UNIVERSIDAD DE GRANADA

RECONOCIMIENTO DE VOZ MEDIANTE MODELOS OCULTOS DE MARKOV: SELECCIÓN Y ESTIMACIÓN DE PARÁMETROS

Antonio M. Peinado Herreros
José C. Segura Luna
Antonio J. Rubio Ayuso

Monografías del Dpto. de Electrónica Nº 31

Facultad de Ciencias
Departamento de Electrónica
y Tecnología de Computadores

RECONOCIMIENTO DE VOZ MEDIANTE MODELOS OCULTOS DE MARKOV: SELECCIÓN Y ESTIMACIÓN DE PARÁMETROS

Antonio M. Peinado Herreros
José C. Segura Luna
Antonio J. Rubio Ayuso

Monografías del Dpto. de Electrónica Nº 31

Granada, Marzo 1994

Editado por el Departamento de Electrónica
y Tecnología de Computadores

Depósito legal GR-531-1994
ISBN 84-7951-008-0

A mis padres,
María Dolores y José María,

... y a Victoria.

"Y en cuanto a las experiencias que hayan hecho ya los demás, aún cuando se le quieran comunicar, (...), se componen en su mayor parte de tantas circunstancias o ingredientes superfluos que le sería muy dificil descifrar lo que haya en ellas de verdad."

DESCARTES

Índice

Índice de Tablas

Índice de Figuras

Capítulo 1

INTRODUCCIÓN

Este capítulo pretende situar el presente trabajo en el contexto del Reconocimiento Automático del Habla. Los primeros apartados están dedicados a la definición, historia y planteamiento general del problema de reconocimiento. Posteriormente se realizará un breve resumen de las principales técnicas actualmente vigentes: Comparación de Objetos, Modelos Ocultos de Markov y Redes Neuronales. Después de hacer un rápida referencia a las aplicaciones y perspectivas del reconocimiento de voz, se concluirá el capítulo con la presentación y justificación del trabajo, enmarcándolo en el contexto desarrollado en los apartados anteriores.

1.1 Introducción General

El hombre se ha sentido siempre fascinado por su capacidad de hablar. Sin esta habilidad para comunicar fluidamente ideas, la sociedad actual probablemente no habría podido desarrollarse. De aquí nace la obsesión por crear instrumentos capaces de producir y reconocer voz, imitándonos a nosotros mismos. Un sistema de Reconocimiento de Voz puede ser definido como cualquier mecanismo, distinto del sistema auditivo humano, capaz de decodificar la señal acústica producida por el aparato fonador de un locutor en una secuencia de unidades lingüísticas que contiene el mensaje que ese locutor desea comunicar. El Reconocimiento Automático del Habla (RAH, en adelante) se sitúa en el marco más general del Procesamiento de la Voz Humana, que incluye, aparte del RAH, problemas tan diversos como Síntesis de Voz, Codificación de Voz y Reconocimiento de Locutores. Aunque los objetivos de cada una de estas ramas son completamente distintos, comparten conceptos, formulaciones y técnicas, lo cual da cuerpo al Procesamiento de Voz como disciplina autónoma englobada en el Procesamiento de Señales.

El objetivo básico del RAH es la comunicación hombre-máquina. Esta idea abarca un amplio espectro de aplicaciones: acceso a sistemas de información automáticos, ayuda a minusválidos, traducción automática, transacciones bancarias automáticas, control oral de sistemas, etc.

1.1.1 Evolución Histórica

La historia del Procesamiento de Voz comienza con los primeros dispositivos capaces de sintetizar voz humana. La importancia histórica de estos dispositivos queda patente si se tiene en cuenta la incidencia que la comprensión de los mecanismos de producción de la voz humana ha tenido sobre el conjunto de técnicas de Procesamiento de Voz, y sobre el RAH, en concreto. El primer sintetizador de voz fue construido por Kratzenstein en 1779, y era capaz de reproducir sonidos vocálicos mediante resonadores. Una máquina más sofisticada capaz de producir también consonantes fue diseñada en 1791 por Von Kempelen. El primer sintetizador eléctrico fue el Voder (Voice Operation Demonstrator) diseñado por Dudley, Riesz y Watkins (Bell Labs) en 1939, basado en un banco de filtros pasa banda que imitaban el tracto vocal humano. Este dispositivo ya distinguía dos tipos de excitaciones del tracto: sonora y no sonora, distinción que ha tenido una especial importancia en el desarrollo del Procesamiento de Voz.

En cambio, los avances en el campo del Análisis de voz son bastante más

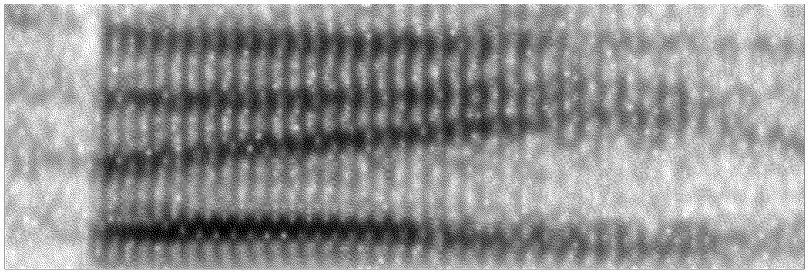


Figura 1.1: *Espectrograma de la palabra **seis**.*

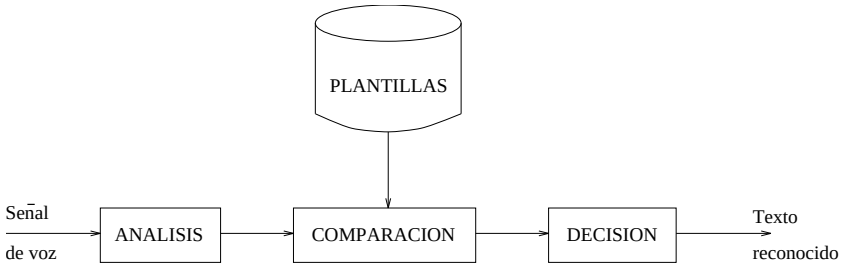


Figura 1.2: *Esquema básico de un sistema de reconocimiento.*

recientes. El espectrógrafo no fue desarrollado hasta la década de los años 40. Este dispositivo es capaz de construir espectrogramas, en los que se pueden observar las características espectrales de una señal de voz (ver Figura 1.1). Sólo a partir de entonces es cuando comenzaron los primeros trabajos referentes a reconocimiento automático de voz. Uno de éstos reconocedores fue el desarrollado por Davis, Bidulph y Balashek en los Laboratorios Bell en 1952. Era capaz de discernir los diez dígitos ingleses con una tasa de reconocimiento de hasta el 98% para un sólo locutor, mediante la comparación de la trayectoria de los dos primeros formantes con una plantilla o prototipo (obtenido mediante análisis de una secuencia de voz) previamente almacenado. Aunque muy rudimentario, este sistema ya establecía las pautas generales de lo que es un sistema de reconocimiento (ver figura 1.2). En 1958, Fry y Denes desarrollan un dispositivo capaz de reconocer fonemas (4 vocales y 9 consonantes) [Ainsworth76]. El rendimiento del sistema no era excesivamente bueno, pero demostró que incorporando cierta información lingüística podía incrementar su precisión del 24 al 44%.

La utilización de computadores para RAH no tardó en llegar, debido a la habilidad de éstos para realizar procesamientos más complejos y tareas de almacenamiento, así como por la sencillez y rapidez en la modificación de los sistemas. Un primer esfuerzo en esta dirección fue el sistema desarrollado por Denes y Mathews en 1960 [Flanagan72] para el reconocimiento de los dígitos ingleses. Este sistema se basaba en la comparación con una serie de prototipos entrenados a partir de secuencias pronunciadas por 5 locutores distintos, y alcanzaba hasta un 6% de error cuando se reconocían secuencias diferentes emitidas por esos mismos locutores realizando una normalización temporal de las mismas. Durante los años 60 hay una gran proliferación de trabajos en este sentido. Cabe destacar el trabajo desarrollado por Reddy [Reddy75] en el campo del reconocimennto de voz continua mediante el seguimiento de fonemas. Reddy inició así un fructífero programa de investigación en Carnegie Mellon University, universidad que sigue siendo líder en el campo de voz continua.

En 1971 EE.UU. aborda el proyecto ARPA-SUR (Advanced Research Projects Agency - Speech Understanding Research) con una fuerte dotación económica, y con el propósito de desarrollar sistemas de reconocimiento de frases en forma continua [Mariani89]. Aunque algunos sistemas desarrollados bajo este proyecto lograron los objetivos propuestos, su coste computacional era excesivamente alto para el resultado obtenido. De hecho, una de las principales conclusiones del proyecto fue la necesidad del desarrollo de una investigación metódica y básica para resolver eficientemente los problemas inherentes al RAH. Frutos importantes de la investigación desarrollada en los años 70 son la aplicación de ideas de Reconocimiento de Formas al RAH, el desarrollo del análisis de voz mediante tácnicas de Predicción Lineal, y la aplicación con éxito de los métodos de Programación Dinámica [Rabiner93].

Después de 20 años de investigación en esta dirección, se han podido estudiar y proponer nuevas y potentes técnicas que, junto con el desarrollo espectacular de los ordenadores, han permitido la aparición de algunas aplicaciones comerciales y prototipos de reconocimiento de voz continua [Roe93]. Aún así, de momento, estos sistemas sólo pueden trabajar con cierto grado de fiabilidad si se les imponen algunas restricciones. Sobre estas técnicas y restricciones se hablará en los próximos apartados. Estos últimos años han dejado claro que el RAH tiene un carácter multidisciplinario, nutriéndose de materias tan diversas como Acústica, Fonética, Procesamiento Digital del Señales, Reconocimiento de Formas, Inteligencia Artificial, etc., sobre las que será necesario apoyarse para conseguir nuevos

avances y mejoras. Material tutorial sobre todas estas materias y sobre distintos aspectos tratados en este apartado, pueden encontrarse en [Ainsworth76, Flanagan72, Casacuberta87, Reddy75, Rabiner78, Fukunaga90, Quilis89, Rabiner93].

1.2 Planteamiento General del Problema

El problema del RAH puede ser planteado en los siguientes términos: *¿Cómo extraer la información contenida en una señal de voz recibida a través de un micrófono de forma que el hombre pueda comunicarse con una máquina mediante su voz?*. Casacuberta y Vidal dan la siguiente respuesta [Casacuberta87]: *Hacer cooperar un conjunto de informaciones plagadas de ambigüedades, incertidumbres y errores inevitables, para llegar a una interpretación aceptable del mensaje acústico recibido.*

Esas informaciones *cooperativas* podrían clasificarse en los siguientes niveles:

1) Nivel Acústico: se extraen una serie de características fundamentales de la señal acústica, eliminando redundancias existentes en dicha señal.

2) Nivel Fonético: se determinan las unidades sonoras básicas (palabras, fonemas, sílabas, ...).

3) Nivel Sintáctico: se aplican reglas gramaticales al conjunto de unidades a reconocer.

4) Nivel Semántico: se pretende realizar una comprensión del mensaje y eliminar interpretaciones sin sentido.

Los distintos niveles pueden estar absolutamente interrelacionados entre sí, de forma que no sea posible atacar un nivel de forma independiente. Además, según el tipo de restricciones que se apliquen al sistema, no siempre serán necesarios todos los niveles en la implementación de los sistemas.

El problema de reconocimiento planteado se ve complicado en la vida real por una serie de problemas adicionales tales como la *variabilidad*, es decir, una misma frase nunca es pronunciada de forma idéntica, ya sea por variaciones intralocutor (variación de las condiciones físicas y psíquicas del locutor) o bien por variaciones interlocutor (distintos acentos, dialectos, sexo, edad, ...), o bien por variaciones en el entorno del locutor (micrófono, amplificación de la señal, conversión A/D, ...). También es frecuente que los reconocedores se vean obligados a trabajar en

presencia de *ruido*, lo cual complica el reconocimiento en dos direcciones. La primera es la propia alteración que supone el ruido al superponerse a la señal de voz. En segundo lugar, el locutor modificará su forma de hablar cuando hay ruido ambiente, lo que es conocido como *efecto Lombard*.

1.3 Categorías de Reconocimiento

Según las *restricciones* que se impongan en la construcción de un sistema de reconocimiento, se pueden establecer diferentes categorías. En primer lugar si se atiende a la forma en que se usan los locutores en las fases de entrenamiento y test del sistema, se tendrán las siguientes categorías de sistemas:

Monolocutor El sistema es entrenado con un solo locutor, y las pruebas de test de fiabilidad del sistema son realizadas con el mismo locutor.

Multilocutor El sistema es entrenado con varios locutores, y los mismos son los que realizan el test del sistema.

Independiente del Locutor Los conjuntos de locutores de entrenamiento y test son distintos.

Obviamente, la variabilidad implicada por estas categorías es creciente, y, por tanto, la cantidad de datos necesaria para su entrenamiento debe ser también creciente con el fin de recoger el mayor espectro posible de variaciones.

Otra decisión importante que hay que tomar a la hora de diseñar un sistema es el tipo de *unidad de reconocimiento* que se vaya a usar, de forma que las frases reconocidas se obtienen a partir de concatenaciones posibles de dichas unidades. La unidad más natural para el hombre, por tener un significado completo, es la *palabra*. El problema de usar estas unidades es que el reconocimiento puede volverse extraordinariamente complejo cuando se usen vocabularios medianamente grandes (más de 100 palabras). Aunque existen sistemas con vocabularios amplios que usan la palabra como unidad (Tangora System [Group85]), lo usual es utilizar *unidades inferiores a la palabra* (Sphinx System [Lee89]). La unidad inferior a la palabra básica es el *fonema* [Quilis89, Sugamura83]. El problema del uso de fonemas en el RAH es que un mismo fonema puede presentar variaciones importantes según el contexto en que aparezca, esto es, según los fonemas que le antecedan y que le sucedan, lo que es conocido como *efecto coarticulatorio*. Otras

unidades inferiores a la palabra como *difonemas*, *sílabas*, *demisílabas* [Lee90c] o *trifonemas* han sido propuestas para solventar este efecto [Lee90c, Lee90].

El tipo de secuencias de voz que se pretenda reconocer da lugar a una segunda clasificación de los sistemas:

Palabras Aisladas Se supone que hay silencios entre palabras, de forma que es posible aislar estas palabras y reconocerlas una a una, usando las propias palabras como unidades.

Palabras Conectadas Las palabras pueden ser emitidas sin silencios intermedios, pero la unidad es nuevamente la palabra. El problema que plantea el reconocimiento de palabras conectadas es la decodificación de la secuencia óptima de palabras correspondiente a la secuencia emitida. Es bastante usual el uso de palabras conectadas por los sistemas de reconocimiento de dígitos conectados [Rabiner89].

Voz Contínua En este caso se aborda el reconocimiento de frases completas sin pausas entre palabras (engloba al tipo anterior). Aunque en principio no existe ninguna restricción respecto al tipo de unidad a usar, se suele imponer el uso de unidades inferiores a la palabra. Nuevamente aparece el problema de la decodificación óptima de la frase. Este no es un problema local (no se reconoce unidad a unidad), sino que la frase reconocida debe responder a un conjunto de reglas que forman una *gramática*. El conjunto de frases generadas por la gramática se llama *lenguaje*, al cual pertenecerá la frase reconocida, y debe pertenecer la frase emitida para un correcto reconocimiento. Detalles sobre este tipo de sistemas pueden ser encontrados en [Lee89, Lee90a].

Word-Spotting Se pretende el reconocimiento de palabras pertenecientes a un vocabulario determinado inmersas en frases en las que pueden aparecer palabras ajenas al mismo. Se usan técnicas similares a las propuestas en palabras conectadas, pero permitiendo el rechazo de las palabras extrañas [Wilpon90].

1.4 Aproximaciones al problema del RAH

En este apartado se realizará un rápida revisión de los métodos más utilizados para la implementación de sistemas de reconocimiento. Para ello se seguirá un

orden aproximadamente histórico, agrupando las distintas técnicas en métodos de *Comparación de Objetos*, *Modelado Oculto de Markov* y *Redes Neuronales*. Casi todos los sistemas de reconocimiento basan su actuación en la partición de la señal u *objeto* en segmentos cortos (20-40 ms) denominados *tramas*, solapados o no entre sí. Esta segmentación está basada en la suposición (no siempre correcta) de que la señal de voz presenta características cuasi-estacionarias en segmentos cortos, por lo que se puede considerar que en ese segmento se mantienen las características espectrales de la señal. De esta forma es posible utilizar la *distancia* entre los espectros de dos tramas como medida de similitud entre ambas. Sobre análisis de señales de voz y medidas de distorsión o distancias se discutirá más extensamente en capítulos posteriores.

1.4.1 Comparación de Objetos

La aproximación básica al problema del RAH (palabras aisladas) consta de dos fases: entrenamiento y test. En la etapa de entrenamiento se obtiene una serie de objetos de referencia o *plantillas* que representan a las distintas clases de señales como secuencias de vectores (cada cual conteniendo la información espectral de la trama correspondiente). En la fase de test, el objeto de entrada es comparado (previo análisis espectral) con todos los objetos de referencia usando una distancia de procesado de señal apropiada. La comparación que arroje una distancia total menor corresponde al objeto reconocido.

Un primer problema que presenta esta aproximación es la dificultad de comparar objetos (secuencias de tramas) de distintas longitudes. Efectivamente, incluso una misma palabra emitida por el mismo locutor presentará normalmente distintas duraciones. Por tanto, para realizar el reconocimiento de secuencias de longitud variable hay que *alinear* el objeto de entrada con el de referencia. Este alineamiento se puede conseguir con un método de *Programación Dinámica*. El método fue inicialmente introducido por Bellman [Bellman57] para el alineamiento óptimo de dos objetos, y aplicado por primera vez a reconocimiento de palabras por Vintsjuk [Vintsjuk68]. En la bibliografía se suele nombrar al método como DTW (Dynamic Time Warping) por la dilatación-compresión que se realiza del eje temporal. El método funciona como sigue: se construye la matriz de distancias entre los objetos de test (con I tramas) y de referencia (con J tramas) $d(i,j)$ $(i = 1, ..., I; j = 1, ..., J)$ (ver figura 1.3). El alineamiento de las dos secuencias consiste en la búsqueda del un "camino óptimo" (en trazo grueso en la

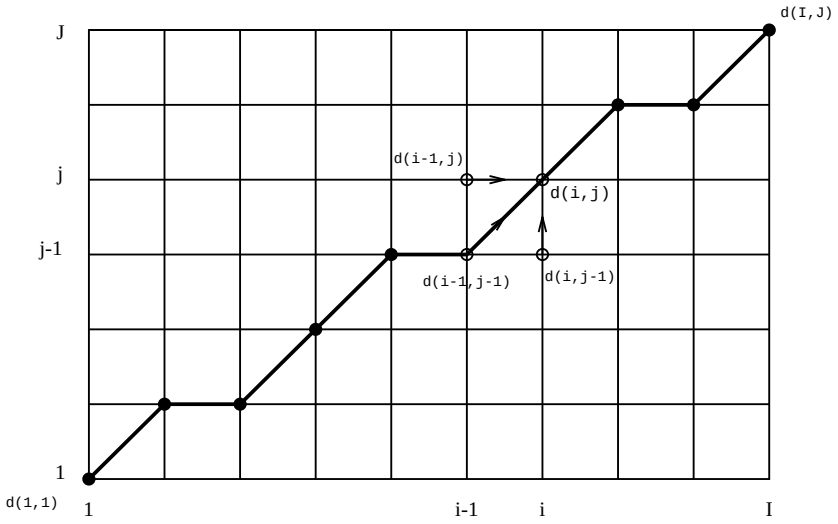


Figura 1.3: *Algoritmo DTW.*

figura), que se corresponderá con el que genere una distorsión total mínima. Si se define $g(i,j)$ como la distancia acumulada en el camino óptimo desde $d(1,1)$ hasta $d(i,j)$, que se puede obtener recursivamente mediante,

$$g(i,j) = min \begin{cases} g(i-1,j) + d(i,j) & \text{insertar trama} \\ g(i-1,j-1) + 2d(i,j) & \text{progresión normal} \\ g(i,j-1) + d(i,j) & \text{borrar trama} \end{cases} \tag{1.1}$$

entonces, la distancia total para ese camino es,

$$D = \frac{g(I,J)}{(I+J)} \tag{1.2}$$

Este proceso informa también sobre cuál es el mejor punto anterior a uno dado, de forma que se puede realizar una búsqueda hacia atrás, a partir del punto (I,J), para obtener el camino óptimo (alineación óptima). El objeto de entrada debe ser alineado con todas las plantillas de referencia. La comparación que proporcione una distancia o distorsión D mínima corresponderá al objeto reconocido.

La construcción de sistemas independientes del locutor mediante este método puede realizarse mediante el *agrupamiento* de las secuencias de entrenamiento

correspondientes a la misma palabra usando alguna técnica de agrupamiento (tal como el método k-medias [Tou74]). De esta forma se obtiene un conjunto de *nubes*, cada una representada por su centroide. Cada centroide será usado en la fase de reconocimiento como un objeto de referencia, y, por tanto, cada palabra queda representada por varias plantillas [Wilpon85].

La extensión del DTW a palabras conectadas presenta algunos problemas como la detección de las palabras contenidas en una frase emitida de forma continua. Además, las palabras son influenciadas por sus inmediatas adyacentes. Estos problemas han sido resueltos con éxito por algunos métodos como el "Two-Level DP Matching" propuesto por Sakoe [Sakoe79], el "Level-Building" propuesto por Myers et al [Myers81] o el "One-Pass DP" propuesto por Bridle [Rabiner93].

Otra posibilidad para reconocimiento por comparación de objetos es el uso de técnicas de Cuantización Vectorial (VQ). La técnica VQ fue inicialmente ideada para codificación de voz a bajo bit-rate (velocidad de transmisión) [Buzo80]. Mediante algún método de *clustering* [Duda73] se agrupan todos los vectores (representando tramas) de la secuencia de entrenamiento en varias clases o nubes de acuerdo con alguna medida de distorsión. Cada una de estas nubes queda representada por su centro (obtenido como promedio de los vectores integrantes de la nube). El conjunto de estos centros constituye un diccionario. A cada uno de estos centros se le asigna un índice entero. Cuando llega al codificador un vector de entrada, éste es reemplazado por el índice del centro vecino más próximo. La sustitución del vector de parámetros por un sólo indice conlleva una sustanciosa reducción del número de bits necesario para representar al vector de entrada. Esta técnica ha sido utilizada en la construcción de algunos sistemas de reconocimiento [Burton85]. El entrenamiento consiste en crear un diccionario por clase de objeto a reconocer. Durante la fase de test, la secuencia de test es evaluada en cada uno de los diccionarios. El diccionario en el que la distorsión total de la secuencia de test sea mínima corresponderá al objeto reconocido. Obviamente, el principal defecto del método es que no realiza ninguna clase de alineamiento temporal, desaprovechando esta información. A pesar de ésto, la técnica VQ ha sido profusamente aprovechada en la construcción de sistemas de reconocimiento como se verá a lo largo de este trabajo.

Con el objeto de reducir los requerimientos de memoria de los sistemas se han introducido unidades inferiores a la palabras, aplicando algoritmos DTW de "palabras" conectadas [Mariani89].

1.4.2 Modelado HMM

A diferencia de la comparación de objetos, en la que se representaba la referencia por un objeto, es posible representar la referencia mediante un modelo estocástico. Comúnmente se usan los *Modelos Ocultos de Markov* (*Hidden Markov Models*, HMM). Los primeros trabajos de aplicación de HMMs en RAH fueron desarrollados en CMU [Baker75] con el desarrollo del sistema DRAGON, y en IBM [Jelinek76].

Si consideramos una señal X, el procedimiento de reconocimiento mediante HMMs consiste en encontrar el texto escrito W (frase) que hace la probabilidad $P(W|X)$ máxima. Haciendo uso de la regla de Bayes es posible escribir,

$$P(W|X) = \frac{P(X|W)P(W)}{P(X)} \tag{1.3}$$

donde $P(W)$ es la probabilidad de la frase, $P(X|W)$ es la probabilidad de la señal X dado W y $P(X)$ es la probabilidad *a priori* de la señal, que, al ser constante, puede ser obviada en el proceso de reconocimiento. $P(W|X)$ puede ser obtenida de un *modelo acústico* y $P(W)$ de un *modelo de lenguaje*. Aunque ambos modelos pueden aproximarse mediante procesos de Markov, éstos son especialmente usados para modelado acústico. En el siguiente subapartado se abordarán muy superficialmente algunos aspectos generales del modelado oculto de Markov, base del modelado acústico, cuyos mecanismos serán extensamente abordados en los próximos temas.

