1348.

[4] J.L. Gauvain, A. Messaoudi, and H. Schwenk, "Language Recognition using Phone Lattices", in Proc. ICSLP 2004.

[5] Shen, W., Campbell, W., Gleason, T., Reynolds, D., Singer, E., "Experiments with Lattice-based PPRLM Language Identification", in Proc. IEEE Odyssey 2006, Puerto Rico, June 2006.

[6] A. 0. Hatch, B. Peskin, & A. Stolcke, "Improved Phonetic Speaker Recognition Using Lattice Decoding", In Proc. ICASSP 2005, Vol. 1, pp. 169-172.

[7] A. Montero-Asenjo, D. T. Toledano, J. González-Domínguez, J. González-Rodríguez, J. Ortega-García, "Exploring PPRLM performance for NIST 2005 Language Recognition Evaluation", in Proc. IEEE Odyssey 2006, Puerto Rico, June 2006.

[8] "OGI multi language telephone speech," http://www.cslu.ogi.edu/corpora/mlts/.

[9] Hidden Markov Model ToolKit (HTK), available on http://htk.eng.cam.ac.uk/ .

[10] ETSI ES 202 050 (v1.1.3): "Speech processing, transmisión and quality aspects (STQ); Distributed speech recognition; Advanced front-end features extraction algorithm; Compression algorithms."

[11] Matejka Pavel, Schwarz Petr, Cernocký Jan, Chytil Pavel, "Phonotactic Language Identification using High Quality Phoneme Recognition", In: Interspeech'2005 - Eurospeech - 9th European Conference on Speech Communication and Technology, Lisbon, PT, 2005, p. 2237-2240.

[12] A. Moreno, D. Poch, A. Bonafonte, E. Lleida, J. Llisterri, J. Mariño, C. Nadeu, "ALBAYZÍN Speech Database: Design of the Phonetic Corpus," in *proceedings of the 3rd European Conference on Speech Communication and Technology (EUROSPEECH)*. Berlin, Germany, 21-23 September 1993. Vol. 1. pp. 175-178.

[13] W. M. Campbell, J. P. Campbell, D. A. Reynolds, E. Singer, and P. A. Torres-Carrasquillo, "Support vector machines for speaker and language recognition," Computer Speech and Language, vol. 20, no. 2-3, pp. 210–229, 2006.

[14] Pedro A. Torres-Carrasquillo, Elliot Singer, Mary A. Kohler, Richard J. Greene, Douglas A. Reynolds and J.R. Deller Jr, "Approaches to Language Identification using Gaussian Mixture. Models and Shifted Delta Cepstrum", ICSLP, 2002.

[15] Gonzalez-Rodriguez, J., Ramos-Castro, D., Torre-Toledano, D., Montero-Asenjo, D., Gonzalez-Dominguez, J., Lopez-Moreno, I., Fierrez-Aguilar, J., Garcia-Romero, D. and Ortega-Garcia, J., "On the Use of High-level Information for Speaker Recognition: the ATVS-UAM system at NIST SRE 2005", to appear in IEEE Aerospace and Electronic Systems Magazine, 2007.

[16] W. Wan and W. Campbell, "Support vector machines for speaker verification and identification," in Proc. of IEEE International Workshop on Neural Networks for Signal Processing, 2000, pp. 775–784.

[17] W. Campbell, "Generalized linear discriminant sequence kernels for speaker recognition," in Proc. of ICASSP, 2002, pp. 161–164.