Modelado Acústico

La suposición básica es que el que proceso de producción de voz responde a un *Proceso de Markov*. Por tanto, $P(X|W)$ corresponde a la probabilidad de generación de la señal X por el modelo correspondiente a W. Un modelo de Markov (ver figura 1.4) es un autómata de estados finitos que se compone de *estados* y *transiciones* entre los estados, siendo a_{ij} la *probabilidad de transición* del estado s_i al s_j. En un modelo de Markov *oculto* y *discreto*, existe además la posibilidad de que cada estado genere, cada vez que es visitado, un símbolo v_k perteneciente a un vocabulario finito. Cada símbolo v_k representa a una trama de señal y puede ser obtenido mediante un previo proceso de VQ sobre la señal. El símbolo v_k es generado por el el estado s_i con una *probabilidad de producción* $b_i(v_k)$. Las probabilidades de transición y producción constituyen los parámetros

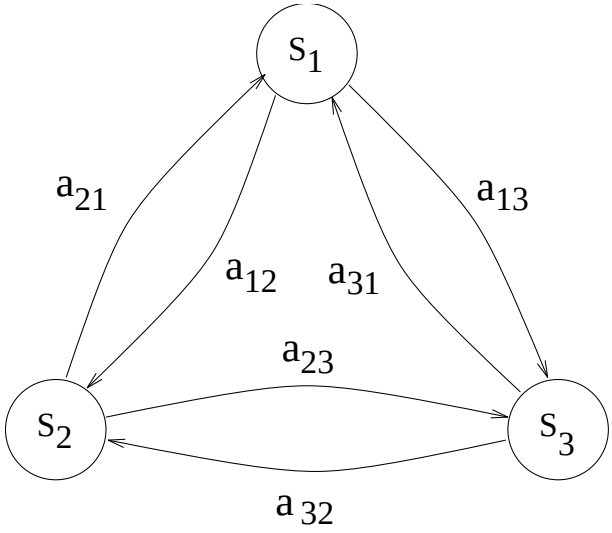


Figura 1.4: *Gráfico de un HMM.*

básicos de un HMM. Dados los parámetros del modelo es posible evaluar la probabilidad de generación $P(X|W)$. Si en lugar de usar símbolos de un alfabeto discreto para representar las tramas de señal usamos directamente los vectores de características el HMM se denomina continuo. En este caso, hay que sustituir las probabilidades de producción $b_i(v_k)$ por densidades continuas de producción $b_i(\mathbf{x})$. Existen otras formas de modelar este proceso de *producción* que dan lugar a otros tipos de HMMs, tales como modelos *semicontinuos* y modelos *MVQ*, que serán introducidos en un capítulo dedicado a Modelos de Markov, en el que además se abordarán los pricipales problemas derivados del uso de HMMs.

Al igual que sucedía con DTW, la "primera idea" es asociar cada HMM a una palabra del vocabulario. Una topología adecuada a la naturaleza secuencial de la voz es el modelo de Bakis o modelo de *izquierda-a-derecha* [Jelinek76, Rabiner86] (ver figura 1.5).

Pero al igual que en la aproximación DTW, el uso de unidades inferiores a la palabra, especialmente en aplicaciones de voz continua, puede representar una considerable reducción de la complejidad del sistema. Es frecuente que los HMM usados como unidades inferiores a la palabra contengan 3 estados representando comienzo, parte estable y final. Un primer tipo de unidad son los *fonemas independientes del contexto*, que ya fueron usados por IBM para el reconocimiento de

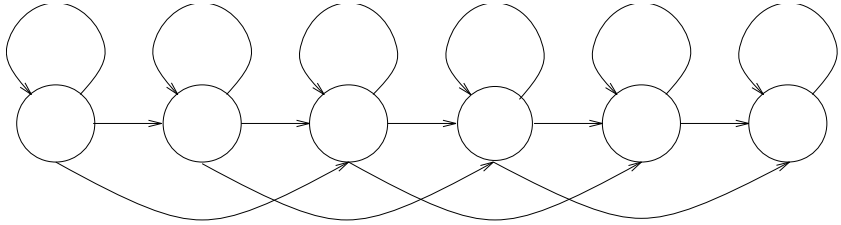


Figura 1.5: *Modelo izquierda-a-derecha.*

palabras aisladas y voz continua [Bahl83]. La principal ventaja de estas unidades es el bajo número de ellos, pero tienen el inconveniente de no considerar la variabilidad que pueden presentar los fonemas al presentarse en distintos contextos. Para solventar este problema aparecen los *fonemas dependientes del contexto* o *trifonemas*, de forma que para modelar cada fonema en un cierto contexto (fonema anterior y posterior) se utiliza un HMM [Schwartz85, Lee89]. Nuevamente surge el problema de tener un número alto de unidades, que puede ser reducido aprovechando que ciertos contextos ejerzan un efecto similar sobre un determinado fonema [Derouault87]. Otra solución son los *trifonemas generalizados* [Lee90], obtenidos mediante la agrupación de trifonemas (de un mismo fonema) similares según una medida de comparación entrópica basada en la pérdida de información cuando dos modelos son fusionados.

En resumen, tres aspectos deben ser tenidos en cuenta a la hora la búsqueda de una "buena" unidad:

Entrenabilidad La unidad ha de encontrarse con suficiente frecuencia en los datos usados para entrenar para que ésta quede bien entrenada. Este aspecto se ve más favorecido cuanto más pequeña sea la unidad.

Sensibilidad Hay que determinar si la unidad es muy dependiente del contexto. La sensibilidad aumenta conforme la unidad se hace más pequeña.

Compartibilidad Es posible enriquecer un modelo detallado a partir un modelo más genérico, por ejemplo, trifonemas con fonemas independientes del contexto. Para ello se puede usar el método de "Deleted Interpolation" [Lee89].

Modelado del Lenguaje

Para sistemas de voz continua se hace absolutamente necesario el uso de un modelo de lenguaje (comúnmente gramática) que colabore con el acústico para obtener tasas de error aceptables. La complejidad de la gramática es medida mediante la *perplejidad* que es, aproximadamente, el número medio de palabras que pueden suceder a una dada. El lenguaje puede ser modelado por una *gramática de estados finitos* que genere el mismo conjunto de frases que la tarea abordada [Bahl80]. Este modelado suele conducir a valores de perplejidad muy pequeños, que conducen a sistemas excesivamente restrictivos. Es común el uso de gramáticas menos restrictivas, de mayor perplejidad, que admiten la posibilidad de aparición de frases que no estaban presentes en el corpus de entrenamiento. Este es el caso de la *gramática de pares de palabras*, en la que sólo se especifica las listas de palabras que pueden seguir a una dada [Lee89].

También es posible modelar el lenguaje mediante un Proceso de Markov. Así se tienen gramáticas como la *bigram*, en la que la probabilidad de una palabra dada la previa es calculada como la frecuencia de aparición de esa pareja de palabras[Bahl83], o la *trigram* que es similar pero teniendo en cuenta las dos palabras precedentes. El problema de estas gramáticas es que a las sucesiones de palabras no presentes en el corpus de entrenamiento se les asigna probabilidad nula (especialmente si el corpus no es muy grande), lo cual conduce nuevamente a sistemas restrictivos. Esto puede solucionarse aplicando alguna probabilidad *umbral* o haciendo uso de la técnica "Deleted Interpolation" a una combinación de gramáticas (unigram, bigram, trigram) [Derouault86].

Comparación de las técnicas DTW y HMM

Algunos autores, como Kaltenmeier et al [Class86], sostienen que aunque los HMM requieren un entrenamiento más complicado que los sistemas basados en DTW, son más eficientes en reconocimiento (en cuanto a precisión y coste computacional). En cambio, Rabiner y Juang [Rabiner86] han comprobado que los modelos de Markov en versión discreta arrojan una tasa de error superior a los sistemas DTW, pero inferior en el caso de modelos continuos. En cualquier caso, gran cantidad de investigadores han adoptado la técnica HMM debido al potencial que supone disponer de un modelo de producción de voz.

1.4.3 Aproximación Conexionista

En esta aproximación las referencias estan representadas por patrones de actividad distribuidos a lo largo de una red de unidades simples de proceso, que por analogía con el sistema nervioso humano recibe el nombre de Red Neuronal Artificial (Artificial Neural Network, ANN).

Perceptrones Multicapa

El origen de las ANN está en el *Perceptrón*, propuesto por Rosenblatt [Rosenblatt59], que fue abandonado ante la imposibilidad de implementar ciertas funciones. Más recientemente, ha reaparecido el interés por este tipo de métodos, debido al interés por los *Perceptrones Multicapa* (Multilayer Perceptron, MLP) que, aparte de superar los problemas de implementación de su predecesor, posee mayor capacidad para clasificación que el Perceptron original [Lippman87]. También ha contribuido la proposición de un algoritmo de entrenamiento bien definido conocido como *Retropropagación* [Diaz91a]. Un MLP se compone de una capa de entrada, otra de salida, y una o varias capas ocultas (ver figura 1.6). Cada capa se compone de una matriz de unidades simples de proceso llamadas *neuronas*. Cada neurona es excitada, en el instante t, por una serie de entradas $a_i(t)$ con las que la se realiza una suma ponderada (con pesos w_i). Esta suma es procesada por una *función de activación* $f[\cdot]$ (normalmente tipo sigmoide) de forma que el valor a la salida de la neurona en el instante $t+1$ es,

$$s(t+1) = f\left[\sum_{i=1}^{N} w_i a_i(t) - T\right] \tag{1.4}$$

siendo T un cierto umbral prefijado. Si la suma ponderada es superior a T, la neurona es excitada, propagando las entradas a la capa superior.

En la fase de entrenamiento, los estímulos son propagados a través de la red hacia la capa de salida. En esta capa, la respuesta obtenida es comparada con la deseada. El error cometido se propaga hacia las capas inferiores (retropropagación) con el objetivo de reajustar los valores de pesos y umbrales. El proceso es iterado hasta alcanzar una situación estable. En reconocimiento, los estímulos son propagados por la red hasta la capa de salida. La neurona de esta capa que presente el valor de salida más alto corresponderá al objeto reconocido. En otros casos, la matriz de valores de salida es comparada con otra serie de matrices (que

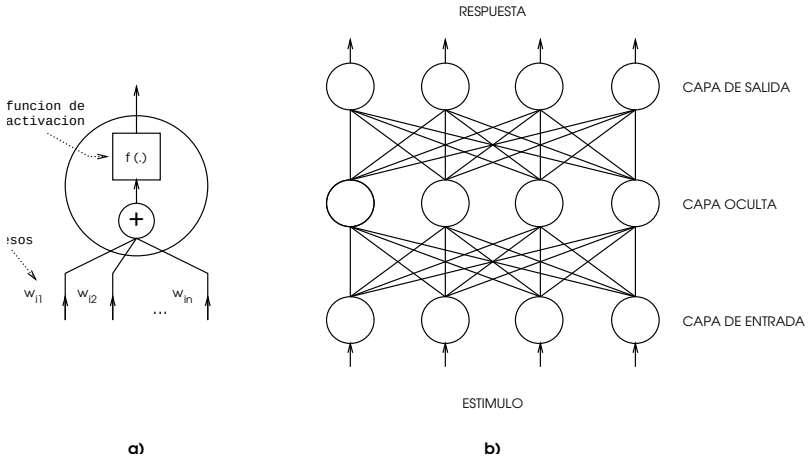


Figura 1.6: *Redes Neuronales: a) neurona básica y b) Perceptrón Multicapa.*

representan a los objetos de referencia) según una cierta distancia, para determinar la clase reconocida.

El MLP posee una estructura estática, es decir, un número de entradas fijo. Esto contrasta con la variabilidad temporal característica de las señales de voz. Varias soluciones han sido propuestas para resolver este problema. Una primera posibilidad es fijar el número de neuronas de entrada según la mayor longitud posible, rellenando el hueco remanente con silencios [Waibel91]. También es posible aplicar alguna técnica de compresión temporal (lineal o no lineal) [Casacuberta90]. Waibel et al [Waibel89] consiguieron obtener un MLP capaz de asimilar la naturalez dinámica de la señal de voz. Este es el *MLP de retardo temporal* (Time Delay MLP, TDMLP). En la figura 1.7 se muestra una red TDMLP para reconocer los fonemas /b/, /d/ y /g/ en diferentes contextos. La capa de entrada se compone de 16×3 nodos que aceptan 16 coeficientes de banco de filtros (en escala mel), y almacena las entradas de los instantes t, $t - \tau$ y $t - 2\tau$ (con $\tau = 10ms$). Las salidas de la capa de entrada alimentan 8 nodos de la primera capa oculta, y son almacenadas con retardos τ, 2τ, 3τ y 4τ. Las salidas de esta capa alimentan 3 nodos de la segunda capa oculta, que a su vez están conectados con los tres nodos de la capa de salida (cada nodo de esta capa representa a uno de los patrones a reconocer) que reciben energía de las capas inferiores durante toda la duración del estímulo. Esta red es invariante a desplazamientos temporales y no precisa

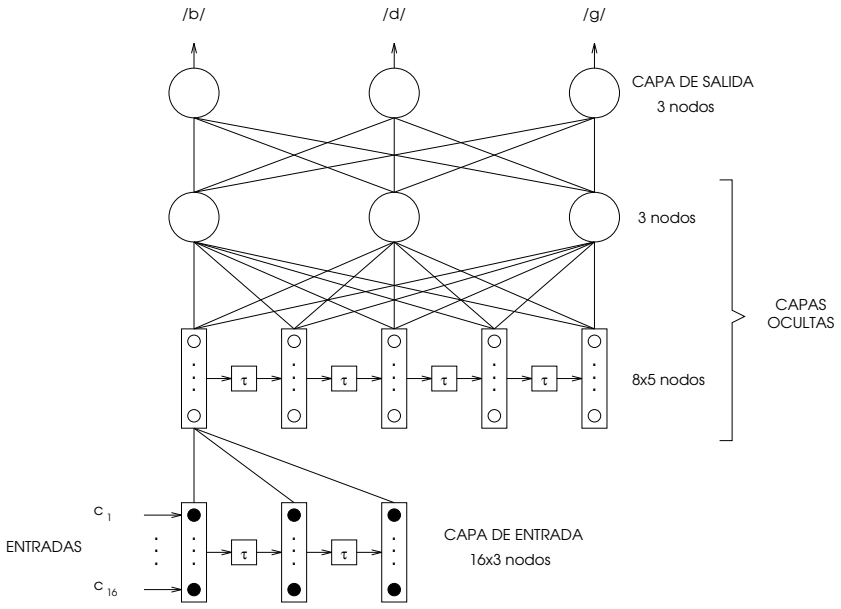


Figura 1.7: *Esquema de un MLP de Retardo Temporal.*

alineamiento o segmentación de la entrada. El porcentaje de acierto de la red TDMLP, en la tarea señalada, alcanza el 98.5%, mientras que un sistema basado en modelos HMM discretos sólo conseguía el 93.7%. Frente a esta mejora en la tasa de reconocimiento, esta red requiere una considerable cantidad de cómputo para su entrenamiento.

Máquinas de Boltzmann

Otra aproximación conexionista distinta de los MLP es la *Máquina de Boltzmann*, en la que, a diferencia de los MLP, las neuronas no se organizan en capas, sino que es posible la conexión de un nodo con otro cualquiera. En este caso, también se hace una distinción entre nodos "ocultos" y nodos "visibles" (de entrada y salida). La salida de cada neurona es binaria (0 ó 1) y depende de la suma ponderada de entradas (ahora llamada diferencia de energía). Para cada neurona se puede definir una función "temperatura". Cuanto más baja sea la temperatura de un nodo, más estará ese nodo influenciado por el estado de los nodos conectados a él y los arcos correspondientes. El proceso de entrenamiento es conocido como *simulated*

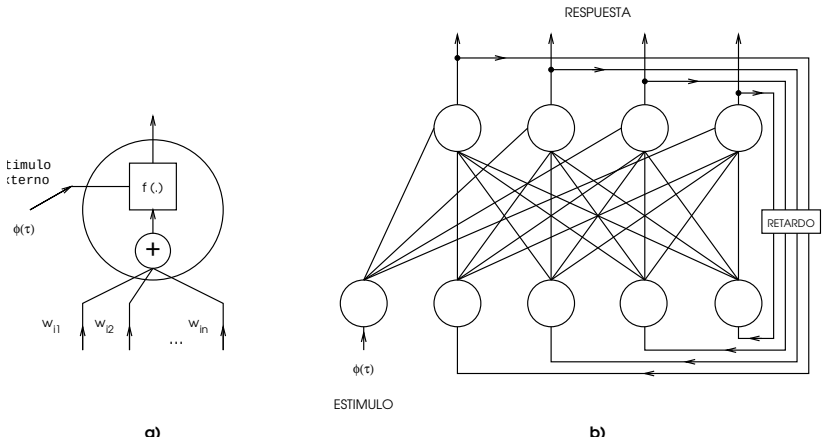


Figura 1.8: *Red Neuronal Recurrente: a) neurona básica y b) red completa.*

annealing [Kirkpatrick83], y consiste en disminuir poco a poco la temperatura de la red hasta alcanzar un mínimo global. Esto es una ventaja respecto a otros métodos que sólo conducen a un extremo local durante el entrenamiento. Aplicada a reconocimiento de voz, esta red ha demostrado ser ligeramente superior al MLP, pero con un coste computacional 10 veces superior [Prager87].

Redes Neuronales Recurrentes

Otra posibilidad son las Redes Neuronales Recurrentes (RNN), en las que el problema del alineamiento temporal es solucionado mediante 2 capas de neuronas (entrada y salida) conectadas recurrentemente [Bridle90] (ver figura 1.8). Las neuronas de la capa de salida, además de la excitación proviniente de la capa de entrada, disponen de una entrada para un estímulo externo. Las redes RNN se activan al recibir los estímulos externos y los propagan cíclicamente por la red, a través de las conexiones recurrentes, de forma que la salida correspondiente a una secuencia de duración T aparece en la capa de salida tras haber procesado los T estímulos, siendo su comportamiento equivalente al de un MLP con T capas ocultas. Las RNN presentan grandes analogías en su funcionamiento con los modelos HMM [Diaz91, Diaz93].

A pesar del atractivo que puedan mostrar las ANN, aún quedan serios problemas por establecer (qué arquitectura elegir, número de capas, número de neuronas, cómo

tratar la variabilidad temporal,...). Además, la experiencia no ha dejado clara su superioridad sobre las otras dos aproximaciones vistas en este capítulo.

1.5 Aplicaciones y Perpectivas del RAH

A pesar del gran avance y cantidad de trabajos realizados en los últimos años, no se ha alcanzado aún la gran meta de obtener un reconocedor capaz de transcribir voz natural. Sin embargo, no es necesario exigir a un sistema de RAH "capacidades humanas" para una amplia gama de aplicaciones. Basta con que posea algunas características básicas como sencillez de uso y minimización del impacto de los errores de reconocimiento. Entre las principales aplicaciones que existen actualmente pueden destacarse:

- Acceso oral y automático a servicios de información remotos, tales como el servicio de operadora, información telefónica, información bancaria, etc.
- Dictáfonos: sistemas capaces de transcribir voz humana.
- Comprensión del mensaje hablado: sistemas para el procesamiento de voz continua.
- Productos orientados al consumidor, como aparatos telefónicos, juegos o equipos stereo para coches.
- Sistemas para ayuda a minusválidos.

Las perspectivas para los próximos años [Rabiner93, Roe93] parecen coincidir en la obtención de sistemas de reconocimiento suficientemente precisos para tareas específicas con gran vocabulario. También ciertas aplicaciones como el acceso telefónico a sistemas de información y control oral como opción software serán previsiblemente de uso común. El esfuerzo investigador llevado actualmente a cabo en los campos de modelado acústico y, en especial, de lenguaje deben producir avances sustanciales, aunque la capacidad de comprensión del mensaje oral por computadores queda aún lejos de la capacidad humana para ello.

1.6 Justificación y Planificación del Trabajo

El presente trabajo está comprometido esencialmente con el estudio de algunos aspectos del modelado acústico con Modelos de Ocultos de Markov, que, por ahora,

sigue siendo la aproximación más extendida en el ámbito del RAH. El tipo de modelado de Markov más simple es el discreto. Su simplicidad radica en el hecho de que todo el espacio de representación es discretizado. Pero esta discretización conlleva una pérdida de la información contenida en los vectores de análisis, al ser cuantizados vectorialmente (usando un diccionario único o universal para todos los modelos). Por el contrario, el modelado continuo es el más completo y potente, ya que usa para cada estado una densidad continua de producción, como ya se indicó en este capítulo. Sin embargo, no siempre es usado en la implementación de sistemas. Esto se debe esencialmente a la gran cantidad de parámetros que hay que estimar para obtener unos modelos detallados. Esta estimación implica la necesidad de una gran cantidad de cálculo y de una base de datos de voz de gran volumen. No es siempre posible satisfacer estos requerimientos con los recursos disponibles. Además, no está clara su superioridad en reconocimiento respecto a otros tipos de modelado [Lee89]. Una alternativa, a medio camino entre el modelado discreto y el continuo, son los modelos *semicontinuos*, que también usan un sólo diccionario universal, pero incorporando varios candidatos de cuantización y sus distorsiones correspondientes. Huang [Huang90] ha demostrado que pueden ofrecer resultados superiores a los continuos, con un sustancial ahorro en la cantidad de cálculo (aunque sin llegar al nivel de los discretos). Una segunda posibilidad es el modelado con *Cuantización Vectorial Múltiple* (Multiple Vector Quantization, MVQ), recientemente propuesto por J. Segura [Segura91]. Estos modelos han sido introducidos como una forma directa de incorporar en el sistema la información perdida durante el proceso de cuantización vectorial al que es sometida la señal en un modelado discreto, con la ventaja de que no modifican esencialmente la estructura de dicho modelado discreto. Para ello, cada modelo HMM posee su propio diccionario VQ, y evalúa la distorsión media de una secuencia entrante en dicho diccionario. El nuevo modelado ha demostrado ser ampliamente superior al clásico modelado discreto, con similar cantidad de cómputo en reconocimiento, pero claramente más barato en la fase de entrenamiento. También se ha demostrado que los modelos MVQ pueden igualar o mejorar el modelado semicontinuo con un sustancial abaratamiento del coste computacional en reconocimiento y entrenamiento.

Todo ésto nos llevó al planteamiento de un estudio de distintos tipos de modelado HMM, dirigido, esencialmente, a la selección y estimación de los parámetros de los modelos, así como su enriquecimiento con la incorporación de informaciones útiles que no son, en otros casos, aprovechadas. Estos trabajos se han venido realizando en el curso de estos últimos años, y los resultados obtenidos han ido

publicándose sucesivamente. En la presente memoria se incluyen algunos de esos trabajos y resultados, y otros inéditos presentados aquí por primera vez, que siguen, básicamante, las siguientes líneas:

1) Enriquecimiento de los vectores de características mediante la incorporación de información sobre la energía de la señal [Peinado90, Peinado91].

2) Estudio sobre las posibilidades de una estimación discriminativa de modelos tipo MVQ y obtención de nuevos algoritmos para el diseño VQ de tipo discriminativo[Peinado93].

3) Creación de un nuevo modelado que aprovecha los principales aspectos de los modelados semicontinuo y MVQ [Peinado94].

Debido a la ingente cantidad de pruebas y cálculo que conlleva este estudio, se ha preferido desarrollarlo en el ámbito de un sistema de reconocimiento de palabras aisladas, obviando todo tipo de consideración referente al lenguaje. Pero ha de tenerse en cuenta que las técnicas aquí propuestas son directamente trasladables a sistemas más complejos de palabras conectadas o voz continua, en los que nuestro equipo trabaja actualmente.

El presente trabajo se puede dividir en dos partes. En la primera (capítulos 1,2,3) se expondrán las bases teóricas del trabajo, y en la segunda (capítulos 4, 5, 6 y 7) se desarrollarán y experimentarán las técnicas introducidas. El capítulo 2 se centrará en la parametrización y modelización de la señal de voz, haciendo un repaso de técnicas de preprocesamiento y modelos HMM en sus distintas variantes. En el capítulo 3 se estudiará el problema de la estimación de parámetros mediante diversas técnicas, haciendo especial hincapié en el caso HMM. El capítulo 4 estará dedicado a la presentación de los datos empleados a lo largo del trabajo, así como a su estructuración para la realización de las experiencias. En el capítulo 5 se establecerán los sistemas de referencia con los que efectuar las comparaciones posteriores. También incluye un estudio sobre la introducción en el sistema de información referente a la energía de la señal. El capítulo 6 introducirá un sistema básico MVQ. Se buscará una forma apropiada para las densidades de probabilidad continuas usadas en este modelado, y se estudiarán las posibilidades de entrenamientos discriminativos. Ésto último servirá como punto de partida para la obtención de algoritmos discriminativos de diseño VQ. Finalmente, en el capítulo 7 se estudiará la integración de las técnicas semicontinua y MVQ, para la obtención

de un nuevo modelado híbrido, denominado *SCMVQ*. La presente memoria se cerrará con las conclusiones del trabajo y las perspectivas de futuro que abre.

Capítulo 2

RECONOCIMIENTO DE VOZ MEDIANTE MODELOS OCULTOS DE MARKOV

El objetivo de este capítulo es el de establecer los principios del reconocimiento de voz mediante Modelos Ocultos de Markov. Para ello, se realizará un estudio de la implementación de un sistema basado en modelos discretos. Esta implementación requerirá un estudio previo de los métodos de análisis y parametrización de la señal de voz, basados en modelos espectrales LPC. La comparación con modelos espectrales de referencia es un punto fundamental en la construcción de sistemas de RAH. Para estudiar la similitud entre dos modelos espectrales, se introducirá el concepto de medida de distorsión, así como algunas de las medidas de distorsión más importantes. También se efectuará un repaso de los principales aspectos de la técnica de Cuantización Vectorial, fundamental en el desarrollo de sistemas con modelos de Markov, y herramienta esencial en este trabajo. Posteriormente, se introducirán los Modelos Ocultos de Markov discretos y los principales aspectos relacionados con reconocimiento de voz. Se concluirá con la deducción y presentación de otras variantes del modelado de Markov mediante un formalismo unificado.

2.1 Análisis de señales de voz

A pesar de la gran variedad de posibilidades existente para la implementación de sistemas de reconocimiento, quizás el denominador común de todos ellos es la etapa de análisis, que convierte a la señal de voz a algún tipo de representación paramétrica, reduciendo considerablemente la cantidad de información contenida en la señal original, y obteniéndose una forma apta para el procesado posterior.

Aunque existen una gran cantidad de opciones, la representación espectral de tiempo corto es probablemente la más usada [Rabiner78]. Existen dos métodos dominantes de análisis espectral: el modelo de análisis basado en un *banco de filtros* y el modelo de *Predicción Lineal* (Linear Prediction Coding, LPC). Para este trabajo se ha escogido el método LPC por diversas razones [Rabiner93]:

1) Proporciona un buen modelo de producción de voz, mejor para sonidos sonoros que sordos, aunque también aceptable para éstos últimos.

2) Es un modelo analíticamente tratable, con unos requerimientos de cálculo inferiores al caso de banco de filtros.

3) En reconocimiento proporciona unos resultados tan buenos o mejores que el análisis basado en banco de filtros.

El apartado siguiente se dedica precisamente a un estudio más en profundidad del método LPC.

2.2 Análisis LPC

El método LPC (ver figura 2.1) consiste en el análisis espectral de segmentos cortos de señal mediante una representación del tipo $G/A(e^{j\omega})$, donde $A(z)$ es un polinomio en z^{-1} de la forma,

$$A(z) = 1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_p z^{-p} \tag{2.1}$$

El objetivo de este análisis es obtener el conjunto de coeficientes representativos $\{a_k\}$, que posteriormente pueden ser transformados a otro tipo de parámetros más adecuados para la tarea a realizar. En esta sección, siguiendo el esquema de la figura 2.1, se repasarán aspectos del análisis LPC relacionados con la preparación de la señal, el modelo LPC de producción de voz y la conversión de parámetros.

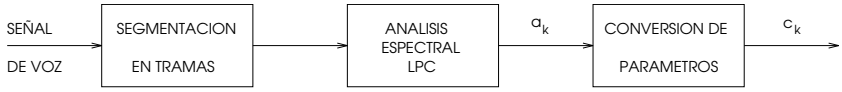


Figura 2.1: *Diagrama de bloques de análisis LPC.*

2.2.1 Preparación de la señal de voz

Antes de iniciar el proceso de análisis espectral de la señal de voz es conveniente someterla a un proceso de acondicionamiento. Una vez que la señal ha sido filtrada y muestreada (ver capítulo 4), ésta suele ser *preenfatizada*. El preénfasis es un proceso consistente en pasar la señal a través de un filtro digital cuya función de transferencia es [Markel76],

$$P(z) = 1 - \mu z^{-1} \tag{2.2}$$

donde μ es un factor real que toma valores inferiores y próximos a la unidad. Este filtro implementa, aproximadamente, la derivada de la señal, y su función es eliminar posibles niveles de contínua y elevar la parte de altas frecuencias del espectro, que presenta una caída promedio de 6 dB/dec en la voz humana.

Como se mencionó anteriormente, el método LPC se basa en el análisis de segmentos cortos de señal. Por tanto, una vez preenfatizada, la señal es segmentada en pequeños segmentos o *tramas* de 15-40 ms de duración (que pueden estar solapadas entre sí), en las que se considera que la señal es cuasi-estacionaria, de forma que se supone que los parámetros que caracterizan la señal permanecen constantes en esa trama. Esta secuencia de tramas se obtiene desplazando una ventana de análisis sobre la señal original. Es recomendable usar una ventana distinta de la rectangular para evitar los efectos de longitud finita de la misma (error de *leakage*) [Brigham74]. Una ventana muy usada es la de *Hamming*,

$$w(n) = 0.54 - 0.46 \cos\left[2\pi \frac{n-1}{L}\right] \tag{2.3}$$

donde n es un índice que recorre la ventana de longitud L [Brigham74]. Existen otros tipos de ventanas tales como la de Barlett, Hanning, Blackman, etc. [Oppenheim75].

2.2.2 Modelo LPC de producción de voz

El análisis LPC fue inicialmente usado en la codificación de la voz humana (Linear Prediction Coding, LPC) [Makhoul75]. El modelo LPC considera que una señal de voz $s(n)$ es la salida de un filtro todo-polos con excitación $u(n)$ [Makhoul75]. La función de transferencia de dicho filtro será de la forma,

$$H(z) = \frac{G}{A(z)} \quad con \quad A(z) = \sum_{k=0}^{p} a_k z^{-k} \;\; (a_0 = 1) \tag{2.4}$$

donde a_k $(k = 1, \ldots, p)$ y G son los coeficientes y la ganancia del filtro, respectivamente. La ecuación en diferencias correspondiente a la expresión anterior es,

$$s(n) = Gu(n) - \sum_{k=1}^{p} a_k s(n-k) \tag{2.5}$$

El modelo LPC (ver figura 2.2) se completa con una excitación que toma la siguiente forma:

1) Si la señal $x(n)$ corresponde a un sonido sonoro, $u(n)$ es un tren de impulsos unitarios, separados unos de otros por un número de muestras constante, denominado *periodo de fundamental* o *pitch*, que corresponde a la inversa de frecuencia a la que vibran las cuerdas vocales.

2 Si la señal corresponde a un sonido sordo, $u(n)$ es un ruido blanco estacionario gausiano de media nula y varianza unitaria,

Dada la señal $s(n)$, un *Predictor Lineal* de dicha señal se define como,

$$\tilde{s}(n) = -\sum_{k=1}^{p} \alpha_k s(n-k) \tag{2.6}$$

donde $\tilde{s}(n)$ es la señal predicha, y los α_k $(k = 1, \ldots, p)$ (p es el orden de predicción) son los *coeficientes de Predicción Lineal* o coeficientes LPC. Al efectuar la predicción, se comete un error (error de predicción) que viene dado por,

$$e(n) = s(n) - \tilde{s}(n) = s(n) + \sum_{k=1}^{p} \alpha_k s(n-k) \tag{2.7}$$

Los coeficientes LPC se calcularán como aquellos que hacen que la energía E del

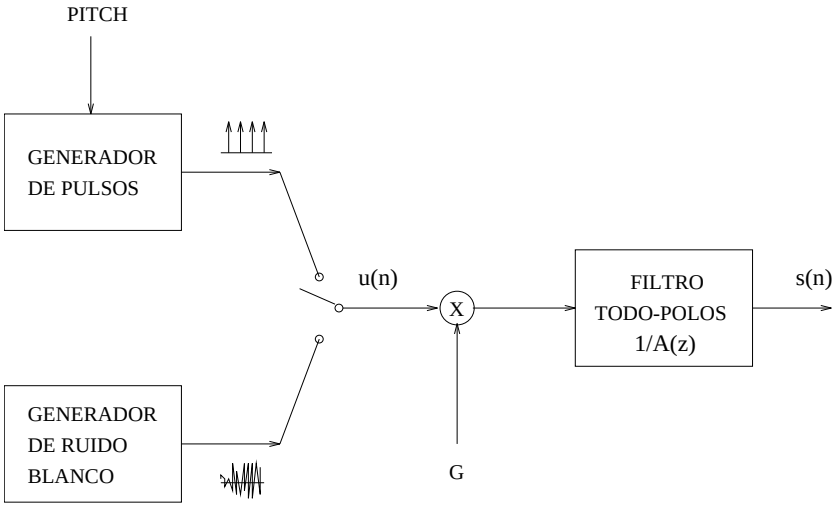


Figura 2.2: *Modelo de producción de voz LPC.*

error de predicción (o energía residual) sea mínima, es decir,

$$\frac{\partial E}{\partial \alpha_k} = 0 \qquad \text{con} \quad E = \sum_n e^2(n) \tag{2.8}$$

Se pueden identificar los coeficientes del filtro todo-polos como los coeficientes LPC ($a_k = \alpha_k$). Tres razones, al menos, avalan esta identificación [Rabiner78]:

a) Si $a_k = \alpha_k$, entonces $e(n) = Gu(n)$. Para sonidos sonoros la excitación es un tren de impulsos, por lo que $u(n)$ será muy pequeña la mayor parte del tiempo. Esto parece estar en consonancia con la búsqueda de unos coeficientes LPC que $\{\alpha_k\}$ que minimizen el error de predicción.

b) Puede comprobarse que si la señal es generada por la expresión (2.5) con coeficientes no variables en el tiempo y excitada por un impulso simple o ruido blanco estacionario, entonces los coeficientes de predicción resultantes de la minimización de la energía residual coinciden con los del filtro LPC de (2.5).

c) Una consideración práctica: el método de resolución propuesto en (2.8) conduce a un sistema de ecuaciones lineales que puede ser eficientemente resuelto.

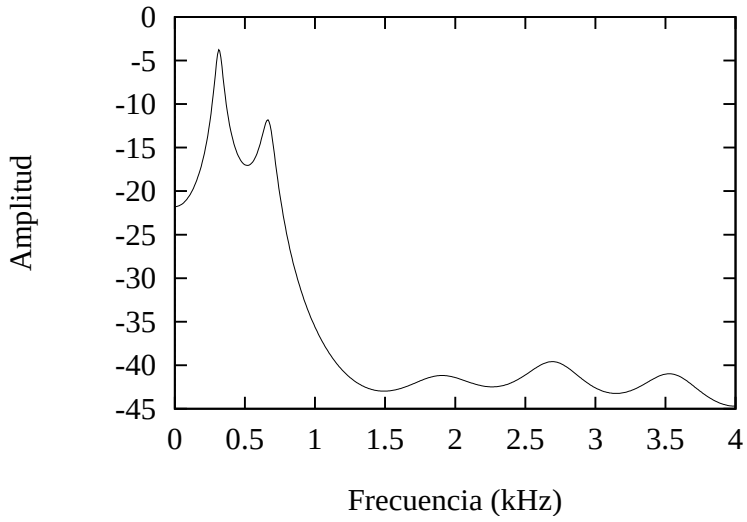


Figura 2.3: *Espectro LPC de un segmento de la vocal /u/.*

El sistema de ecuaciones que se deriva de (2.8) puede resolverse por dos métodos: autocorrelación y covarianza. En este trabajo se hace uso del primero, en conjunción con el algoritmo de Durbin, ya que de esta forma queda asegurada la estabilidad del filtro resultante [Rabiner78].

$H(e^{j\omega})$ es una representación autorregresiva del espectro de la señal,conocida como *espectro LPC*. En la figura 2.3 se representa el espectro $|H(e^{j\omega})|$ correspondiente a un segmento de la vocal /e/.

2.2.3 Coeficientes cepstrales

Tal como se indica en la figura 2.1, el conjunto de coeficientes LPC suele ser transformado para obtener una representación adecuada en reconocimiento. Las actuales técnicas de preprocesamiento de señales para reconocimiento de voz tienden a hacer uso de una derivación conocida como *cepstrum LPC* [Oppenheim75, Rabiner93], en lugar de usar los coeficientes LPC, los coeficientes PARCOR, u otras derivaciones. El cepstrum, $c(n)$, es la señal temporal correspondiente al espectro LPC

logarítmico, es decir,

$$\hat{H}(e^{j\omega}) = \log H(e^{j\omega}) = \sum_{n=-\infty}^{\infty} c(n)e^{j\omega n} \tag{2.9}$$

Las muestras del cepstrum suelen denominarse *coeficientes cepstrales*, y su dominio *cuefrencia* (aunque en realidad corresponde al dominio del tiempo). Los coeficientes cepstrales pueden obtenerse [Oppenheim75] mediante una relación recursiva que los relaciona con la respuesta impulsiva $h(n)$ del filtro $H(z)$ (supuesto causal y estable),

$$c(n) = \begin{cases} 0 & n < 0 \\ \log h(0) & n = 0 \\ \frac{h(n)}{h(0)} - \sum_{k=0}^{n-1} \frac{k}{n} c(k) \frac{h(n-k)}{h(0)} & n > 0 \end{cases} \tag{2.10}$$

Por otro lado, la respuesta impulsiva del filtro inverso $A(z)$ es,

$$h_A(n) = \begin{cases} 0 & n < 0 \\ 1 & n = 0 \\ a_n & 0 < n \leq p \\ 0 & n > p \end{cases} \tag{2.11}$$

Por tanto, los coeficientes cepstrales correspondientes a $A(z)$ se calculan (según (2.10)) como,

$$c_A(n) = \begin{cases} 0 & n < 0 \\ \log(1) & n = 0 \\ a_1 & n = 1 \\ a_n - \sum_{k=1}^{n-1} \frac{k}{n} c_A(k) a_{n-k} & n > 1 \end{cases} \tag{2.12}$$

Si tomamos espectros normalizados en ganancia $H(z) = 1/A(z)$, se tendrá que $\hat{H}(e^{j\omega}) = -\hat{A}(e^{j\omega})$, por lo que $c(n) = -c_A(n)$, lo que nos permite expresar el

cálculo recursivo de los coeficientes cepstrales en función de los coeficientes LPC,

$$c(n) = \begin{cases} 0 & n \leq 0 \\ -a_1 & n = 1 \\ -a_n - \sum_{k=1}^{n-1} \frac{k}{n} c(k) a_{n-k} & n > 1 \end{cases} \tag{2.13}$$

2.3 Distancias para procesado de señal

Una *medida de distorsión* o *distancia* [Gray80, Peinado89] se define como la asignación de una cantidad positiva a una pareja de espectros (de entrada y salida), que representa la distorsión resultante cuando la entrada es reproducida por la salida. Para que una distancia sea útil, debe poseer las siguientes propiedades: 1) ser subjetivamente significativa en el sentido de que distorsiones pequeñas y grandes correspondan a buenas y malas calidades subjetivas (en la sustitución de un espectro por otro), 2) ser analíticamente tratable, y 3) ser numéricamente tratable. Hay que hacer notar el abuso de lenguaje que supone el uso del término "distancia", ya que la definición de distancia requiere las propiedades de simetría y desigualdad triangular, que, en realidad, no son satisfechas por algunas distancias de procesado de señal, aunque se seguirá usando por ser más intuitivo, especialmente cuando se emplean representaciones vectoriales. A continuación se presentan algunas distancias de procesado de señal basadas en análisis LPC.

Una distancia clásicamente empleada en procesamiento digital de señales para medir la distorsión entre un espectro test H_t y otro de referencia H_r es la de *Itakura-Saito* [Gray80], definida como,

$$d_{IS}(H_t, H_r) = \int_{-\pi}^{\pi} \left| \frac{H_t(e^{j\omega})}{H_r(e^{j\omega})} - ln \frac{H_t(e^{j\omega})}{H_r(e^{j\omega})} - 1 \right| \frac{d\omega}{2\pi} \tag{2.14}$$

Esta distancia presenta el inconveniente de ser sensible a los cambios de energía de la señal. Esto puede evitarse usando una distancia normalizada en ganancia como es el la distancia *Razón de Semejanza* [Juang82], definida como,

$$d_{RS}(H_t, H_r) = \int_{-\pi}^{\pi} \left| \frac{A_r(e^{j\omega})}{A_t(e^{j\omega})} \right| \frac{d\omega}{2\pi} - 1 \tag{2.15}$$

Estas dos distancias se engloban en un amplio rango de medidas de distorsión conocidas como *distorsiones de semejanza*, dado que se derivan de una formu-

lación de "máxima semejanza" (ver capítulo 3) del análisis LPC. Aparte de las dos distancias mencionadas, existe un amplio rango de distancias que se engloba en esta familia [Rabiner93].

En los últimos años, se ha demostrado que las distancias definidas sobre el *cepstrum LPC* de una señal ayudan considerablemente a mejorar el rendimiento de los sistemas [Nocerino85]. La *distancia cepstral* se define como,

$$d_C(H_t, H_r) = \int_{-\pi}^{\pi} \left| logH_t(e^{j\omega}) - logH_r(e^{j\omega}) \right|^2 \frac{d\omega}{2\pi} = \sum_{n=-\infty}^{+\infty} \left(c_t(n) - c_r(n) \right)^2 \tag{2.16}$$

El sumatorio de la ecuación anterior puede aproximarse por una suma finita sobre los L primeros coeficientes cepstrales, con lo que la distancia cepstral se convierte en una distancia euclídea en el espacio de vectores $\mathbf{c}$ definido por dichos coeficientes,

$$d_C(\mathbf{c}_t, \mathbf{c}_r) = \sum_{n=1}^{L} \left(c_t(n) - c_r(n) \right)^2 = \left\| \mathbf{c}_t - \mathbf{c}_r \right\|^2 \tag{2.17}$$

suponiendo espectros normalizados en ganancia. En realidad, esto es equivalente aplicar una ventana en el dominio de la cuefrencia, en este caso rectangular. Este proceso también es conocido como *filtrado cepstral*. Aplicando diferentes tipos de ventanas se obtienen distintos tipos de distancias cepstrales. En el capítulo 5 se procedera a un estudio experimental de las mismas.

Estudios psicofisiológicos han demostrado que la habilidad humana para distinguir frecuencias sigue una función logarítmica de la frecuencia, y no lineal (como hasta ahora se ha considerado). Esta respuesta humana puede ser tenida en cuenta mediante una transformación de la escala de frecuencias a otra conocida *escala MEL* [Rabiner93], dada por las relaciones,

$$f = 600 \sinh(g/6) \tag{2.18a}$$

$$g = 6 \log \left[(f/600) + \sqrt{1 + (f/600)^2} \right] \tag{2.18b}$$

donde f va expresada en Hertzios (escala LINEAL) y g en Barks (escala MEL). La gráfica 2.4 muestra la relación existente entre las escalas MEL y lineal.

Otra posibilidad de compresión del eje frecuencial es aplicar una *transfor-*

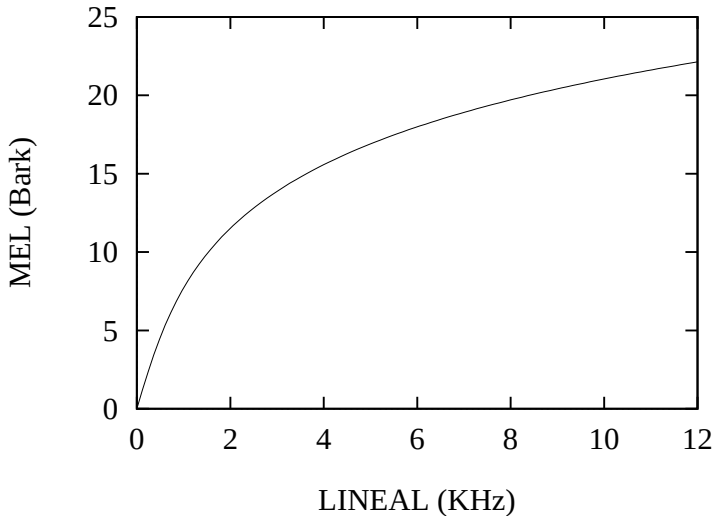


Figura 2.4: *Relación entre las escalas de frecuencia MEL y lineal.*

mación bilineal [Oppenheim75] de la siguiente forma,

$$z_t^{-1} = \frac{z^{-1} - a}{1 - az^{-1}} \tag{2.19}$$

$$\omega_t = \omega + 2\tan^{-1}\left[\frac{a\sin\omega}{1 - a\cos\omega}\right] \tag{2.20}$$

donde z_t y ω_t representan las variables z y ω transformadas, y a es un parámetro de transformación que toma valores entre -1 y 1. Usando valores de a comprendidos entre 0.4 y 0.8 la escala de frecuencias obtenida mediante transformación bilineal es similar a la escala MEL. Estas transformaciones del eje de frecuencia pueden ser directamente aplicadas a cualquier distancia espectral para aproximar la habilidad humana para discernir sonidos [Lee89].

Como se puede ver en el caso de la distancia cepstral, es posible dar un formalismo vectorial al problema de la comparación de espectros, asignando a cada espectro un *vector de características*. Este formalismo es adecuado para codificar la voz mediante Cuantización Vectorial (objetivo del próximo apartado). El espacio vectorial de características recibe el nombre de *espacio de representación*. También

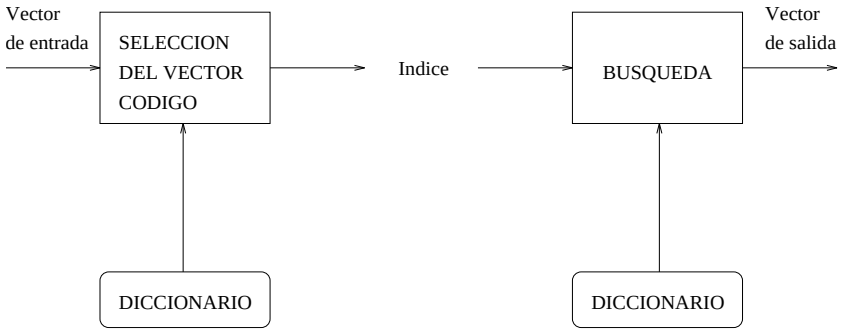


Figura 2.5: *Esquema general de un cuantizador vectorial.*

es posible dar una representación vectorial a las distancias IS y RS [Peinado89]. En lo sucesivo, se considerará que cada señal que llega al sistema es segmentada en tramas, cada una de las cuales queda representada por un vector de características, constituido por un conjunto de parámetros cuya elección depende del tipo de distancia empleada. Así, la señal completa queda finalmente representada por una secuencia de vectores $X = x_1, x_2, \ldots, x_T$ de longitud T. Estudios detallados sobre distancias de procesado de señal pueden encontrarse en [Gray80, Shikano91, Lee90b].

2.4 Cuantización Vectorial

La *Cuantización Vectorial* (Vector Quantization, VQ) fué aplicada por primera vez como técnica para codificación de la señal de voz por Buzo et al [Buzo80] en 1980, encontrando posteriormente una amplia aplicación en el campo del RAH. La idea básica es la de sustituir un cierto vector de entrada, obtenido previo análisis de un cierto segmento de señal, por un vector similar, llamado *vector código* perteneciente a un conjunto finito o *diccionario*. Cada vector código tiene asociado un índice que es transmitido en lugar del vector, reduciéndose considerablemente la cantidad de información que se ha de transmitir (ver figura 2.5). Así vista, la técnica VQ puede considerarse como una forma de Reconocimiento de Patrones, en la que un "objeto de entrada" es sustituido por el "objeto reconocido". También puede considerarse a la VQ como una generalización de la Cuantización Escalar al pasar de un espacio monodimensional a otro multidimensional. Un amplio tratamiento de la VQ para

codificación de señales de voz puede encontrarase en [Gray84] y [Gersho91].

Un cuantizador vectorial Q puede considerarse como una aplicación inyectiva del espacio $\mathcal{R}^p$ p-dimensional en un conjunto finito de vectores C (diccionario),

$$Q : \mathcal{R}^p \longrightarrow C \tag{2.21}$$

con $C = \{\mathbf{y}_i \in \mathcal{R}^p, i = 1, \ldots, N\}$. El cuantizador Q lleva asociada una partición en celdas S_i tal que $S_i = Q^{-1}(\mathbf{y}_i)$. En realidad, la operación de un cuantizador, como el de la figura 2.5, se descompone en dos, una $E : \mathcal{R}^p \longrightarrow J$ para codificación, y otra $D : J \longrightarrow C$ para decodificación, de forma que $Q(\mathbf{x}) = D(E(\mathbf{x}))$.

Para evaluar convenientemente un cuantizador vectorial se suele usar alguna distancia de procesado de señal, que se notará genéricamente como $d(\mathbf{x}, \mathbf{y})$, que indique la distorsión introducida al sustituir un vector $\mathbf{x}$ por otro $\mathbf{y}$. Así, la precisión de un cuantizador puede medirse mediante la distorsión media $E\left[d(\mathbf{x}, Q(\mathbf{x}))\right]$ a que da lugar la sustitución de $\mathbf{x}$ por su versión cuantizada $\mathbf{y}$.

Dado un diccionario C, la forma óptima de la función Q, en el sentido de distorsión media mínima, viene dada por la *regla del vecino más próximo*,

$$Q(\mathbf{x}) = \mathbf{y}_i \quad \text{si} \quad d(\mathbf{x}, \mathbf{y}_i) \leq d(\mathbf{x}, \mathbf{y}_j) \quad \forall j \neq i \tag{2.22}$$

Esta regla da lugar a que la partición se realice como,

$$S_i = \{x \in \mathcal{R}^p \ : \ d(\mathbf{x}, \mathbf{y}_i) \leq d(\mathbf{x}, \mathbf{y}_j) \ \ \forall j \neq i\} \tag{2.23}$$

Por otro lado, dada una cierta partición, el diccionario óptimo, el que produce una distorsión media mínima, es aquel para el que los vectores código son los *centros* de las celdas S_i de la partición, definidos como,

$$\mathbf{y}_i = cent(S_i) = \min_{\mathbf{y}}{}^{-1} E\left[d(\mathbf{x}, \mathbf{y})|x \in S_i\right] \tag{2.24}$$

es decir, aquel vector para el que el valor esperado de la distancia entre un vector $\mathbf{x} \in S_i$ y él es mínima. Este resultado es conocido como *condición del centro* [Gersho91]. En concreto, usando distancias tipo euclídeo, cada centro $\mathbf{y}_i$ es el vector medio de la celda correspondiente S_i,

$$\mathbf{y}_i = cent(S_i) = E\left[\mathbf{x}|\mathbf{x} \in S_i\right] \tag{2.25}$$

El problema más importante en VQ es el diseño del diccionario óptimo C a par-

tir de un conjunto T de datos empíricos (conjunto de vectores de entrenamiento). El método más extendido es el *algoritmo de Lloyd generalizado*, más comúnmente conocido como *algoritmo LBG* (Linde-Buzo-Gray [Linde80]), o, también, *k-medias*, que consta de los siguientes pasos:

1) Se construye un diccionario inicial C_1. Se fija $m = 1$.

2) Dado un diccionario $C_m = \{\mathbf{y}_i\}$, se realiza una partición del conjunto T de datos en nubes S_i, usando la regla del vecino más próximo, tales que,

$$S_i = \{x \in T \ : \ d(\mathbf{x}, \mathbf{y}_i) \leq d(\mathbf{x}, \mathbf{y}_j) \ \ \forall j \neq i\}$$

3) Se calculan los centros de cada nube aplicando la condición del centro (ecuación (2.24)), obteniendo así un nuevo diccionario C_{m+1}. Si una celda estuviese vacía, se realizará una asignación alternativa (en lugar de calcular su centro) a dicha celda.

4) Se calcula la distorsión promedio de C_{m+1}. Si la variación de dicha distorsión respecto a la iteración anterior es suficientemente pequeña, el algoritmo concluye. En caso contrario se retorna al paso 2 con $m \leftarrow m + 1$.

Es posible comprobar que este algoritmo disminuye monótonamente la distorsión media y que converge en un número finito de iteraciones [Gersho91].

Existen múltiples alternativas para la elección del diccionario inicial. Puede escogerse aleatoriamente o como un subconjunto tomado del conjunto de vectores de entrenamiento. Una posibilidad más refinada es la *bipartición* iterativa del conjunto de datos [Gray84]. El algoritmo LBG conduce solamente a una solución localmente óptima. Existen métodos para la búsqueda de soluciones globalmente óptimas. Entre estas alternativas cabe destacar los métodos de *Simulated Annealing* [Vaisey88] o el *algoritmo de aprendizaje de Kohonen* aplicado a VQ [Yair92] (ver siguiente subapartado).

Un punto importante a considerar, es que al diseñar un diccionario VQ también se está diseñando un *clasificador* con densidades componentes tipo gausiano, en el que el vector media y la matriz de covarianza de cada componente coinciden con un centro VQ y la matriz de covarianza de los vectores contenidos en su celda asociada, respectivamente. En concreto, el algoritmo LBG es una aproximación a una estimación de *Máxima Semejanza* de dicho clasificador [Duda73] (en el capítulo 3 se dan detalles sobre este tipo de estimación).

2.4.1 Algoritmo de aprendizaje de Kohonen y LVQ

Aunque el *algoritmo de aprendizaje de Kohonen* (Kohonen's Learning Algorithm, KLA) se desarrolla originalmente para el aprendizaje no supervisado de ANNs, es fácilmente aplicable al entrenamiento de cuantizadores vectoriales [Kohonen90]. En lo que sigue se usará únicamente distancia euclídea.

El método KLA está esencialmente basado en el algoritmo competitivo que se expone a continuación. Se parte de un conjunto de vectores de entrenamiento $T = \{\mathbf{x}(t)\}$, tal que, en cada instante t, el sistema es entrenado por un vector $\mathbf{x}(t)$ (los vectores llegan secuencialmente). Se notará como $\mathbf{y}_c(t)$ al centro vecino más próximo a $\mathbf{x}(t)$. El algoritmo competitivo es el siguiente,

$$\begin{array}{ll} \mathbf{y}_c(t+1) = \mathbf{y}_c(t) + \alpha(t)\left[\mathbf{x}(t) - \mathbf{y}_c(t)\right] & \\ \mathbf{y}_i(t+1) = \mathbf{y}_i(t) & \forall i \neq c \end{array} \tag{2.26}$$

donde $\alpha(t)$ es una secuencia monótonamente decreciente de valores tales que $0 < \alpha(t) < 1$.

Basándose en la existencia de interacciones laterales entre neuronas en el entrenamiento de ANNs de inspiración biofísica, Kohonen propone que cada vector de entrada debe entrenar no sólo al vecino más próximo, sino a todos los centros de una vecindad N_c definida en torno a él. El radio de la vecindad puede variar a lo largo del tiempo (es conveniente que vaya reduciéndose durante el entrenamiento). El algoritmo KLA es el siguiente,

$$\begin{array}{ll} \mathbf{y}_i(t+1) = \mathbf{y}_i(t) + \alpha(t)\left[\mathbf{x}(t) - \mathbf{y}_c(t)\right] & \mathbf{y}_i \in N_c(t) \\ \mathbf{y}_i(t+1) = \mathbf{y}_i(t) & \mathbf{y}_i \notin N_c(t) \end{array} \tag{2.27}$$

Si el cuantizador vectorial va a ser usado como un "clasificador", hay que asociar a cada clase un conjunto de centros, de forma que la clase del centro más próximo a un cierto vector de entrada define el objeto reconocido. En este caso, es obvio que el proceso de entrenamiento debe realizarse de distinta manera. En lugar de una representación lo más fidedigna posible del vector de entrada, lo que interesa ahora es obtener un clasificador que discrimine la clase correcta de dicho vector. Es decir. el objetivo ahora es obtener una fronteras de decisión entre clases tan óptimas como sea posible. Para ello, Kohonen propone una serie de estrategias conocidas como *Learning Vector Quantization* (LVQ) [Schalkoff92, Kohonen90]. A diferencia de los algoritmos VQ presentados hasta ahora, de naturaleza no supervisada, los métodos LVQ son supervisados, es decir, se debe disponer de un

conjunto de vectores de entrenamiento de clase conocida. Se puede realizar el siguiente proceso de inicialización,

1) Se obtiene un diccionario entrenado por cualquier técnica VQ (incluyendo el algoritmo KLA antes expuesto).

2) Se presentan una serie de vectores de entrada de clase conocida. Cada centro del diccionario se asocia a la clase a la que pertenecen la mayoria de los vectores de entrada de los que es vecino más próximo.

Aunque con este procedimiento ya se dispone de un clasificador, existen distintas versiones LVQ para proceder a su refinamiento. Estas son las siguientes:

LVQ1 La idea básica es alejar los centros de las superficies de decisión para delimitar, de una forma más precisa, los bordes de cada clase. El algoritmo es el siguiente,

$$\begin{array}{ll} \mathbf{y}_c(t+1) = \mathbf{y}_c(t) + \alpha(t)\left[\mathbf{x}(t) - \mathbf{y}_c(t)\right] & \mathbf{x}(t) \text{ correctamente clasificado} \\ \mathbf{y}_c(t+1) = \mathbf{y}_c(t) - \alpha(t)\left[\mathbf{x}(t) - \mathbf{y}_c(t)\right] & \mathbf{x}(t) \text{ incorrectamente clasificado} \\ \mathbf{y}_i(t+1) = \mathbf{y}_i(t) & \forall i \neq c \end{array} \tag{2.28}$$

LVQ2 El algoritmo anterior puede ser fácilmente modificado para ajustarse mejor a la filosofía de decisión dada por la regla de Bayes. Supóngase que $\mathbf{y}_i$ e $\mathbf{y}_j$ son una pareja de centros vecinos más próximos en el espacio de representación y que corresponden a clases distintas, tal y como se representa en la figura 2.6 (las curvas representan las densidades de probabilidad de las clases correspondientes). La superficie de discriminación (incorrecta) es el plano medio entre $\mathbf{y}_i$ e $\mathbf{y}_j$. Se define una ventana centrada en el plano medio, y las correcciones son únicamente realizadas si el vector de entrada $\mathbf{x}$ cae dentro de la ventana y, además, en el lado incorrecto, de forma que el plano medio se mueve hacia la superficie de separación de las densidades (coincidiendo

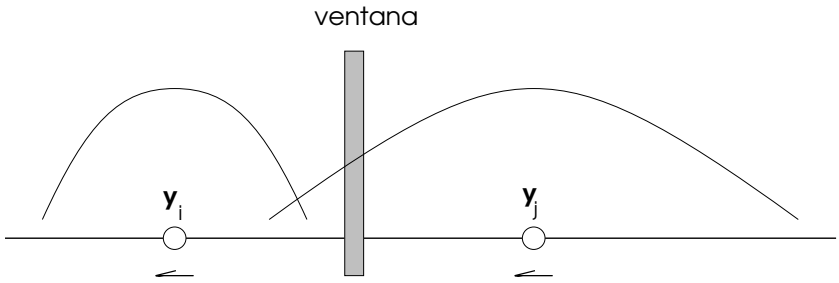


Figura 2.6: *Ilustración de la ventana usada en LVQ2 y LVQ3.*

asintóticamente con ella). El algoritmo de corrección es el siguiente,

$$\left.\begin{aligned}\mathbf{y}_i(t+1) &= \mathbf{y}_i(t) - \alpha(t)\left[\mathbf{x}(t) - \mathbf{y}_i(t)\right] \\ \mathbf{y}_j(t+1) &= \mathbf{y}_j(t) + \alpha(t)\left[\mathbf{x}(t) - \mathbf{y}_j(t)\right]\end{aligned}\right\}$$ Si C_i es la clase más próxima, pero $\mathbf{x}(t) \in C_j \neq C_i$, siendo C_j el segundo vecino más próximo. Además, $\mathbf{x}(t)$ debe caer dentro de la ventana.

$$\mathbf{y}_k(t+1) = \mathbf{y}_k(t)$$ En cualquier otro caso

(2.29)

El ancho de la ventana debe ser determinado experimentalmente.

LVQ3 En el algoritmo LVQ2 pueden detectarse 2 efectos no deseados:

1) Como las correcciones son proporcionales a la diferencia entre el vector de entrada y el centro considerado, la corrección sobre $\mathbf{y}_j$ es de mayor magnitud que la realizada sobre $\mathbf{y}_i$. La consecuencia de ésto es que $\mathbf{y}_i$ e $\mathbf{y}_j$ tienden a acercarse. Este problema puede solventarse exigiendo únicamente que el vector de entrada caiga dentro de la ventana.

2) El proceso definido en LVQ2 puede conducir a un nuevo equilibrio asintótico no óptimo de $\mathbf{y}_i$. Es necesario incluir correcciones que aseguren que $\mathbf{y}_i$ continúa aproximando las densidades de las clases.

Combinando las 2 ideas anteriores se obtiene el algoritmo LVQ3,

$$\left.\begin{aligned}\mathbf{y}_i(t+1) &= \mathbf{y}_i(t) - \alpha(t)\left[\mathbf{x}(t) - \mathbf{y}_i(t)\right] \\ \mathbf{y}_j(t+1) &= \mathbf{y}_j(t) + \alpha(t)\left[\mathbf{x}(t) - \mathbf{y}_j(t)\right]\end{aligned}\right\}$$ Si $\mathbf{y}_i$ e $\mathbf{y}_j$ son 2 centros vecinos más próximos, y $\mathbf{x}(t)$ es de la clase C_j y está dentro de la ventana.

$$\mathbf{y}_k(t+1) = \mathbf{y}_k(t) + \epsilon\alpha(t)\left[\mathbf{x}(t) - \mathbf{y}_k(t)\right]$$ Para $k \in \{i, j\}$, si $\mathbf{y}_i$, $\mathbf{y}_j$ y $\mathbf{x}(t)$ son de la misma clase.

$$\mathbf{y}_l(t+1) = \mathbf{y}_l(t)$$ En cualquier otro caso (2.30)

El valor óptimo de ϵ parece depender del tamaño de la ventana, siendo más pequeño para ventanas más pequeñas. Para un número suficientemente grande de iteraciones, el algoritmo parece converger a un punto estacionario.

Obsérvese que mientras LVQ1 actualiza un sólo centro en cada iteración, LVQ2 y LVQ3 actúan sobre 2 centros simultáneamente.

2.5 Modelado de procesos

Un problema clásico en Física e Ingeniería es el de la obtención de un *modelo* para un determinado proceso del mundo real. Las técnicas de *modelado* han ofrecido una gran ayuda en problemas tan diversos como la comprensión del movimiento planetario o el estudio de la estructura atómica. La aplicación de modelos a señales es importante por varias razones. En primer lugar, porque el conocimiento de un modelo de señal puede ayudar en su procesado (p. ej., limpiar esa señal si está distorsionada). También, un modelo puede ser de gran utilidad en la comprensión de la fuente que ha generado la señal, sin tener que disponer físicamente de ella. Finalmente, el uso de modelos puede ser de gran utilidad en la construcción de sistemas de predicción, reconocimiento o identificación.

Es posible establecer dos tipos de modelos de señal: deterministas y probabilísticos. Los primeros se basan en la extracción de características específicas de la señal. Por ejemplo, una señal senoidal queda completamente especificada mediante su amplitud, frecuencia y fase. Por el contrario, los modelos probabilísticos se construyen a partir de características estadísticas de la señal. Este modelado se basa en la suposición de que la señal puede ser caracterizada mediante un proceso

aleatorio cuyos parámetros pueden ser estimados convenientemente. Ejemplos de este tipo son los procesos gausianos, poisonianos, de Markov, y Ocultos de Markov.

Los próximos apartados estarán dedicados al modelado acústico de señales mediante Modelos Ocultos de Markov (Hidden Markov Models, HMM) y su aplicación al reconocimiento automático del habla. Aunque la teoría básica sobre HMMs es conocida desde los años 60, no fue aplicada a RAH por primera vez hasta mediados de los años 70, como ya se señaló en el capítulo primero, y sólo ampliamente adoptada por la comunidad científica en los 80. Sobre los fundamentos teóricos de los HMMs se puede encontrar amplia información en [Levinson83] y sobre su aplicación en RAH en [Rabiner83, Rabiner89a].

2.6 Procesos discretos de Markov

La teoría sobre HMMs se desarrolla como una generalización de los Procesos de Markov. En lo que sigue se centrará la atención sobre éstos últimos. Supóngase un proceso (ver figura 1.4) que puede ser descrito por un conjunto de N estados $\{s_1, s_2, \ldots, s_N\}$, cada uno representando un cierto evento físico u observación, de forma que el sistema cambia de estado cada cierto intervalo de tiempo (transición). Se notará con q_t al estado correspondiente al instante t.

Los procesos de Markov se caracterizan por la dependencia que presenta el estado actual respecto a los anteriores, o, dicho de otra forma, por poseer "memoria". En el caso de un *proceso de Markov discreto de primer orden*, el estado actual depende exclusivamente del estado anterior, independientemente del instante considerado. Este proceso queda definido por las *probabilidades de transición* de un estado a otro,

$$a_{ij} = P(q_t = s_j | q_{t-1} = s_i) \tag{2.31}$$

con

$$\sum_{j=1}^{N} a_{ij} = 1 \tag{2.32}$$

Como ejemplo, supóngase un modelo de la situación meteorológica de una determinada área, que reduzca el estado del tiempo a 3 estados (observaciones): lluvioso (s_1), nublado (s_2) y soleado (s_3). El modelo (se correspone con el de la figura 1.4) permitiría calcular la probabilidad de observar, por ejemplo, de tener la secuencia "soleado-soleado-lluvioso-nublado" ($O = (s_3, s_3, s_1, s_2)$) de la siguiente

forma,

$$P(O|Modelo) = P(s_3 s_3 s_1 s_2|Modelo) = \\ P(s_3)P(s_3|s_3)P(s_1|s_3)P(s_2|s_1) = \\ \pi_3 a_{33} a_{31} a_{12} \tag{2.33}$$

donde se ha usado la notación $\pi_3 = P(s_3)$ para la probabilidad inicial del estado s_3.

2.7 Modelos HMM discretos

Los procesos ocultos de Markov surgen como generalización de los procesos de Markov anteriormente estudiados. El siguiente ejemplo da cuenta de cómo se realiza esta transformación. Se pretende modelar el siguiente proceso: una persona va a realizar el experimento de lanzar monedas al aire, del que se obtiene una secuencia de caras (ca) y cruces (cr). El experimentador se encuentra oculto a los ojos de un observador, de forma que éste último sólo conoce el resultado del experimento, es decir, la secuencia de caras y cruces, sin saber cómo se ha realizado. El experimentador dispone de 3 monedas. Dos de ellas son usadas para obtener la secuencia de caras y cruces y están "cargadas", de forma que la primera moneda presenta un 75% de probabilidad de obtener cara, mientras que la segunda presenta un 75% para cruz. La tercera moneda, no cargada, es usada por el experimentador para decidir cuál de las otras dos es arrojada cada vez. Se puede construir un modelo del experimento con dos estados (cada uno representando una de las monedas que se arrojan). En cada estado es posible generar "ca" o "cr" (observaciones) según las probabilidades indicadas en la figura 2.7. Se acaba de construir un modelo oculto de Markov, en el que a diferencia de los procesos de Markov anteriormente analizados, se presentan dos procesos estocásticos superpuestos. Uno de ellos es observable (secuencia de observaciones, caras y cruces) y otro "oculto" (secuencia de estados, monedas arrojadas). Al igual que ocurría con los procesos de Markov, los modelos HMM permiten obtener la probabilidad de una secuencia de observaciones $O = o_1, \ldots, o_T$ de longitud T, por ejemplo, O=(ca,ca,cr,ca,cr,cr,cr). Pero antes de este punto, se definirán formalmente los modelos HMMs.

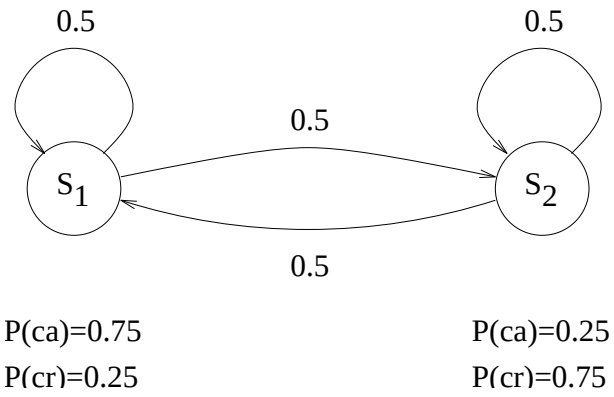


Figura 2.7: *Modelo HMM para el experimento de las monedas.*

2.7.1 Definición de HMM discreto

Un HMM discreto está caracterizado por los siguientes elementos:

1) Un conjunto S con N estados, $S = \{s_1, s_2, \ldots, s_N\}$, que están interconectados entre sí, de forma que en cada instante t el modelo se halla en un cierto estado, que se notará como q_t.

2) Un conjunto V con M símbolos de observación, $V = \{v_1, v_2, \ldots, v_M\}$. En cada instante de tiempo t el modelo genera un símbolo, que se notará como o_t.

3) Una matriz A de transiciones entre estados, definida como,

$$A = \{a_{ij}\} \quad \text{con} \quad a_{ij} = P(q_{t+1} = s_j | q_t = s_i) \quad (i, j = 1, \ldots, N) \tag{2.34}$$

Estas probabilidades verifican la siguiente condición de normalización,

$$\sum_{j=1}^{N} a_{ij} = 1 \quad (i = 1, \ldots, N) \tag{2.35}$$

4) Una matriz B de probabilidades de producción de símbolos en cada uno de los estados,

$$B = \{b_i(v_k)\} \quad \text{con} \quad b_i(v_k) = P(o_t = v_k | q_t = s_i) \\ (i = 1, \ldots, N; k = 1, \ldots, M) \tag{2.36}$$

Estas probabilidades verifican la siguiente condición de normalización,

$$\sum_{k=1}^{M} b_i(v_k) = 1 \quad (i = 1, \ldots, N) \tag{2.37}$$

5) Una matriz Π de estados iniciales,

$$\Pi = \{\pi_i\} \quad \text{con} \quad \pi_i = P(q_1 = s_i) \quad (i = 1, \ldots, N) \tag{2.38}$$

Estas probabilidades verifican la siguiente condición de normalización,

$$\sum_{i=1}^{N} \pi_i = 1 \tag{2.39}$$

Como se puede observar, basta proporcionar las matrices A, B y Π para definir completamente un cierto HMM. En lo sucesivo, se notará a éste conjunto de parámetros como $\lambda = (A, B, \Pi)$.

En la figura 2.8 se detalla un diagrama de flujo representando la generación de una secuencia $O = (o_1, o_2, \ldots, o_T)$. La secuencia de estados $Q = (q_1, q_2, \ldots, q_T)$ es desconocida. En lo sucesivo se hará referencia a Q como *camino*. Cada uno de los posible caminos tiene asociada una probabilidad $P(Q|O, \lambda)$.

2.8 Los tres problemas básicos

Es bastante usual [Rabiner89a, Lee89] agrupar en tres los problemas relacionados con los modelos HMM. Estos son:

1) Problema de evaluación: dada una secuencia O, hallar la probabilidad $P(O|\lambda)$ de que sea generada por el modelo λ.

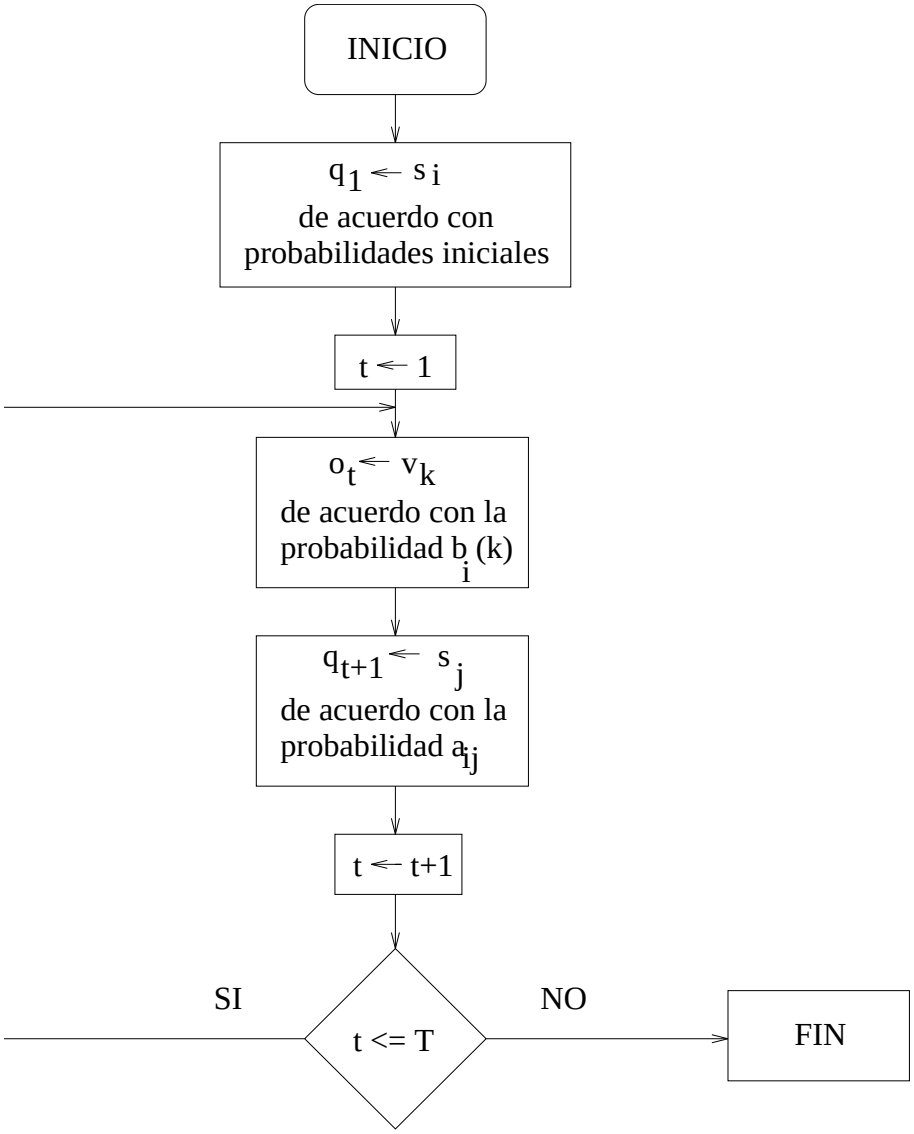


Figura 2.8: *Generación de una secuencia de símbolos por un modelo HMM.*

2) Problema de decodificación: dada una secuencia O y el modelo λ, hallar el camino Q óptimo.

3) Problema de estimación: dada una secuencia O, hallar el conjunto de parámetros λ que mejor se ajusta a dicha secuencia.

El primero es el problema básico del reconocimiento. Dada una secuencia de entrada O al sistema, éste tendrá que evaluarla en los distintos modelos que componen el sistema de reconocimiento (cada uno representando una clase a reconocer), de forma que aquel modelo λ para el que $P(O|\lambda)$ sea máxima corresponderá a la clase reconocida. El segundo problema plantea la recuperación de la parte oculta del proceso, es decir, la secuencia de estados Q. Desgraciadamente, no existe una respuesta única, y, por tanto, deberá elegirse un criterio de optimización, como se verá más adelante. El proceso de decodificación tiene especial interés y aplicaciones en el reconocimiento de voz contínua, donde se construye un "macromodelo", incluyendo todas las opciones permitidas por la gramática, sobre el que se debe encontrar el camino óptimo (la secuencia "oculta" de estados proporciona el texto reconocido). El último de los problemas se refiere a la estimación de parámetros de los modelos. Para construir (o entrenar) un sistema de reconocimiento, se deberán estimar los parámetros de los distintos modelos del sistema a partir de un conjunto de datos empíricos (secuencia de entrenamiento). Nuevamente, no hay una única respuesta, pero, en cualquier caso, el método que se siga debe perseguir el objetivo de hacer el rendimiento del sistema tan alto como sea posible (o, equivalentemente, minimizar la posibilidad de error). Dada la complejidad de este problema, se dedicará el próximo capítulo exclusivamente al estudio de los principales métodos que lo abordan. En lo que sigue se estudiarán y propondrán soluciones a los dos primeros.

2.8.1 Solución al problema de evaluación

Como ya se indicó en el ejemplo de las monedas, el establecimiento de un modelo permite el cálculo de la probabilidad $P(O|\lambda)$ de una determinada secuencia $O = (o_1, o_2, \ldots, o_T)$. La forma más directa de hacerlo es evaluar primero la secuencia dado un cierto camino,

$$P(O|Q, \lambda) = b_{q_1}(o_1)b_{q_2}(o_2)\cdots b_{q_T}(o_T) \tag{2.40}$$

y la probabilidad de dicho camino,

$$P(Q|\lambda) = \pi_{q_1} a_{q_1 q_2} a_{q_2 q_3} \cdots a_{q_{T-1} q_T} \tag{2.41}$$

Sumando la probabilidades conjuntas de secuencia y camino $P(O, Q|\lambda)$ (que es el producto de estas dos últimas) para todos los caminos posibles se obtiene la probabilidad de que O sea generada por λ,

$$P(O|\lambda) = \sum_{Q} P(O|Q, \lambda) P(Q|\lambda) =$$

$$\sum_{Q} \pi_{q_1} b_{q_1}(o_1) a_{q_1 q_2} b_{q_2}(o_2) a_{q_2 q_3} \cdots a_{q_{T-1} q_T} b_{q_T}(o_T) \tag{2.42}$$

Como hay N^T posibles caminos y cada camino requiere $2T - 1$ multiplicaciones, se precisan $N^T - 1$ sumas y $(2T - 1)N^T$ multiplicaciones, es decir, unas $2TN^T$ operaciones para calcular $P(O|\lambda)$ mediante (2.42). En una aplicación típica con $T = 50$ y $N = 5$ se requerirían del orden de 10^{37} cálculos, lo cual complicaría en exceso el proceso de evaluación de secuencias. Afortunadamente, se dispone de un algoritmo conocido como *Adelante-Atrás* (Forward-Backward) que simplifica considerablemente el cálculo. Se basa en el cálculo recursivo de una función *probabilidad hacia adelante* $\alpha_t(i)$, definida como,

$$\alpha_t(i) = P(o_1 o_2 \cdots o_t, q_t = s_i|\lambda) \tag{2.43}$$

es decir, la probabilidad de que la secuencia de símbolos, hasta el instante t, sea $o_1 o_2 \cdots o_t$ y el estado en dicho momento sea s_i, dado el modelo λ. El procedimiento es como sigue,

1) Iniciación:

$$\alpha_1(i) = \pi_i b_i(o_1) \qquad (1 \leq i \leq N) \tag{2.44}$$

2) Recursión:

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij} \right] b_j(o_{t+1}) \quad (1 \leq j \leq N; 1 \leq t \leq T - 1) \tag{2.45}$$

3) Fin:

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i) \tag{2.46}$$

El paso 1 inicia el procedimiento haciendo $\alpha_1(i)$ igual a la probabilidad conjunta de que s_i y o_1 tengan lugar conjuntamente en el primer instante. El paso 2 indica como calcular todos los $\alpha_{t+1}(j)$ siguientes de forma recursiva. Se calculan sumando todas las posibilidades de que se haya dado $o_1 o_2 \cdots o_t$ y de que el estado en $t+1$ sea s_j. En el último paso se aplica que,

$$P(O|\lambda) = \sum_{i=1}^{N} P(o_1 o_2 \cdots o_T, q_T = s_i|\lambda) \tag{2.47}$$

En el cómputo de (2.46) se requieren $(N-1)N(T-1)$ sumas y $N(N+1)(T-1)+N$ multiplicaciones, es decir, del orden de $2N^2T$ operaciones en total. Para el caso $N = 5$ y $T = 50$ se precisarían unas 2500 operaciones, lo que supone una reducción aproximada de 10^{33} en complejidad de cómputo frente al método directo.

También es posible resolver el problema usando una probabilidad hacia atrás $\beta_t(i)$ definida como,

$$\beta_t(i) = P(o_{t+1} o_{t+2} \cdots o_T|q_t = s_i, \lambda) \tag{2.48}$$

es decir, la probabilidad de la secuencia O desde $t+1$ en adelante, dado que el estado actual es s_i y el modelo es λ. De forma análoga al caso anterior es posible calcular estas probabilidades mediante un procedimiento iterativo de la siguiente manera,

1) Iniciación:

$$\beta_T(i) = 1 \qquad (1 \leq i \leq N) \tag{2.49}$$

2) Recursión:

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)$$
$$(1 \leq j \leq N; t = T-1, T-2, \ldots, 1) \tag{2.50}$$

3) Fin:

$$P(O|\lambda) = \sum_{i=1}^{N} \pi_i b_i(o_1) \beta_1(i) \tag{2.51}$$

La complejidad computacional es similar a la del caso anterior.

2.8.2 Solución al problema de decodificación

La dificultad de este problema reside, como ya se ha señalado, en establecer qué es un camino óptimo. Una posibilidad sería escoger en cada instante t el estado más probable. Para esto, se puede definir la variable,

$$\gamma_t(i) = P(q_t = s_i|O, \lambda) \tag{2.52}$$

que es la probabilidad de que el modelo λ se encuentre en el estado s_i en el instante de tiempo t, durante la generación de la secuencia de símbolos O. que puede ser expresada en función de las probabilidades adelante-atrás como,

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{P(O|\lambda)} = \frac{\alpha_t(i)\beta_t(i)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)} \tag{2.53}$$

con lo que se puede determinar el camino óptimo Q^* como,

$$q_t^* = \max_{1\leq i\leq N}{}^{-1} \left[\gamma_t(i)\right] \qquad (1 \leq t \leq T) \tag{2.54}$$

El problema de este criterio es que determina q_t^* en cada instante sin preocuparse si la secuencia resultante es globalmente óptima, o, incluso, si se trata de una secuencia de estados posible. Así, parece más razonable buscar el camino Q^* que maximiza la probabilidad $P(Q|O, \lambda)$ o, equivalentemente, $P(Q, O|\lambda)$, ya que

$$P(Q, O|\lambda) = P(Q|O, \lambda)P(O|\lambda) \tag{2.55}$$

y $P(O|\lambda)$ no depende del camino. Por tanto, el problema es encontrar el camino Q^* tal que,

$$Q^* = \max_{Q}{}^{-1} \left[P(Q, O|\lambda)\right] \tag{2.56}$$

Esto se puede resolver mediante un procedimiento recursivo, similar al Adelante-Atrás, conocido como *algoritmo de Viterbi*. Para ello hay que definir la siguiente variable,

$$\delta_t(i) = \max_{q_1 q_2 \cdots q_{t-1}} P(q_1 q_2 \cdots q_{t-1}, q_t = s_i, o_1 o_2 \cdots o_t|\lambda) \tag{2.57}$$

dada por la mejor secuencia $q_1 q_2 \cdots q_{t-1}$ que hace posible la generación de $o_1 o_2 \cdots o_t$ y que el estado actual sea $q_t = s_i$. El algoritmo se apoya en una función auxiliar $\phi_t(j)$ que permite recuperar el camino óptimo cuando concluye la recursión. El procedimiento es el siguiente:

1) Iniciación:

$$\delta_1(i) = \pi_i b_i(o_1) \qquad (1 \leq i \leq N) \tag{2.58a}$$

$$\phi_1(i) = 0 \tag{2.58b}$$

2) Recursión:

$$\delta_t(j) = \max_{1 \leq i \leq N} [\delta_{t-1}(i) a_{ij}]\, b_j(o_t) \quad (1 \leq j \leq N; 2 \leq t \leq T) \tag{2.59a}$$

$$\phi_t(j) = \max_{1 \leq i \leq N}{}^{-1} [\delta_{t-1}(i) a_{ij}] \quad (1 \leq j \leq N; 2 \leq t \leq T) \tag{2.59b}$$

3) Fin:

$$P^* = \max_{1 \leq i \leq N} [\delta_T(i)] \tag{2.60a}$$

$$q_T^* = \max_{1 \leq i \leq N}{}^{-1} [\delta_T(i)] \tag{2.60b}$$

4) Rastreo hacia atrás de Q^*:

$$q_t^* = \phi_{t+1}(q_{t+1}^*) \qquad (T-1, T-2, \ldots, 1) \tag{2.61}$$

La probabilidad P^* obtenida en el paso 3 puede considerarse una aproximación de $P(O|\lambda)$. Obsérvese la similitud entre en algoritmo de Viterbi y el Adelante-Atrás, de forma que también son similares las cantidades de cálculo consumidas, con la diferencia de sustituir sumas por decisiones de máximo. Esta cantidad de cálculo puede llegar a ser inmanejable cuando se tratan tareas de reconocimiento de voz contínua suficientemente complejas, en cuyo caso hay que recurrir a la limitación del número de candidatos posibles conforme avanza el algoritmo, ya sea mediante la imposición de un umbral de probabilidad [Schwartz85] o, como se ha propuesto recientemente, mediante búsqueda en profundidad de los mejores candidatos [Garcia93].

2.8.3 Escalamiento

Tanto el algoritmo Adelante-Atrás como el de Viterbi son procedimientos recursivos en los que fácilmente pueden aparecer problemas de "desbordamiento de cálculo", ya que las funciones α_t, δ_t y β_t se calculan a partir de términos del tipo,

$$\prod_{s=1}^{t-1} a_{q_s q_{s+1}} \prod_{s=1}^{t} b_{q_s}(o_s) \tag{2.62}$$

que tienden exponencialmente a cero cuando t va aumentando, lo cual hace necesario introducir cambios de escala o "escalamientos" en el cálculo recursivo de las funciones anteriormente citadas.

En el caso del algoritmo Adelante-Atrás hay que multiplicar cada $\alpha_t(i)$ (en cada paso de la recursión) por un coeficiente K_t que depende sólo de t y no del estado s_i, obteniéndose una nueva función escalada,

$$\tilde{\alpha}_t(i) = K_t \alpha_t(i) \qquad (i = 1, \ldots, N) \tag{2.63}$$

con

$$K_t = \prod_{s=1}^{t} c_s \tag{2.64}$$

Análogamente para la función $\beta_t(i)$,

$$\tilde{\beta}_t(i) = D_t \beta_t(i) \qquad (i = 1, \ldots, N) \tag{2.65}$$

con

$$D_t = \prod_{s=t}^{T} c_s \tag{2.66}$$

La implementación del algoritmo recursivo y la obtención de los coeficientes c_s se encuentra detallada en el apéndice A. En dicho apéndice se demuestra también que,

$$P(O|\lambda) = \frac{1}{K_T} = \frac{1}{K_t D_{t+1}} = \frac{1}{\prod_{s=1}^{T} c_s} \tag{2.67}$$

La expresión (2.67) puede dar también problemas de desbordamiento, por lo que

es aconsejable usar la probabilidad logarítmica,

$$\log P(O|\lambda) = -\sum_{s=1}^{T} \log c_s \tag{2.68}$$

Aunque también es posible realizar un escalamiento similar para el algoritmo de Viterbi [Peinado89], es más sencillo el uso directo de probabilidades logarítmicas [Segura91].

2.9 Generalización y tipos de modelado HMM

Durante todo el desarrollo anterior se ha supuesto que la señal de voz llega al núcleo del sistema de reconocimiento como una secuencia de observaciones $O = o_1, \ldots, o_T$ pertenecientes a un alfabeto discreto, obtenida, por ejemplo, después de someter a la secuencia de vectores de análisis $X = \mathbf{x}_1, \ldots, \mathbf{x}_T$ a un proceso VQ. Este proceso supone irremediablemente una pérdida de la información transportada por la secuencia X.

Para solventar este problema, es posible hacer que el modelo HMM trabaje directamente con las observaciones $\mathbf{x}_t$ pertenecientes al espacio de representación continuo p-dimensional $\mathcal{R}^p$. Para ello, es necesario sustituir la probabilidad de producción de símbolos $b_i(v_k)$ por una función densidad de probabilidad (*fdp*),

$$b_i(\mathbf{x}) = P(\mathbf{x}|s_i, \lambda) \tag{2.69}$$

con la condición de normalización,

$$\int_{\mathcal{R}^p} b_i(\mathbf{x}) d^p\mathbf{x} = 1 \tag{2.70}$$

La forma de representación más general para una *fdp*, para la que existe un procedimiento de estimación adecuado [Rabiner85], es una mezcla de *fdp*s de la forma,

$$b_i(\mathbf{x}) = P(\mathbf{x}|s_i, \lambda) = \sum_{v_k \in V(s_i,\lambda)} P(\mathbf{x}|v_k, s_i, \lambda) P(v_k|s_i, \lambda) \tag{2.71}$$

donde cada $P(\mathbf{x}|v_k, s_i, \lambda)$ es una *fdp* logarítmicamente cóncava o elípticamente simétrica, muy frecuentemente una gausiana multivariada con vector medio μ_k y matriz de covarianza Σ_k (también pueden usarse gausianas autorregresivas [Juang85]). Cada *fdp* es etiquetada por un índice v_k que varía en un conjunto

$V(s_i, \lambda)$ definido para el estado s_i del modelo λ. Los factores $P(v_k|s_i, \lambda)$ son los coeficientes de la mezcla (su suma extendida a $V(s_i, \lambda)$ debe ser la unidad).

Se acaba de definir un modelo HMM *continuo* (CHMM) (en las referencias [Rabiner85, Juang85, Rabiner85a] pueden encontrarse estudios más detallados sobre estos modelos). La tarea de reconocimiento está ahora basada en el cálculo de las densidades $P(X|\lambda)$. Las respuestas a los tres problemas básicos son totalmente análogas al caso discreto, sustituyendo probabilidades de producción por las densidades $b_i(\mathbf{x})$ que se han introducido.

2.9.1 Simplificaciones al modelado continuo

La gran cantidad de cómputo y número de parámetros que implica el uso de modelos continuos, ha motivado la búsqueda de simplificaciones. La primera de ellas corresponde al caso en que se considera que todos los estados de todos los modelos comparten el mismo conjunto de *fdp*s, es decir,

$$V(s_i, \lambda) = V \qquad \text{para todo} \qquad s_i, \lambda \tag{2.72}$$

y, por lo tanto,

$$b_i(\mathbf{x}) = P(\mathbf{x}|s_i, \lambda) = \sum_{v_k \in V} P(\mathbf{x}|v_k) P(v_k|s_i, \lambda) \tag{2.73}$$

Este nuevo tipo de HMM es conocido como modelado *semicontinuo* (SCHMM) [Huang90]. El sumatorio en (2.73) suele simplificarse extendiéndolo únicamente a los elementos más probables de V (mejores candidatos para $\mathbf{x}$). El conjunto de *fdp*s de V puede ser obtenido mediante la construcción de un diccionario VQ (tal como se expuso en el apartado 2.4).

Un caso especial del modelado anterior es el derivado de conservar sólo el mejor candidato o para el vector $\mathbf{x}$ en (2.73), lo cual sería válido en el caso de no existir solapamiento entre las distintas *fdp*s de V. Esta suposición da lugar a la siguiente aproximación,

$$P(\mathbf{x}|s_i, \lambda) = P(\mathbf{x}|o) P(o|s_i, \lambda) \tag{2.74a}$$

$$o = \max_{v_j \in V}{}^{-1} \left[P(\mathbf{x}|v_j) \right] \tag{2.74b}$$

La maximización de (2.74b) suele reducirse a la búsqueda del centro vecino más

próximo en el diccionario VQ. Es fácil comprobar que, la densidad $P(X|\lambda)$ se puede descomponer como el producto de una densidad de *probabilidad de cuantización*, independiente del modelo considerado, por una *probabilidad de generación* dada por el modelo λ,

$$\begin{aligned} P(X|\lambda) &= \sum_{Q} P(X|Q,\lambda)P(Q|\lambda) = \\ & P(X|O)\sum_{Q} P(O|Q,\lambda)P(Q|\lambda) = \\ & P(X|O)P(O|\lambda) \end{aligned} \tag{2.75}$$

Dado que $P(X|O)$ es la misma, cualquiera que sea el modelo λ empleado, es irrelevante en el proceso de reconocimiento, por lo que puede obviarse. Esto reduce el modelo al caso *discreto* (DHMM), anteriormente estudiado.

Una tercera simplificación puede derivarse suponiendo un conjunto de *fdp*s $V(\lambda)$ distinto para cada modelo λ y considerando que existe poco solapamiento entre las distintas densidades, por lo que,

$$P(\mathbf{x}|s_i,\lambda) = P(\mathbf{x}|o,\lambda)P(o|s_i,\lambda) \tag{2.76a}$$

$$o = \max_{v_j \in V(\lambda)}{}^{-1} \left[P(\mathbf{x}|v_j,\lambda)\right] \tag{2.76b}$$

En este caso también puede comprobarse, en forma similar al caso discreto, que la probabilidad $P(X|\lambda)$ se expresa como un producto de probabilidades,

$$P(X|\lambda) = P(X|O,\lambda)P(O|\lambda) \tag{2.77}$$

De la expresión (2.77) se deduce que esta nueva aproximación es muy similar a la discreta, pero ahora no es posible eliminar el factor correspondiente a la probabilidad de cuantización, por depender ésta del modelo considerado. Cada conjunto $V(\lambda)$ puede obtenerse nuevamente mediante la construcción de un diccionario VQ para cada modelo λ. Por ello, suele denominarse a esta aproximación modelado HMM con *cuantización dependiente* [Segura94] o *MVQ*, en referencia al uso de múltiple VQ.

2.10 Aplicación de HMMs a reconocimiento de voz

En los últimos diez años, gran número de investigadores han dedicado sus esfuerzos a la aplicación de la técnica HMM en RAH, en sus distintas vertientes (palabras aisladas, conectadas, voz contínua), y asignando distintas unidades a cada modelo (palabras o unidades inferiores). Como ya se introdujo en el primer capítulo, el proceso de reconocimiento de una señal representada por una secuencia de vectores X puede expresarse como la búsqueda del texto escrito W (frase) para el cual la probabilidad $P(W|X)$ es máxima. Como ya se indicó, esta probabilidad puede descomponerse como,

$$P(W|X) = \frac{P(X|W)P(W)}{P(X)} \tag{2.78}$$

La probabilidad a priori de la señal $P(X)$, al ser constante en el proceso, es obviada, $P(X|W)$ viene dada por el modelo acústico y $P(W)$ por el modelo de lenguaje. Precisamente, es en el modelado acústico donde los modelos HMM han encontrado mayor uso, asociando a cada frase W un modelo λ.

En la figura 2.9 se muestra un sistema básico de reconocimiento de palabras aisladas usando la propia palabra como unidad (sin modelo de lenguaje). En el caso DHMM habría que introducir en dicho diagrama un módulo VQ después del de análisis LPC/cepstrum, y el módulo HMM trabajaría con la secuencia cuantizada $O = o_1 \cdots o_T$.

En el caso de aplicación de modelos HMM a voz continua, se podría proceder de la misma manera que con palabras aisladas, asignando un modelo de Markov a cada una de las frases permitidas por la gramática. El número de posibles frases que se maneja normalmente en reconocimiento de voz puede hacer computacionalmente inviable esta solución. En la práctica lo que se suele hacer es construir un "macromodelo" que incluya todas las posibles frases permitidas por la gramática aplicada tal como se indicó en el apartado 2.8. Para reconocer una secuencia de entrada se aplica el algoritmo de Viterbi (con limitación del número de candidatos en cada paso de la iteración, como se indicó en el subapartado 2.8.2) sobre el macromodelo, de forma que la frase reconocida viene dada por el camino óptimo. La figura 2.10 representa el macromodelo correspondiente a una gramática sencilla.

En general, en la construcción de sistemas de reconocimiento basados en HMMs hay que tener en cuenta gran cantidad de aspectos. A continuación se mencionan tres de los más importantes.

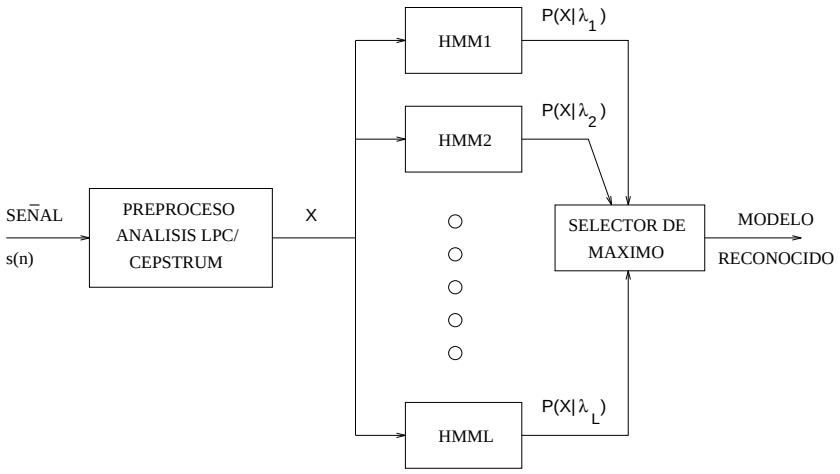


Figura 2.9: *Sistema de reconocimiento de palabras aisladas basado en modelos HMM.*

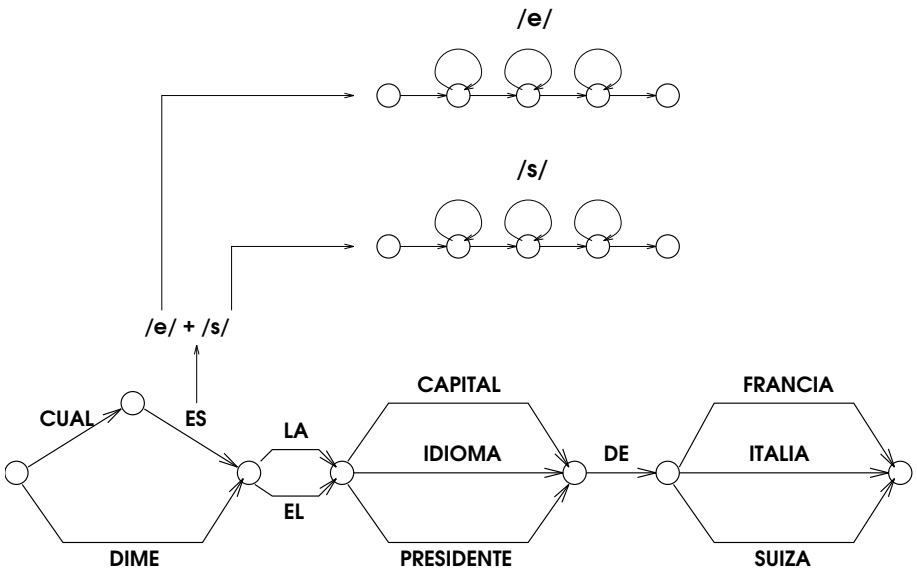


Figura 2.10: *Sistema de reconocimiento de voz continua basado en modelos HMM.*

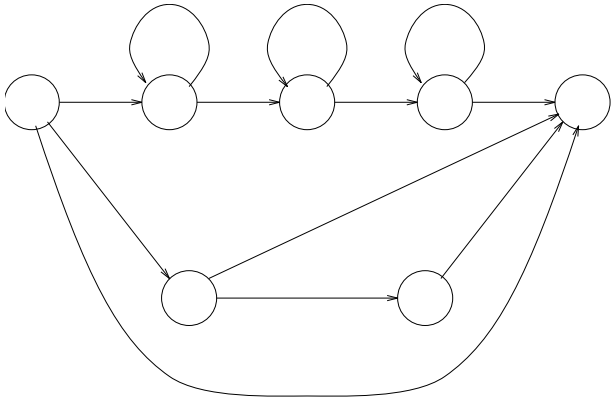


Figura 2.11: *Modelo HMM de fonema.*

Topología La topología más general corresponde al tipo *ergódico* que asegura la existencia de un camino desde un estado dado a otro cualquiera. La figura 1.4 corresponde a un modelo ergódico con 3 estados. En la construcción de sistemas de reconocimiento, la estructura más extendida es la *izquierda-a-derecha* o de *Bakis*, que corresponde a la figura 1.5 mostrada en el capítulo primero. La ventaja de esta topología es clara, por adecuarse a la naturaleza secuencial de la señal de voz. En el caso de voz contínua, hay autores [Lee89] que prefieren un modelo de fonema como el de la figura 2.11 para mejorar el modelado de la duración del fonema.

Modelado de duración Como se acaba de mencionar es posible modelar la duración de eventos acústicos mediante la topología del modelo. También es posible incluir explícitamente en la mecánica de los HMM densidades de duración de los estados $p_i(d)$ (probabilidad de obtener d observaciones consecutivas en el estado s_i) [Rabiner89a]. Pero la práctica ha demostrado que se pueden alcanzar resultados similares, y mucho menos costosos computacionalmente, calculando esas densidades heurísticamente a partir de la segmentación dada por Viterbi, e incluyéndolas en la probabilidad final de la secuencia en la siguiente manera [Peinado91],

$$\log \hat{P}(O|\lambda) = \log P(O|\lambda) + \alpha \sum_{i=1}^{N} \log p_i(d_i) \qquad (2.79)$$

donde α es un factor experimental y d_i es la duración, medida mediante Viterbi, del estado s_i (se supone topología izquierda-a-derecha).

Rechazo Un punto importante en la implementación de sistemas de reconocimiento es el rechazo de aquellas secuencias de entrada sobre las que no existe suficiente certeza respecto a su correcto reconocimiento una vez procesadas, por lo que el sistema no "pronuncia" ninguna decisión. Para llevar a cabo esta tarea pueden utilizarse informaciones auxiliares que son generadas durante el proceso de reconocimiento, tales como los valores de las probabilidades obtenidas en cada modelo (no sólo la detección del máximo), relaciones entre dichas probabilidades, duraciones de estados, etc. [Moreno90, Peinado91a].

Son muchos más los aspectos relacionados con el diseño de sistemas de RAH basados en modelos HMM. Parte de ellos serán analizados en el próximo capítulo, dedicado a estimación de parámetros, por estar íntimamente ligados a este problema.

Capítulo 3

ESTIMACIÓN DE PARÁMETROS EN MODELOS ACÚSTICOS

Una vez establecido el modelo acústico en el capítulo anterior, este capítulo se centra en el estudio de varios métodos de estimación de los parámetros del modelo. La estimación de parámetros consiste en la búsqueda de aquellos valores de los parámetros que hacen que el sistema se comporte de una forma óptima. La diferencia entre unos métodos y otros se deriva del uso de distintos criterios de optimización. En este capítulo se analizarán tres métodos: Máxima Semejanza, Máxima Información Mutua y Error de Clasificación Mínimo. Como ejemplo, en cada uno de los métodos, se procederá a la estimación de modelos HMM discretos. Al final del capítulo se abordarán algunos aspectos prácticos de estimación de parámetros en HMMs.

3.1 Planteamiento General

El problema de reconocimiento o *clasificación*, en la forma en que se ha tratado hasta el momento, puede formularse de la siguiente manera: al sistema de reconocimiento llega un vector X representando a un cierto tipo de objeto (para aplicación en RAH, X está formado por una secuencia de vectores), y se pretende encontrar el tipo o *clase* a la que pertenece dicho objeto. Se supone un conjunto finito $\{W_1, W_2, \ldots, W_L\}$ de clases. En el caso de que sean conocidas las probabilidades a posteriori $P(W_i|X)$, la regla de decisión de Bayes da la siguiente respuesta al problema enunciado,

$$W(X) = W_i \quad \text{si} \quad P(W_i|X) = \max_j P(W_j|X) \tag{3.1}$$

donde $W(X)$ es la función de clasificación. Esta regla conduce a un error de clasificación mínimo [Duda73]. La expresión (3.1) puede escribirse en función de las probabilidades a priori $P(W_j)$ de las clases y las probabilidades condicionales $P(X|W_j)$ como,

$$W(X) = W_i \quad \text{si} \quad P(X|W_i)P(W_i) = \max_j P(X|W_j)P(W_j) \tag{3.2}$$

Desgraciadamente, es poco frecuente conocer de forma exacta de las probabilidades $P(X|W_i)$ y $P(W_i)$, por lo que hay que recurrir a la construcción de modelos que, aunque de forma aproximada, permitan calcularlas. Dichos modelos vienen definidos por un conjunto de parámetros $\Omega = \{\Lambda, \Theta\}$, Λ para las probabilidades condicionales y Θ para las a priori. Esta modelización de las probabilidades se tendrá en cuenta usando la notación $P_\Lambda(X|W_i)$ y $P_\Theta(W_i)$ para cada una de las probabilidades. En el caso del RAH, estos modelos se denominan acústico (para las probabilidades condicionales) y de lenguaje (para las a priori), como ya se ha mencionado en capítulos anteriores. El estudio del modelo de lenguaje queda fuera del alcance de este trabajo, por lo que en lo sucesivo se supondrá que las probabilidades $P_\Theta(W_i)$ son conocidas (en concreto, una distribución equiprobable), y se centrará la atención en el estudio de los modelos acústicos y sus parámetros Λ. El proceso de modelado puede descomponerse en dos problemas diferenciados:

1) **Problema de Selección**: elegir el tipo de modelo que se va a emplear, lo cual ya fue abordado en el capítulo anterior mediante la adopción de HMMs para modelado acústico.

2) **Problema de Estimación**: estimar el conjunto de parámetros Λ que definen el modelo seleccionado. Lo usual es que el diseñador del sistema únicamente disponga de un conjunto de $\mathcal{S}$ vectores, $\mathcal{T} = \{X_1, X_2, \ldots, X_{\mathcal{S}}\}$, cada uno de los cuales pertenece a una clase conocida (cada vector está etiquetado). A partir de $\mathcal{T}$ debe obtenerse el conjunto de parámetros Λ que hace que el sistema rinda de forma óptima. El problema planteado también es conocido como problema de *aprendizaje* o de *entrenamiento*. La solución dependerá del criterio de *optimización* que se emplee.

En los últimos años ha ido surgiendo y aplicándose una gran cantidad de métodos para la estimación de modelos acústicos. El más simple y usado es el método de Máxima Semejanza (Maximum Likelihood, ML), basado en la suposición de independencia entre clases. Los esfuerzos más recientes han ido dirigidos a la implementación de métodos *discriminativos* que, al contrario que el método ML, tienen en cuenta la interrelación entre las distintas clases, con objeto de disminuir el error del sistema, que es, finalmente, el parámetro más importante para la evaluación del sistema. Entre ellos figura el método de *Máxima Información Mutua* (Maximum Mutual Information, MMI), propuesto por Bahl et al [Bahl86] que, basándose en argumentos de Teoría de la Información, busca el conjunto de parámetros que minimiza la longitud de código promedio necesaria para la correcta decodificación del texto W a partir de la señal acústica X. El método de *Información de Discriminación Mínima* (Minimum Discrimination Information, MDI), propuesto por Ephraim et al [Ephraim89] presenta un marco más general, englobando a las dos anteriores (ML y MMI) y está basado en la minimización de la función de *divergencia dirigida* entre la fuente modelada y su modelo. Aunque estas dos últimos métodos tienen como objetivo la reducción del error del sistema, lo cual se comprueba experimentalmente, no es posible demostrar teóricamente que estas estimaciones conducen a un esquema de reconocimiento que minimiza la probabilidad de error [Ephraim90]. Otras aproximaciones han enfocado de forma directa la minimización del error. Este es el caso del *entrenamiento correctivo* propuesto por Bahl et al [Bahl87], el entrenamiento para *Error Empírico Mínimo* propuesto por Ljolje et al [Ljolje90], y el entrenamiento para *Error de Clasificación Mínimo* (Minimum Classification Error, MCE) propuesto recientemente por Juang [Juang92].

Los próximos apartados están dedicados al estudio de varios de estos criterios, en concreto, los criterios de Máxima Semejanza, Máxima Información Mutua y Error de Clasificación Mínimo.

3.2 Estimación de Máxima Semejanza (ML)

El conjunto de datos de entrenamiento es dividido en L subconjuntos T_i, conteniendo cada uno todos los vectores de $\mathcal{T}$ que son de la misma clase. Se notará como S_i el cardinal de T_i. Se supone que cada probabilidad $P_\Lambda(X|W_i)$ tiene una forma paramétrica conocida, y queda por tanto determinada por un subconjunto de parámetros $\lambda_i \subset \Lambda$ (en adelante se escribirá $P_{\lambda_i}(X|W_i)$). El problema es estimar el conjunto de parámetros $\Lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_M\}$.

La estimación ML [Duda73] simplifica el problema suponiendo que el subconjunto de datos T_j no aporta ninguna información respecto a λ_i ($j \neq i$). Esta suposición permite entrenar cada clase de forma separada y simplificar la notación, suprimiendo el índice correspondiente a la clase. La *función de semejanza* para una clase W, dado un conjunto de entrenamiento para ella T, de cardinal S, y un conjunto de parámetros λ, se define como,

$$P_\lambda(T|W) = \prod_{k=1}^{S} P_\lambda(X_k \in T|W) \tag{3.3}$$

La estimación ML, $\hat{\lambda}$, de λ es aquella que hace máxima la función de semejanza,

$$\hat{\lambda} = \max_{\lambda}{}^{-1} P_\lambda(T|W) \tag{3.4}$$

Intuitivamente, $\hat{\lambda}$ corresponde al valor de λ que, en cierto sentido, mejor representa a los datos observados de la clase considerada.

Es bastante frecuente trabajar con el logaritmo de la función de semejanza en lugar de con ella misma, por ser más tratable a efectos de cómputo. Dado que el logaritmo es una función monótonamente creciente, la maximización de la primera implica la de la segunda. La forma estándar de llevar a cabo esta maximización es haciendo uso de la función gradiente,

$$\bigtriangledown_\lambda \log P_\lambda(T|W) = \sum_{k=1}^{S} \bigtriangledown_\lambda \log P_\lambda(X_k|W) = 0 \tag{3.5}$$

A. Nadas [Nadas83] demostró que, bajo ciertas suposiciones, la estimación ML conduce al conjunto de parámetros verdadero y, por tanto, a un rendimiento óptimo. Estas suposiciones son:

1) Que el conjunto de datos $\mathcal{T}$ ha sido generado por el modelo supuesto. Esto

es lo mismo que decir que se conoce el modelo verdadero.

2) Que el conjunto $\mathcal{T}$ tiene un número suficientemente grande de elementos.

3) El modelo verdadero de lenguaje es conocido.

4) El rendimiento del sistema no puede deteriorarse conforme los parámetros se aproximan a los verdaderos.

En sistemas de RAH, las suposiciones anteriores son difícilmente sostenibles:

1) Un modelo preciso debería incorporar los modelados del tracto vocal, cavidad nasal, labios, canal acústico y micrófono, lo cual es estrictamente falso.

2) Es difícil disponer de bases de datos excesivamente grandes. Además, sería poco práctico a efectos de cómputo.

3) Los estudios sobre lenguaje natural no permiten, por el momento, disponer de modelos de lenguaje suficientemente precisos.

4) La suposición número 4 es probablemente correcta, pero de poca aplicación al no disponer de un buen modelo.

Este conjunto de objeciones abre la puerta a otro tipo de estimaciones más robustas y coherentes con el objetivo de rendimiento óptimo (como se verá en los próximos apartados). A pesar de todo, la estimación ML sigue siendo ampliamente usada por su sencillez y buenos resultados en multitud de aplicaciones.

3.2.1 Estimación ML en modelos HMM discretos

Existe un algoritmo bien conocido para la estimación ML de modelos HMM, conocido como método de *Baum-Welch* [Levinson83]. Para la obtención de las fórmulas de reestimación que propone dicho método, se supondrá un modelo discreto λ y una sóla secuencia de entrenamiento $O = o_1 o_2 \cdots o_T$ de duración T. La clase W estará representada por el modelo λ, por lo que se abrevia la notación mediante,

$$P_\lambda(O|W) = P(O|\lambda) \tag{3.6}$$

que puede escribirse en función de las probabilidades adelante y atrás como,

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_t(i)\beta_t(i) \tag{3.7}$$

Para el desarrollo siguiente, conviene definir las siguientes probabilidades,

$$\xi_t(i,j) = P(q_t = s_i, q_{t+1} = s_j|O,\lambda) = \frac{\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)}{P(O|\lambda)} \tag{3.8a}$$

$$\gamma_t(i) = P(q_t = s_i|O,\lambda) = \sum_{j=1}^{N}\xi_t(i,j) = \frac{\alpha_t(i)\beta_t(i)}{P(O|\lambda)} \tag{3.8b}$$

Si se suma la función $\xi_t(i,j)$ para todo instante de tiempo t (excepto $t = T$), se obtiene el valor esperado del número de transiciones desde el estado s_i al s_j para la secuencia O. Asímismo, la misma operación sobre $\gamma_t(i)$ da como resultado el valor esperado del número de transiciones realizadas desde s_i. Por tanto,

$$\sum_{t=1}^{T-1}\xi_t(i,j) = \text{número esperado de transiciones desde } s_i \text{ a } s_j \tag{3.9a}$$

$$\sum_{t=1}^{T-1}\gamma_t(i) = \text{número esperado de transiciones desde } s_i \tag{3.9b}$$

Puede comprobarse (ver apéndice B) que las siguientes fórmulas de reestimación,

$$\hat{\pi}_i = \text{número esperado de veces en } S_i \text{ en el instante } (t=1) = \gamma_1(i) \tag{3.10a}$$

$$\hat{a}_{ij} = \frac{\text{número esperado de transiciones desde } s_i \text{ a } s_j}{\text{número esperado de transiciones desde } s_i} = \frac{\sum_{t=1}^{T-1}\xi_t(i,j)}{\sum_{t=1}^{T-1}\gamma_t(i)} \tag{3.10b}$$

$$\hat{b}_j(v_k) = \frac{\text{número esperado de veces en } s_j \text{ observando } v_k}{\text{número esperado de veces en } s_j} = \frac{\sum_{t=1}^{T}\gamma_t(j)\delta_{o_t,v_k}}{\sum_{t=1}^{T}\gamma_t(j)} \tag{3.10c}$$

donde $\delta_{x,y}$ es la función Delta de Dirac entre x e y, hacen que $P(O|\hat{\lambda}) \geq P(O|\lambda)$. Mediante la aplicación iterativa de éstas fórmulas puede alcanzarse un máximo local de probabilidad [Rabiner89a].

El uso de un conjunto de entrenamiento $T = \{O^1, O^2, \ldots, O^S\}$ con múltiples

secuencias genera el siguiente conjunto de fórmulas de reestimación,

$$\hat{\pi}_i = \frac{1}{\mathcal{S}} \sum_{l=1}^{S} \gamma_1^l(i) \tag{3.11a}$$

$$\hat{a}_{ij} = \frac{\sum_{l=1}^{S} \sum_{t=1}^{T^l-1} \xi_t^l(i,j)}{\sum_{l=1}^{S} \sum_{t=1}^{T^l-1} \gamma_t^l(i)} \tag{3.11b}$$

$$\hat{b}_j(k) = \frac{\sum_{l=1}^{S} \sum_{t=1}^{T^l} \gamma_t^l(j) \delta_{o_t^l, v_k}}{\sum_{l=1}^{S} \sum_{t=1}^{T^l} \gamma_t^l(j)} \tag{3.11c}$$

Para su implementación práctica es conveniente el uso de las probabilidades adelante y atrás escaladas, tomando las siguientes expresiones,

$$\hat{\pi}_i = \frac{\sum_{l=1}^{S} \frac{1}{P_l} \alpha_1^l(i) \beta_1^l(i)}{\sum_{l=1}^{S} \frac{1}{P_l} \sum_{j=1}^{N} \alpha_1^l(j) \beta_1^l(j)} = \frac{\sum_{l=1}^{S} \tilde{\alpha}_1^l(i) \tilde{\beta}_1^l(i) / c_1^l}{\sum_{l=1}^{S} \sum_{j=1}^{N} \tilde{\alpha}_1^l(j) \tilde{\beta}_1^l(j) / c_1^l} \tag{3.12a}$$

$$\hat{a}_{ij} = \frac{\sum_{l=1}^{S} \sum_{t=1}^{T^l-1} \tilde{\alpha}_t^l(i) a_{ij} b_j(o_{t+1}^l) \tilde{\beta}_{t+1}^l(j)}{\sum_{l=1}^{S} \sum_{t=1}^{T^l-1} \tilde{\alpha}_t^l(i) \tilde{\beta}_t^l(i) / c_t^l} \tag{3.12b}$$

$$\hat{b}_j(k) = \frac{\sum_{l=1}^{S} \sum_{t=1}^{T^l} \tilde{\alpha}_t^l(j) \tilde{\beta}_{t+1}^l(j) \delta_{o_t^l, v_k} / c_t^l}{\sum_{l=1}^{S} \sum_{t=1}^{T^l} \tilde{\alpha}_t^l(j) \tilde{\beta}_t^l(j) / c_t^l} \tag{3.12c}$$

3.3 Estimación de Máxima Información Mutua (MMI)

Según el Teorema Fundamental de la Teoría de la Información, el número de bits promedio necesario para transmitir un cierto evento aleatorio $\mathcal{A}$, bajo un esquema de codificación óptimo, viene dado por la *entropía* de $\mathcal{A}$, definida como,

$$H(\mathcal{A}) \equiv -\sum_{A} P(A) \log P(A) \tag{3.13}$$

donde se ha notado $P(\mathcal{A} = A) = P(A)$ para simplificar notación. La entropía puede interpretarse como una medida de incertidumbre sobre $\mathcal{A}$ (cuanto mayor sea el número de bits promedio para codificar $\mathcal{A}$, mayor será la incertidumbre sobre ella).

También se puede definir la *entropía condicional* de una variable aleatoria $\mathcal{A}$ dada otra $\mathcal{B}$ como,

$$H(\mathcal{A}|\mathcal{B}) \equiv -\sum_{A,B} P(A,B) \log P(A|B) \tag{3.14}$$

que se interpreta como la incertidumbre sobre $\mathcal{A}$ una vez conocido $\mathcal{B}$. Intuitivamente, la cantidad de información que aporta $\mathcal{B}$ sobre $\mathcal{A}$ puede medirse mediante,

$$I(\mathcal{A},\mathcal{B}) = H(\mathcal{A}) - H(\mathcal{A}|\mathcal{B}) = \sum_{A,B} P(A,B) \log \frac{P(A,B)}{P(A)P(B)} \tag{3.15}$$

que es conocida como *información mutua* entre $\mathcal{A}$ y $\mathcal{B}$.

Supóngase un sistema de reconocimiento en el que $\mathcal{W}$ es la variable aleatoria sobre el texto a reconocer y $\mathcal{X}$ sobre las secuencias de información acústica. Una medida sobre la incertidumbre respecto al texto dada la información acústica viene dada por la entropía condicional,

$$H(\mathcal{W}|\mathcal{X}) = H(\mathcal{W}) - I(\mathcal{W},\mathcal{X}) \tag{3.16}$$

La entropía $H(\mathcal{W}|\mathcal{X})$ puede entenderse de la siguiente manera: un cierto locutor pretende comunicar un mensaje W a un reconocedor mediante la señal X. El locutor quiere asegurarse de que el mensaje es correctamente reconocido, para lo cual envía una secuencia adicional de bits b, de longitud $|b|$. Según el teorema fundamental, anteriormente expuesto, $|b| \geq H(\mathcal{W}|\mathcal{X})$. Para W y X determinadas, la longitud mínima de b correspondería a $-\log P(W|X)$, lo cual sería posible en

el caso de que locutor y reconocedor tuviesen conocimiento de las probabilidades condicionales. Al no ser ésto posible, se debe recurrir al uso de un modelo de parámetros Ω para su cálculo y sustituirlas por $P_\Omega(W|X)$. En este caso, el número promedio de bits de b vendrá dado por la siguiente cantidad,

$$H_\Omega(\mathcal{W}|\mathcal{X}) = -\sum_{W,X} P(W,X)\log P_\Omega(W|X) \tag{3.17}$$

Es fácil comprobar, haciendo uso de la desigualdad de Gibbs, que,

$$H_\Omega(\mathcal{W}|\mathcal{X}) \geq H(\mathcal{W}|\mathcal{X}) \tag{3.18}$$

Cuanto menor sea la cantidad de bits que el locutor necesita enviar para asegurar el correcto reconocimiento de $\mathcal{X}$, menor será la incertidumbre de $\mathcal{W}$ dado $\mathcal{X}$. Por tanto, $H_\Omega(\mathcal{W}|\mathcal{X})$ es una medida sobre la incertidumbre del texto W dado X. En el caso óptimo, es decir, el de menor incertidumbre o longitud promedio de b mínima, se tiene que $H_\Omega = H$, que se producirá cuando el modelo propuesto coincida con el verdadero ($P_\Omega = P$).

Como conclusión a la discusión anterior se tiene que, dado un modelo de parámetros Ω, la menor incertidumbre respecto al texto $\mathcal{W}$ dada la evidencia acústica $\mathcal{X}$ se obtiene minimizando la cantidad,

$$H_\Omega(\mathcal{W}|\mathcal{X}) = H_\Omega(\mathcal{W}) - I_\Omega(\mathcal{W}|\mathcal{X}) \tag{3.19}$$

donde,

$$H_\Omega(\mathcal{W}) = -\sum_{W} P(W)\log P_\Omega(W) \tag{3.20}$$

$$I_\Omega(\mathcal{W}|\mathcal{X}) = \sum_{W,X} P(W,X)\log\frac{P_\Omega(W,X)}{P_\Omega(W)P_\Omega(X)} \tag{3.21}$$

Por tanto, la minimización de $H_\Omega(\mathcal{W}|\mathcal{X})$ implica la minimización de $H_\Omega(\mathcal{W})$ y la maximización de $I_\Omega(\mathcal{W}|\mathcal{X})$. Obviamente, lo primero da un criterio para la obtención del modelo de lenguaje Θ y lo segundo para el modelo acústico Λ. Como ya se indicó en el apartado de introducción, se supone que el modelo de lenguaje, más o menos preciso, es conocido, por lo que el problema queda restringido a la maximización de $I_\Omega(\mathcal{W}|\mathcal{X})$ (que se escribirá como $I_\Lambda(\mathcal{W}|\mathcal{X})$ en adelante), que es una medida promedio del número de bits que el reconocedor puede extraer sobre el texto $\mathcal{W}$ a partir de la señal $\mathcal{X}$ dado el modelo de parámetros Λ. Hay que

hacer notar que el método recibe el nombre de *Máxima Información Mutua* por la semejanza de $I_\Lambda(\mathcal{W}|\mathcal{X})$ a dicha función, aunque estrictamente hablando, no es una información mutua.

Al igual que en el caso ML, la maximización debe efectuarse sobre un conjunto de datos de entrenamiento $\mathcal{T}$. Además, las probabilidades $P(W,X) = P(W|X)P(X)$ son desconocidas, por lo que debe suponerse que todas la señales de $\mathcal{T}$ son igualmente probables a priori ($P(X)$ constante) y que $P(W|X) = 1$ si W es la clase correcta de X, por lo que, finalmente, se debe proceder a la maximización de la siguiente función,

$$I^{\mathcal{T}}(\Lambda) = \sum_{i=1}^{L} \sum_{X \in T_i} \log \frac{P_\Omega(W_i, X)}{P_\Theta(W_i) P_\Omega(X)} = \sum_{i=1}^{L} \sum_{X \in T_i} \log \frac{P_\Lambda(X|W_i)}{\sum_{l=1}^{L} P_\Lambda(X|W_l) P_\Theta(W_l)} \tag{3.22}$$

Para cada secuencia $X \in \mathcal{T}$ puede escribirse la función,

$$I_m^X(\Lambda) = \log P_m^X(\Lambda) \tag{3.23}$$

con

$$P_m^X(\Lambda) = \frac{P_\Lambda(X|W_m)}{\sum_{l=1}^{L} P_\Lambda(X|W_l) P_\Theta(W_l)} \tag{3.24}$$

donde W_m representa a la clase a la que pertenece X, por lo que,

$$I^{\mathcal{T}}(\Lambda) = \sum_{X \in \mathcal{T}} I_m^X(\Lambda) \tag{3.25}$$

La maximización de $I^{\mathcal{T}}$ corresponde, por tanto, a la maximización superpuesta de las distintas $I_m^X(\Lambda)$, y, por tanto, de $P_m^X(\Lambda)$. Observando la expresión (3.24) se aprecia que la diferencia fundamental con el método ML está en la inserción del denominador, que hace que $P_m^X(\Lambda)$ dependa, no sólo de los parámetros λ_m de la clase correcta, sino de los de las demás clases también. Es decir, la secuencia X no sólo entrena a los parámetros de su clase, sino a todo el conjunto Λ. Este entrenamiento tiende a maximizar la probabilidad $P_\Lambda(X|W_m)$ (lo mismo que el método ML), pero minimizando también las $P_\Lambda(X|W_l)$ de las clases incorrectas, tendiendo, por tanto, a disminuir el error.

La discusión anterior apunta a la superioridad del método MMI sobre el ML.

Otro argumento en favor del método MMI, frente al ML, es que el funcionamiento óptimo de este último se apoya sobre una serie de suposiciones que, como ya fue analizado, son básicamente falsas. En concreto, Nadas et al [Nadas88] han demostrado que el método MMI es más robusto que el ML cuando el modelo es incorrecto.

La maximización de $I^{\mathcal{T}}$ puede abordarse mediante un *descenso en gradiente* [Duda73], de forma que se busca un máximo local de forma iterativa haciendo uso de la expresión,

$$\Lambda_{n+1} = \Lambda_n + \eta \bigtriangledown_{\Lambda_n} I^{\mathcal{T}}(\Lambda_n) \tag{3.26}$$

donde n representa a la iteración en curso, y η es un factor de escala, *parámetro de convergencia*, que controla la convergencia. En general, el valor de η depende de la iteración en curso, aunque en la práctica el máximo puede alcanzarse más rápidamente eligiendo un η constante más pequeño de lo necesario [Duda73]. También es posible abordar la maximización mediante la generalización del algoritmo de Baum para funciones racionales, como es el caso de la función dada en 3.24, propuesto por Gopalakrishnan et al [Gopalakrishnan89].

3.3.1 Estimación MMI en modelos HMM discretos

Al igual que se hizo en el caso ML, se considerará en principio una sóla secuencia $O = o_1 o_2 \cdots o_T$, perteneciente a la clase W_m, para entrenar un conjunto de modelos HMM discretos $\Lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_L\}$. Para simplificar, se empleará la notación,

$$P_\Lambda(O|W_s) = P(O|\lambda_s) \tag{3.27}$$

y se supondrá que todas las clases W_s son igualmente probables a priori (se prescinde de modelo de lenguaje). La función a maximizar es,

$$I_m(\Lambda) = \log P_m(\Lambda) \tag{3.28}$$

con,

$$P_m(\Lambda) = \frac{P(O|\lambda_m)}{\sum_{l=1}^{L} P(O|\lambda_l)} \tag{3.29}$$

Obsérvese que maximizando I_m también se maximiza P_m, es decir, hacer esta probabilidad tan próxima a la unidad como sea posible, por lo que se está actuando

para disminuir el error del sistema. La maximización se lleva a cabo por el método del gradiente anteriormente expuesto. El problema se reduce, por tanto, al cálculo de las derivadas parciales de $I_m(\Lambda)$ respecto a cada uno de los parámetros de Λ. Por ejemplo, para la probabilidad inicial π_i^s del modelos λ_s se tiene,

$$\frac{\partial I_m}{\partial \pi_i^s} = \frac{\partial I_m}{\partial P(O|\lambda_s)} \frac{\partial P(O|\lambda_s)}{\partial \pi_i^s} \tag{3.30}$$

Es fácil comprobar que,

$$\frac{\partial I_m}{\partial P(O|\lambda_s)} = \frac{\delta_{m,s} - P_m}{P(O|\lambda_m)} \tag{3.31}$$

Por tanto,

$$\frac{\partial I_m}{\partial \pi_i^s} = \frac{\delta_{m,s} - P_m}{P(O|\lambda_m)} \beta_1^s(i) b_i^s(o_1) \tag{3.32}$$

Análogamente, para las matrices para las matrices $A_s = \left\{a_{ij}^s\right\}$ y $B_s = \left\{b_j^s(v_k)\right\}$,

$$\frac{\partial I_m}{\partial a_{ij}^s} = \frac{\delta_{m,s} - P_m}{P(O|\lambda_m)} \sum_{t=1}^{T-1} \alpha_t^s(i) b_j^s(o_{t+1}) \beta_{t+1}^s(j) \tag{3.33}$$

$$\frac{\partial I_m}{\partial b_j^s(v_k)} = \frac{\delta_{m,s} - P_m}{P(O|\lambda_m)} \frac{1}{b_j^s(v_k)} \sum_{t=1}^{T} \alpha_t^s(j) \beta_t^s(j) \delta_{o_t, v_k} \tag{3.34}$$

Si se usan probabilidades escaladas queda el siguiente conjunto de derivadas,

$$\frac{\partial I_m}{\partial \pi_i^s} = (\delta_{m,s} - P_s) \tilde{\beta}_1^s(i) b_i^s(o_1) \tag{3.35a}$$

$$\frac{\partial I_m}{\partial a_{ij}^s} = (\delta_{m,s} - P_s) \sum_{t=1}^{T-1} \tilde{\alpha}_t^s(i) b_j^s(o_{t+1}) \tilde{\beta}_{t+1}^s(j) \tag{3.35b}$$

$$\frac{\partial I_m}{\partial b_j^s(v_k)} = (\delta_{m,s} - P_s) \frac{1}{b_j^s(v_k)} \sum_{t=1}^{T} \tilde{\alpha}_t^s(j) \tilde{\beta}_t^s(j) \delta_{o_t, v_k}\ c_t^s \tag{3.35c}$$

donde se ha notado,

$$P_s(\Lambda) = \frac{P(O|\lambda_s)}{\sum_{l=1}^{L} P(O|\lambda_l)} \tag{3.36}$$

En el caso de un conjunto de secuencias de entrenamiento $\mathcal{T}$, teniendo en cuenta la expresión (3.25), se tendrá que para un parámetro $c \in \Lambda$,la expresión del gradiente será,

$$\frac{\partial I^{\mathcal{T}}}{\partial c} = \sum_{O \in \mathcal{T}} \frac{\partial I_m^O}{\partial c} \tag{3.37}$$

3.4 Estimación para Error de Clasificación Mínimo

Este método ha sido recientemente propuesto por Juang et al [Juang92], y aplicado con éxito en reconocimiento basado en DTW [Chang93] y modelos HMM [Chou92]. Se fundamenta en el uso de una serie de *funciones discriminantes* $g_i(X, \Lambda)$ ($i = 1, \ldots, L$), en lugar de requerir un conocimiento explícito de las probabilidades a posteriori, usando la siguiente regla de decisión,

$$W(X) = W_i \quad \text{si} \quad g_i(X, \Lambda) = \max_j g_j(X, \Lambda) \tag{3.38}$$

La forma de obtener el conjunto de parámetros Λ, una vez seleccionada la forma de las funciones discriminantes, es integrar dichas funciones en alguna *función criterio* escalar apropiada para un proceso de optimización (en el sentido de error mínimo). Ejemplos de estas funciones son el *criterio del Perceptrón*, el *criterio de la distancia cuadrática selectiva* o el *criterio del error cuadrático mínimo* [Duda73]. El problema de estos criterios es su difícil aplicación a problemas reales (caso de los dos primeros), o no conducir necesariamente a esquemas de error mínimo (caso del último).

Como alternativa, Juang propone un método estructurado en los siguientes tres pasos:

1) Definir las funciones discriminantes $g_k(X, \Lambda)$ ($k = 1, \ldots, L$).

2) Definir una *medida de error* $d_k(X)$ que sea función de las funciones discriminantes, y tal que presente valores altos para una clasificación incorrecta del objeto X, y bajos en caso contrario.

3) Construir una *función de coste* $l_k(d_k)$ tal que si X es de la clase W_k,

$$l_k(d_k(X)) = l_k(X, \Lambda) \rightarrow \begin{cases} 0 & \text{X correctamente clasificado} \\ 1 & \text{X incorrectamente clasificado} \end{cases} \tag{3.39}$$

Por tanto, la función $l_k(d_k)$ efectúa una "cuenta" sobre el error cometido en la clasificación de X.

Supuestas conocidas las funciones discriminantes, el principal objetivo es el de obtener unas funciones d_k y l_k diferenciables para poder abordar el problema de la optimización en función de Λ. Precisamente, el principal inconveniente derivado del uso del error empírico como criterio de optimización sería la no diferenciabilidad del mismo, dado su carácter intrínsecamente discreto. Juang hace las siguientes propuestas:

a) Medida de error:

$$d_k(X) = -g_k(X,\Lambda) + \left[\frac{1}{L-1}\sum_{j\neq k} g_j(X,\Lambda)^{\beta}\right]^{\frac{1}{\beta}} \tag{3.40}$$

Obsérvese que en el caso $\beta \to \infty$,

$$d_k(X) = -g_k(X,\Lambda) + g_i(X,\Lambda) \tag{3.41}$$

donde W_i es la clase con mayor $g_j(X,\Lambda)$ ($j \neq k$), y se cumple que, si X es de la clase W_k,

$$d_k(X) = \begin{cases} > 0 & \text{X incorrectamente clasificado} \\ \leq 0 & \text{X correctamente clasificado} \end{cases} \tag{3.42}$$

Obsérvese que, por tanto, β controla la aportación de las clases clases incorrectas a d_k.

b) Función de coste: la función sigmoide parece apropiada para generar una "cuenta" de error,

$$l_k(d_k) = \frac{1}{1+e^{-\alpha d_k}} \tag{3.43}$$

Obsérvese que,

$$l_k(d_k) = \begin{cases} 0 & d_k \ll 0 \\ 1 & d_k \gg 0 \end{cases} \tag{3.44}$$

El factor α controla la rapidez de la transición error-correcto (1-0).

Dada la secuencia X, el error producido por el sistema de parámetros Λ al

clasificarla debe ser expresado como,

$$l(X,\Lambda) = \sum_{k=1}^{L} l_k(X,\Lambda)\delta(X \in W_k) \tag{3.45}$$

donde se ha notado,

$$\delta(x) = \begin{cases} 1 & x \text{ verdadero} \\ 0 & x \text{ falso} \end{cases} \tag{3.46}$$

Se plantea la definición de una función criterio (basada en las funciones l_k) para llevar a cabo una minimización basada en técnicas del gradiente, teniendo en cuenta que se dispone únicamente de un conjunto de entrenamiento $\mathcal{T} = \{X_1, \ldots, X_{\mathcal{S}}\}$ en el que se conoce la clase a la que pertenece cada uno de sus elementos. Pueden seguirse dos criterios:

a) Coste empírico promedio:

$$L_0(\Lambda) = \frac{1}{\mathcal{S}} \sum_{i=1}^{\mathcal{S}} \sum_{k=1}^{L} l_k(X_i,\Lambda)\delta(X_i \in W_k) \tag{3.47}$$

b) Coste esperado:

$$L(\Lambda) = E\left[l_k(X,\Lambda)\right] = \sum_{k=1}^{L} P(W_k) \int_{\mathcal{X}} l_k(X,\Lambda)\delta(X \in W_k) P(X|W_k) dX \tag{3.48}$$

que es una aproximación a la probabilidad de error del sistema (ver apéndice C.1).

En el primer criterio es posible realizar una minimización mediante descenso en gradiente,

$$\Lambda_{n+1} = \Lambda_n + \eta \bigtriangledown_{\Lambda_n} L_0(\Lambda_n) \tag{3.49}$$

En el segundo, la minimización del error no es específica para el conjunto de muestras de entrenamiento $\mathcal{T}$, sino que se refiere al error verdadero esperado. Esta minimización no tiene una solución directa dado que las probabilidades condicionales y a priori son desconocidas. Afortunadamente, mediante el teorema del *Descenso Probabilístico Generalizado* (Generalized Probabilistic Descent, GPD) (ver apéndice C.2) es posible ajustar el conjunto de parámetros Λ cada vez que un objeto $X \in \mathcal{T}$ es presentado según el coste en el que haya incurrido dicho objeto,

haciendo $L(\Lambda)$ cada vez más pequeño. Así cuando se presente la n-sima muestra de $\mathcal{T}$ (perteneciente a la clase W_k) habrá que ajustar como sigue,

$$\Lambda_{n+1} = \Lambda_n + \delta\Lambda_n(X_n, W_k, \Lambda_n) \tag{3.50}$$

El teorema es el siguiente [Juang92]: dado un objeto X de la clase W_k, si

$$\delta\Lambda(X, W_k, \Lambda) = -\eta \mathbf{U} \bigtriangledown l_k(X, \Lambda) \tag{3.51}$$

entonces,

$$E\left[\delta L(\Lambda)\right] \leq 0 \tag{3.52}$$

donde **U** es una matriz de elementos positivos, y η es un parámetro de convergencia real positivo pequeño.

Si en la obtención de la secuencia $\{\Lambda_n\}$ (con una secuencia de entrenamiento infinita) se emplea una serie $\{\eta_n\}$ que verifique,

$$\sum_{n=1}^{\infty} \eta_n \longrightarrow \infty \tag{3.53a}$$

$$\sum_{n=1}^{\infty} \eta_n^2 < \infty \tag{3.53b}$$

entonces la secuencia $\{\Lambda_n\}$ obtenida mediante la ecuación (3.50) converge a un mínimo local de $L(\Lambda)$ [Duda73].

3.4.1 Estimación MCE en modelos HMM discretos

Lo mismo que en el caso MMI, se considera una secuencia de entrenamiento $O = o_1 o_2 \cdots o_T$ perteneciente a la clase W_m. Lo primero es establecer el esquema de minimización del error mediante los tres pasos básicos. Chou [Chou92] propone el siguiente esquema:

1) Funciones discriminantes:

$$g_k(O, \Lambda) = g_k(O, \lambda_k) = \log P(O|\lambda_k) \tag{3.54}$$

2) Medidas de error:

$$d_k(O) = -g_k(O, \lambda_k) + \log \left[\frac{1}{L-1} \sum_{j \neq k} e^{\beta g_j(O, \lambda_j)} \right]^{1/\beta} \tag{3.55}$$

3) Función de coste tipo sigmoide. Se considera $\alpha = 1$ para simplificar cálculo.

Tanto si se actúa sobre el error empírico como sobre el coste esperado, se debe evaluar el gradiente de $l_m(O, \Lambda)$. Por ejemplo, para las probabilidades iniciales π_i^s del modelos λ_s,

$$\frac{\partial l_m(O, \Lambda)}{\partial \pi_i^s} = \frac{\partial l_m}{\partial d_m} \frac{\partial d_m}{\partial \pi_i^s} \tag{3.56}$$

La primera derivada parcial del segundo miembro se calcula como,

$$\frac{\partial l_m}{\partial d_m} = l_m(O, \Lambda) \left[1 - l_m(O, \Lambda) \right] \tag{3.57}$$

Para la segunda derivada hay que distinguir si el parámetro respecto al que se deriva pertenece al modelo correcto o no,

$$\frac{\partial d_m}{\partial \pi_i^m} = -\frac{\partial g_m}{\partial \pi_i^m} = -\frac{\partial g_m}{\partial P(O|\lambda_m)} \frac{\partial P(O|\lambda_m)}{\partial \pi_i^m} = -\frac{1}{P(O|\lambda_m)} \beta_1^m(i) b_i^m(o_1) \tag{3.58}$$

$$\frac{\partial d_m}{\partial \pi_i^s} = \frac{e^{\beta g_s(O,\Lambda)}}{\sum_{l \neq m} e^{\beta g_l(O,\Lambda)}} \frac{\partial g_s}{\partial \pi_i^s} = \frac{e^{\beta g_s(O,\Lambda)}}{\sum_{l \neq m} e^{\beta g_l(O,\Lambda)}} \frac{1}{P(O|\lambda_s)} \beta_1^s(i) b_i^s(o_1) \qquad s \neq m \tag{3.59}$$

De forma análoga se procede con el resto de parámetros HMM. Si se definen los factores,

$$\nu_m(O, \Lambda) = l_m(O, \Lambda) \left[1 - l_m(O, \Lambda) \right] \tag{3.60}$$

$$\phi_{sm}(O, \Lambda) = \frac{e^{\beta g_s(O,\Lambda)}}{\sum_{l \neq m} e^{\beta g_l(O,\Lambda)}} \tag{3.61}$$

es posible escribir el siguiente conjunto de derivadas (usando probabilidades adelante y atrás escaladas),

$$\frac{\partial l_m(O, \Lambda)}{\partial \pi_i^s} = F(O, \Lambda) \tilde{\beta}_1^s(i) b_i^s(o_1) \tag{3.62a}$$

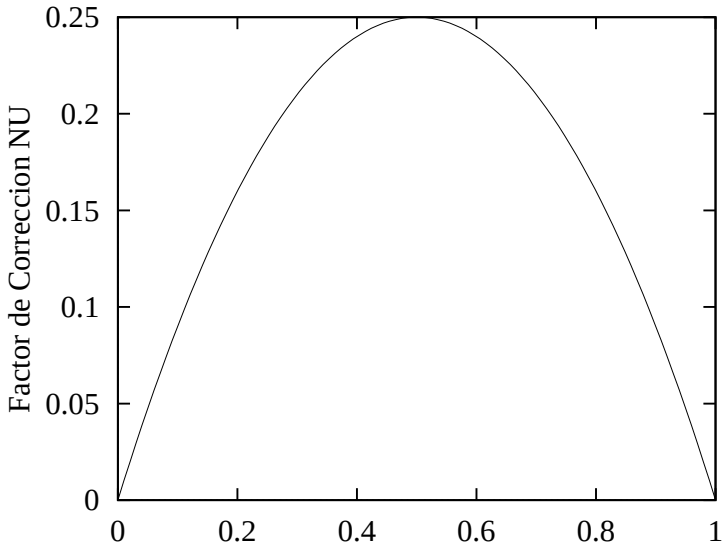


Figura 3.1: *Variación del factor ν_m con el coste l_m.*

$$\frac{\partial l_m(O,\Lambda)}{\partial a_{ij}^s} = F(O,\Lambda)\sum_{t=1}^{T-1}\tilde{\alpha}_t^s(i)b_j^s(o_{t+1})\tilde{\beta}_{t+1}^s(j) \tag{3.62b}$$

$$\frac{\partial l_m(O,\Lambda)}{\partial b_j^s(v_k)} = F(O,\Lambda)\frac{1}{b_j^s(v_k)}\sum_{t=1}^{T}\tilde{\alpha}_t^s(j)\tilde{\beta}_t^s(j)\delta_{o_t,v_k}/c_t^s \tag{3.62c}$$

donde

$$F(O,\Lambda) = \begin{cases} -\nu_m(O,\Lambda) & s = m \\ \nu_m(O,\Lambda)\phi_{sm}(O,\Lambda) & s \neq m \end{cases} \tag{3.63}$$

Es importante observar la similitud entre la fórmulas (3.35) del método MMI y las obtenidas en (3.62). Ambas fórmulas son idénticas salvo en los factores de corrección que se aplican. En el caso MMI dicho factor es $(\delta_{m,s} - P_s)$ que aumenta el peso de la corrección conforme aumenta la posibilidad de error. Ahora el factor de corrección es $F(O,\Lambda)$ calculado a partir de ν_m y ϕ_{sm}. En la figura 3.1 se representa al factor de corrección ν_m en función del coste l_m (que toma valores en el rango [0,1]). Se observa que este factor da más peso a las secuencias para las que l_m está próximo a 0.5. Estas secuencias serán denominadas en lo sucesivo como *casi-errores*, es decir, secuencias de la clase correcta W_m que pueden ser

reconocidas fácilmente como perteneciente a otra clase incorrecta W_s ($s \neq m$), o secuencias de una clase incorrecta (W_s ($s \neq m$) que pueden reconocerse fácilmente como pertenecientes a W_m. Así, ν_m da más importancia a los casi-errores que a no-errores claros ($l_m = 0$) o errores claros ($l_m = 1$). El factor ϕ_{sm} se activa en el caso de que el modelo λ_s, aún siendo incorrecto, sea claramente superior a los otros incorrectos.

3.5 Estimaciones iniciales y entrenamiento insuficiente en HMMs

Todos los métodos de estimación descritos en este capítulo trabajan iterativamente, por lo que es necesario que estos procedimientos arranquen de alguna estimación inicial. Además, dado que estos métodos conducen sólo a extremos locales, una buena selección de las estimaciones iniciales puede ser crítica, a la vez que aligerar la convergencia. La experiencia demuestra que una selección uniforme o incluso aleatoria de las probabilidades Π y A es aceptable. No ocurre así con las probabilidades B, para las que una buena estimación inicial puede ser importante si se usan modelos discretos, y esencial para los contínuos [Rabiner89a]. Una *segmentación manual* [Peinado89] o *lineal* [Peinado91] (dividir cada secuencia en segmentos de igual longitud) puede ser aceptable. En el caso de modelos contínuos, puede emplearse el procedimiento *segmental k-means*, que aplica una segmentación por Viterbi en cada iteración, lo cual permite la reestimación de las densidades de producción [Rabiner86a, Rabiner89a]. Peinado et al [Peinado91b] han demostrado que también es posible el uso de *medidas de cohesión entrópicas* para la obtención de segmentaciones y modelos iniciales.

Otro problema inherente a la reestimación de parámetros HMM es que la longitud de la secuencia de entrenamiento es, irremediablemente, finita. Esto provoca que el número de veces que se produce un cierto evento (p. ej., número de veces de un cierto símbolo en un cierto estado) no sea suficiente para proporcionar una buena estimación. Es incluso posible que alguna probabilidad se haga nula, dificultando el proceso de reestimación. Una solución puede ser reducir el número de parámetros a estimar, aunque ésto no es siempre posible. Otra posible solución es interpolar las estimaciones de los parámetros del modelo λ con otras estimaciones más robustas $\tilde{\lambda}$ de los mismos, para obtener unos nuevos modelos $\bar{\lambda}$ como,

$$\bar{\lambda} = \epsilon\lambda + (1 - \epsilon)\tilde{\lambda} \tag{3.64}$$

donde el parámetro ϵ controla la contribución de cada uno de los modelos. La técnica conocida como *deleted interpolation* [Lee89] soluciona el problema del valor que ha de asignarse a ϵ.

Otra solución más simple y efectiva [Rabiner89a] es añadir nuevas condiciones a los parámetros del modelo que impidan que las estimaciones de las probabilidades caigan por debajo de un cierto umbral. Por ejemplo, en el caso de modelos discretos es posible realizar, en cada iteración de la reestimación, las siguientes asignaciones,

$$a_{ij} = \delta \;\; si \;\; a_{ij} < \delta \tag{3.65a}$$

$$b_j(v_k) = \epsilon \;\; si \;\; b_j(v_k) < \epsilon \tag{3.65b}$$

donde δ y ϵ son umbrales experimentales de valores pequeños respecto a los valores típicos de dichas probabilidades. Tras realizar esta asignación, es necesario recalcular todos los parámetros del modelo para mantener las condiciones de normalización de las probabilidades [Peinado89].

Capítulo 4

LA BASE DE DATOS

La intención de este capítulo es realizar una descripción de los datos empíricos utilizados en el desarrollo de los sistemas de RAH que se describen en los siguientes capítulos, así como del uso que de ellos se hace. Se comienza por una descripción de la tarea y selección del vocabulario. Posteriormente se analiza la distribución de los datos para la construcción y test de sistemas. Finalmente, se describe el proceso de adquisición de esos datos.

4.1 Descripción y Vocabulario

La construcción y test de un sistema de RAH requiere tener a disposición un amplio conjunto de señales de voz (base de datos) que permitan la consecución de un doble objetivo:

1) Obtener un sistema suficientemente entrenado, es decir, capaz de asimilar la variabilidad intrínseca de la señal de voz.

2) Poder realizar un test estadísticamente significativo para obtener valores del error de reconocimiento del sistema.

La tarea escogida, tal como se argumentó en el capítulo de introducción, corresponde a palabras aisladas. La distribución de probabilidades a priori de las palabras se supone equiprobable, lo cual suprime la necesidad de caracterización del lenguaje. El problema se reduce, pues, a la definición de un vocabulario apropiado y la obtención de un número suficiente de señales correspondientes a cada una de las palabras que lo componen. Concretamente, la tarea de sistema consiste en la implementación de un control oral sobre un *Robot Teach-200*, articulado por 6 motores de corriente continua. Para ello, se diseñó un vocabulario compuesto por 6 palabras de control, una para cada uno de los motores [Peinado89], al que posteriormente se añadieron los 10 dígitos castellanos (ver tabla 4.1) para versatilizar el control sobre los motores. Queda, por tanto, un vocabulario compuesto por 16 palabras, 10 de las cuales (los dígitos) presentan un grado de confusión importante entre sí, debido a su corta duración y a la similitud de sus contenidos acústicos.

4.2 Composición y Organización de los datos

Como ya se ha mencionado, será vital en la construcción del sistema el disponer de una gama lo más amplia posible de datos correspondientes a distintas situaciones, para garantizar la correcta acción del sistema. Así, se han seleccionado 40 locutores, 20 masculinos y otros 20 femeninos, con edades comprendidas entre 18 y 70 años. Este último aspecto, así como los de procedencia y y nivel socio-cultural, ha estado muy condicionado a la disponibilidad de locutores en el ámbito de la comunidad universitaria, profesores y alumnos, de Granada. El acento predominante es, pues, el andaluz, aunque, esporádicamente, pueden aparecer de otras procedencias.

De cada uno de estos locutores, sin entrenamiento previo, se recogieron 3 repeticiones de cada palabra. Ésto proporciona un total de 120 repeticiones distintas

DÍGITOS	PALABRAS CLAVE
CERO	CUERPO
UNO	HOMBRO
DOS	CODO
TRES	MUÑECA
CUATRO	MANO
CINCO	DEDOS
SEIS	
SIETE	
OCHO	
NUEVE	

Tabla 4.1: *Composición del Vocabulario.*

de cada palabra del vocabulario, y un total de 1920 señales para toda la base de datos.

El conjunto de datos debe ser dividido en dos; una parte será dedicada a entrenamiento y la otra a test para la obtención del error del sistema. Un grave dilema surge a la hora de realizar esta partición: por un lado sería deseable un subconjunto de entrenamiento lo más amplio posible para obtener un sistema bien entrenado, pero, por otro, también es necesario reservar un número suficientemente grande de datos para test, con el fin de obtener valores significativos del error del sistema. Por supuesto, la partición en dos del conjunto de datos no es la única solución. Es posible hacer distintas divisiones y realizar un promediado de los valores de error de cada una de ellas, para dar un error final. En un caso extremo de carencia de datos, es posible usar todos los datos para entrenamiento, salvo uno que es usado para test, y realizar el procedimiento tantas veces como datos haya, lo cual es conocido como técnica *leaving-one-out* [Duda73]. Este tipo de técnicas permiten que toda la base de datos sea usada para test.

En las experiencias realizadas en este trabajo, se han realizado dos tipos de partición según se trate de sistemas Multilocutor o Independientes del Locutor:

1) Para sistemas **Multilocutor** todos los locutores están presentes entrenamiento y test, dejándose 2 de las 3 repeticiones de cada palabra para entrenamiento y la restante para test. Se realizaron 3 particiones distintas: MULTI_0, MULTI_1 y MULTI_2, donde el último dígito hace referencia al

número de repetición reservada para test. Así, se dispone, en cada partición, de 80 repeticiones por palabra para entrenamiento (1380 en total), y 40 para test (640 en total). En lo sucesivo, se denominará experiencia *MULTI* a la resultante del promedio de la experiencias realizadas con cada una de las 3 particiones (obtención de la tasa de error promedio).

2) Para sistemas **Independientes del Locutor** los locutores empleados en entrenamiento deben ser distintos de los de test (ahora cada locutor aporta sus 3 repeticiones de cada palabra). Se han realizado 5 particiones distintas, en cada una de las cuales se reservan 32 locutores para entrenamiento, y los 8 restantes para test. Así, cada partición posee 96 repeticiones por palabra para entrenamiento (1536 en total), y 24 para test (384 en total). Estas particiones son: LOCI_0, LOCI_1, LOCI_2, LOCI_3 y LOCI_4. El último dígito hace referencia al grupo de locutores reservados para test: 0-7 (grupo 0), 8-15 (grupo 1), 16-23 (grupo 2), 24-31 (grupo 3) y 32-39 (grupo 4). Cada subconjunto de entrenamiento y test contiene igual número de locutores femeninos que masculinos. En lo sucesivo, se denominará experiencia *LOCI* a la resultante del promedio de la experiencias realizadas con cada una de las 5 particiones (obtención de la tasa de error promedio).

4.3 Adquisición de los datos

Para la obtención del conjunto de señales antes descrito, nuestro grupo de investigación desarrolló un sistema de adquisición sobre un microordenador IBM-PC. El control del sistema lo realiza un programa PASCAL denominado ADAMENU, que gestiona una tarjeta ADA PC-14 de conversión A/D/A de 14 bits (rango 0-16386) con dos canales de entrada y otro de salida. Para la representación digital de las señales se emplearon 12 bits (rango 0-4095), correspondientes a un rango dinámico de entrada al conversor A/D de 0-4 Volts. Dado que la señal de voz posee media nula, se añadió un circuito sumador para remontar las señales 2 Volts. Posteriormente, ADAMENU resta a la señal su valor medio, con lo que queda finalmente representada en el rango [-2047,+2048].

La frecuencia de muestreo seleccionada fue de 8 kHz, por lo que se añadieron a la entrada del sistema dos filtros analógicos Butterworth: uno antialiasing paso-baja de orden 8 y con una frecuencia de corte de 3.8 kHz, y otro paso-alta de orden 6 con frecuencia de corte en 60 Hz, para eliminar posibles ruidos de baja frecuencia

(como la red de suministro eléctrico).

El programa ADAMENU incorpora, además, utilidades para almacenamiento y recuperación de señales en disco, visualización, detección automática de posibles saturaciones y segmentación automática de la señal para eliminar los silencios inicial y final. Para esto último se utilizó el algoritmo de delimitación propuesto por Rabiner y Sambur [Rabiner75], y modificado sucesivamente por nuestro grupo en los trabajos [Segura84] y [Peinado89]. El algoritmo divide la señal en segmentos de 128 muestras de longitud solapados 64 muestras entre sí. Para la caracterización del silencio se requiere que, al menos, los doce primeros segmentos de señal sean de silencio. También se calcula el máximo absoluto de energía en la señal. A partir de este proceso de caracterización, se obtienen unos umbrales superior e inferior de energía (UESUP y UEINF) y otro de cruces por cero (UZS). A continuación, se trazan la curvas de energía y cruces por cero y se procede de la siguiente manera, para encontrar el primer límite:

1) Hay que situarse en el primer segmento, y avanzar hasta encontrar un segmento que supere el umbral UESUP.

2) Desde este segmento, se retrocede hasta encontrar otro que no supere el umbral inferior UEINF, ni el de cruces por cero UZS.

3) Desde éste último, se comprueba si en un número de segmentos prefijado hacia atrás no se supera UEINF. En caso contrario se vuelve al paso 2.

4) A partir del segmento actual se comprueba si en un cierto número de segmentos prefijado hacia atrás no se supera UZS, en cuyo caso se decide que ese es el primer segmento de no silencio. En caso contrario se vuelve al paso 2.

El proceso es descrito gráficamente en la figura 4.1. Un proceso análogo debe seguirse para el silencio final.

La grabación de las señales se realizó en un ambiente de laboratorio, incluyendo el ruido inevitable del microordenador empleado en la adquisición, con un micrófono de alta fidelidad. La relación señal-ruido promedio resultante para las 1920 señales adquiridas fue de 23.95 dB.

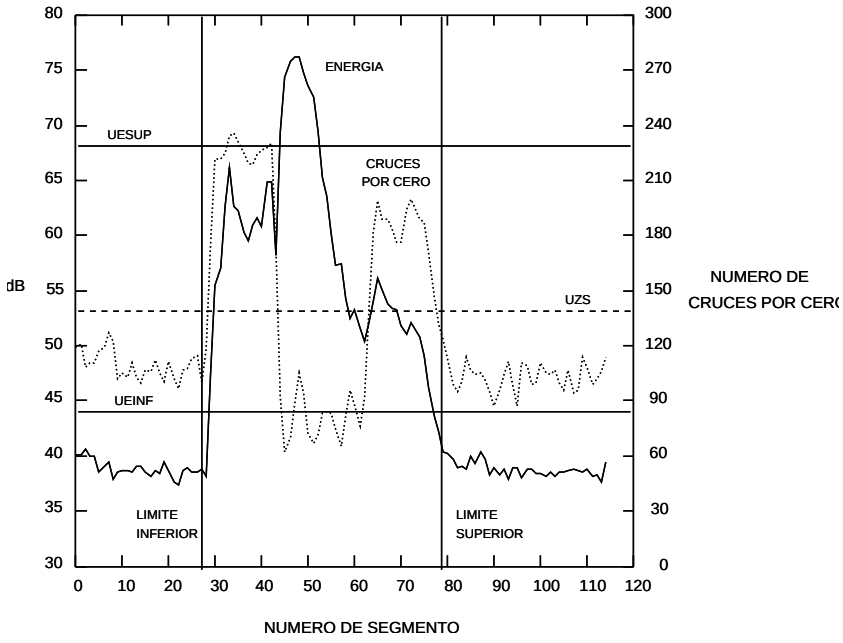


Figura 4.1: *Delimitación de la palabra /SEIS/.*

Capítulo 5

EL SISTEMA DE REFERENCIA

En este capítulo se establecerá un sistema de reconocimiento básico basado en modelos discretos de Markov que servirá de referencia para los sistemas y técnicas que se irán introduciendo en capítulos posteriores. Primeramente, se realizará un estudio sobre la distancia de procesado de señal que se va a usar, y sobre cómo modificar dicha distancia cuando se considera un grupo heterogéneo de características. Al vector formado por el cepstrum y su derivada, se incorporará información sobre la energía de cada trama. Una vez establecido el método de vectorización y cuantización de la señal de voz, se compararán las estimaciones ML y MMI de los modelos. También se procederá a la ampliación del sistema diseñado al caso semicontinuo.

5.1 Preliminares

La selección de técnicas y parámetros en los próximos apartados y capítulos se realizará en función a las mejoras en las tasas de error. Para desarrollar este trabajo se deben tomar algunas de las decisiones básicas, referentes tanto al proceso de análisis o preprocesamiento, como a los modelos HMM, para la construcción de un sistema de partida.

5.1.1 Preprocesamiento

Tras ser preenfatizada (con factor $\mu = 0.95$), la señal es segmentada en tramas (usando ventanas de Hamming) de $32\ ms$ desplazadas $8\ ms$ entre sí. El proceso de parametrización y vectorización, abordado en los próximos apartados, puede ser entonces aplicado a cada una de las tramas, obteniendo un conjunto de parámetros o vector de características para cada una de ellas. Con el objetivo de no aumentar excesivamente la cantidad de cálculo implicada en el entrenamiento y test de los sistemas, los parámetros de cada dos tramas consecutivas son promediados, usando estos promedios en lugar de los parámetros originales. El resultado es como si se hubiese segmentado en tramas de $32\ ms$ desplazadas $16\ ms$.

5.1.2 Modelos HMM y VQ

Tal como se describió en el capítulo anterior la tarea considerada es de palabras aisladas. Cada palabra del vocabulario es representada por un modelo tipo DHMM con una topología como la que se muestra en la figura 5.1, con $N = 10$ estados, un factor de salto $\Delta = 1$ (número máximo de estados que es posible saltar hacia adelante), y un diccionario con $M = 64$ centros.

Como se puede ver se incluyen dos estados nulos (que no emiten símbolo): I (estado inicial) y F (estado final). La probabilidades de transición a_{Ii} $(i = 1, \ldots, N)$ son totalmente equivalentes a las probabilidades de estados iniciales π_i $(i = 1, \ldots, N)$. Obsérvese que sólo $a_{I1} = \pi_1$ es no nula. Además, todos los estados tienen permitida la transición al estado final, con lo cual se está teniendo en cuenta la posibilidad de que cualquier secuencia concluya en cualquier estado. Los modelos son inicializados mediante segmentación lineal y entrenados con el procedimiento ML de Baum-Welch. Los problemas derivados del entrenamiento insuficiente son solventados mediante la imposición de umbrales de probabilidad.

El diccionario universal será construído haciendo uso del algoritmo LBG,

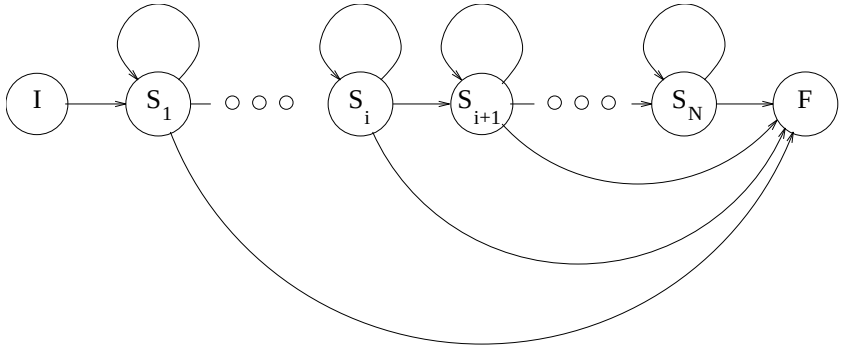


Figura 5.1: *Topología de los HMM empleados.*

inicializado por bipartición. En los próximos apartados se razonará y especificará la distancia empleada para VQ.

5.2 Uso de distancias cepstrales

En el capítulo 2 se introdujeron algunas de las distancias de procesado de señal más importantes. La distancia cepstral entre los espectros de las señales de test y referencia, H_t y H_r, venía dado por la siguiente expresión,

$$d_c(\mathbf{c}_t, \mathbf{c}_r) = \int_{-\pi}^{\pi} \left| logH_t(e^{j\omega}) - logH_r(e^{j\omega}) \right|^2 \frac{d\omega}{2\pi} = \sum_{n=-\infty}^{+\infty} \left(c_t(n) - c_r(n) \right)^2 \tag{5.1}$$

La implementación práctica de la distancia cepstral requiere la aplicación de una ventana cepstral $w(n)$ de longitud finita L en el dominio de la *cuefrencia* [Bogert63],

$$\tilde{c}(n) = c(n)w(n) \qquad\qquad (n = 1, \ldots, L) \tag{5.2}$$

que reduzca la sumatoria en la ecuación (5.1) a un número finito de términos, de forma que la nueva distancia queda expresada como,

$$d_c(\tilde{\mathbf{c}}_t, \tilde{\mathbf{c}}_r) = \sum_{n=1}^{L} \left(\tilde{c}_t(n) - \tilde{c}_r(n) \right)^2 = \left\| \tilde{\mathbf{c}}_t - \tilde{\mathbf{c}}_r \right\|^2 \tag{5.3}$$

que es una distancia euclídea en el espacio de los vectores $\tilde{\mathbf{c}}$ formados por los coeficientes de la ventana. Obsérvese que el proceso es equivalente a un filtrado pasa-banda en el dominio cuefrencial, por lo que se suele denominar *proceso de liftering*.

Tohkhura [Tohkura87] y Juang [Juang87] demostraron que el uso de la distancia cepstral, incorporando ciertas ventanas en el dominio de la cuefrencia, puede contribuir a la mejora del rendimiento de los sistemas respecto a otro tipo de distancias extendidas en RAH como la *Razón de Semejanza* [Juang82].

La ventana más inmediata es la ventana *rectangular*,

$$w(n) = \begin{cases} 1 & 1 \leq n \leq L \\ 0 & \text{en caso contrario} \end{cases} \tag{5.4}$$

Otra posible ventana es la formada por las inversas de las desviaciones estandard de los coeficientes cepstrales (pesado estadístico) [Tohkura87],

$$w(n) = \begin{cases} 1/\sigma_c(n) & 1 \leq n \leq L \\ 0 & \text{en caso contrario} \end{cases} \tag{5.5}$$

Dado que la varianza cepstral es una función con variación tipo $1/i^2$ (ver figura 5.2), esta ventana se asemeja a la ventana triangular propuesta por Paliwal [Paliwal82] (ver figura 5.3). Es fácilmente demostrable que,

$$\sum_{n=-\infty}^{+\infty} n^2 (c_t(n) - c_r(n))^2 = \int_{-\pi}^{\pi} \left| \frac{\partial log H_t(e^{j\omega})}{\partial \omega} - \frac{\partial log H_r(e^{j\omega})}{\partial \omega} \right|^2 \frac{d\omega}{2\pi} \tag{5.6}$$

por lo que la distancia basada en el pesado estadístico de los coeficientes cepstrales es, aproximadamente, una medida de distancia entre las pendientes de los cepstrum de las dos señales comparadas.

Juang et al [Juang87] estudiaron las fuentes de variabilidad del cepstrum LPC, llegando a las siguientes conclusiones:

a) La variabilidad en los primeros coeficientes cepstrales es debida principalmente a las variaciones en las condiciones de captación y transmisión de la señal de voz, características del locutor, esfuerzos vocales, etc.,

b) La variabilidad de los coeficientes cepstrales de orden más alto es inherente al proceso de análisis.

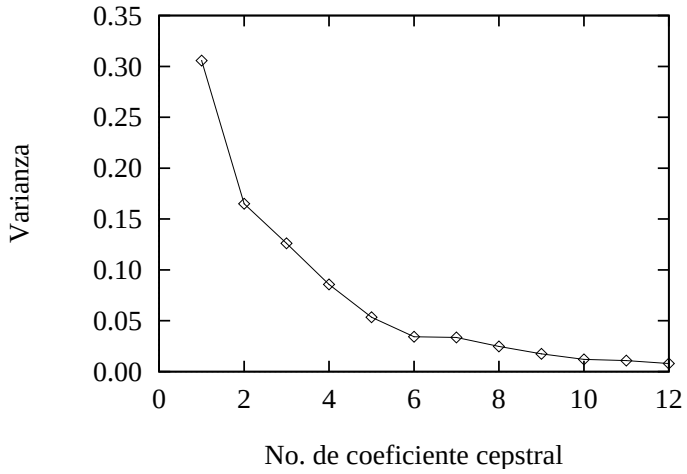


Figura 5.2: *Varianzas de los coeficientes cepstrales en función del número de coeficiente.*

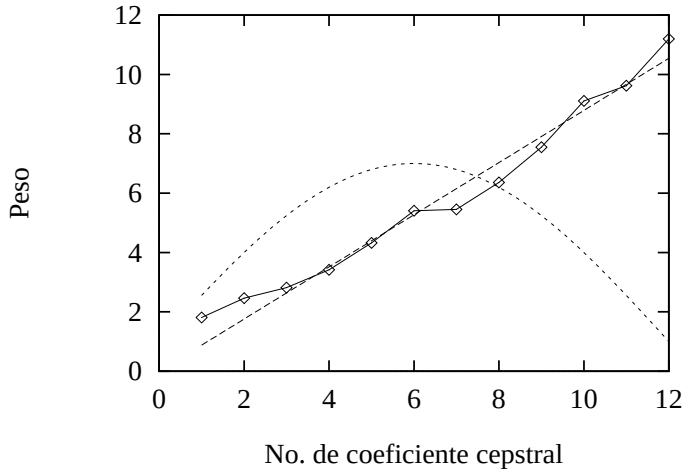


Figura 5.3: *Ventanas triangular/pesado-estadístico y seno remontado*

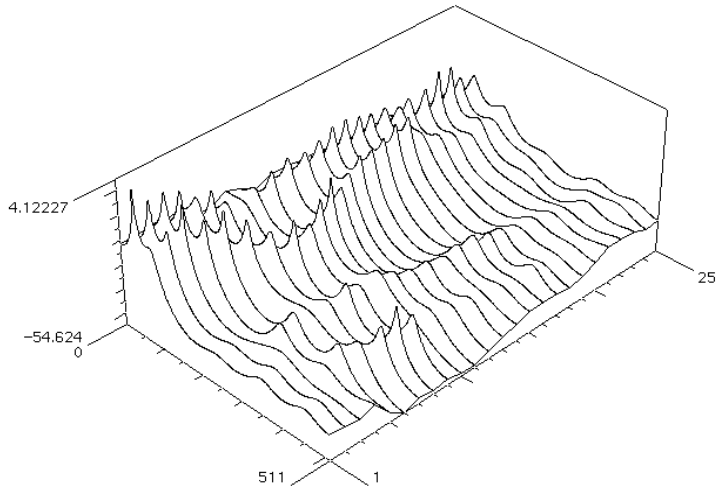
La conclusión es que la ventana cepstral debe disminuir las variaciones indeseables debidas a los coeficientes de orden más bajo y más alto. El pesado estadístico consigue evitar la variabilidad del tipo *a*, pero puede incluso potenciar la variabilidad tipo *b* cuando la longitud de la ventana es suficientemente grande. Como alternativa, Juang propone la ventana *seno remontado*, con la intención de eliminar ambos tipos de variación. Esta ventana tiene la siguiente expresión,

$$w(n) = \begin{cases} 1 + \frac{L}{2} sen\left(\frac{\pi n}{L}\right) & 1 \leq n \leq L \\ 0 & \text{en caso contrario} \end{cases} \tag{5.7}$$

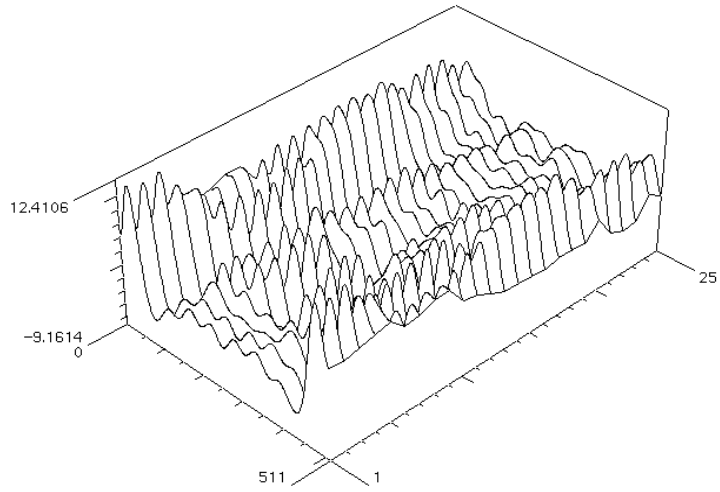
con la forma representada en la figura 5.3. Un amplio estudio sobre la aplicación de distintos tipos de ventanas en reconocimiento puede encontrarse en [Junqua89]. Este estudio demuestra que la ventana seno remontado presenta un comportamiento bastante regular en condiciones variadas de ruido ambiente, aunque existen otras ventanas que trabajan mejor bajo condiciones específicas.

Como se puede ver en la figura 5.4, el principal efecto de aplicar una ventana cepstral es el de suavizar los picos del espectro LPC de la señal, manteniendo la estructura de formantes, con lo cual se consigue una distancia menos sensible a la posición de estos picos.

En las figuras 5.5 se muestran los resultados de reconocimiento en los modos MULTI y LOCI para las 3 ventanas cepstrales anteriormente expuestas, en función de la longitud de la ventana. En el caso de la ventana rectangular, salvo la irregularidad de las gráficas para 14-16 coeficientes, la tendencia es que el error decrece con la longitud de la ventana. La gráfica con pesado estadístico presenta un comportamiento decreciente para longitudes pequeñas (hasta 8), pero se vuelve creciente o inestable para longitudes mayores, mejorando, en general, los resultados de la ventana rectangular. Los mejores resultados corresponden a la ventana seno remontado, para la que las gráficas de error presentan un mínimo alrededor de $L = 14\ 18$ para MULTI, y $L = 14$ para LOCI. En lo sucesivo, se usará la ventana seno remontado con $L = 14$, para no aumentar excesivamente el coste computacional. Los valores de error obtenidos en $L = 14$ son 10.16% (MULTI) y 13.49% (LOCI).

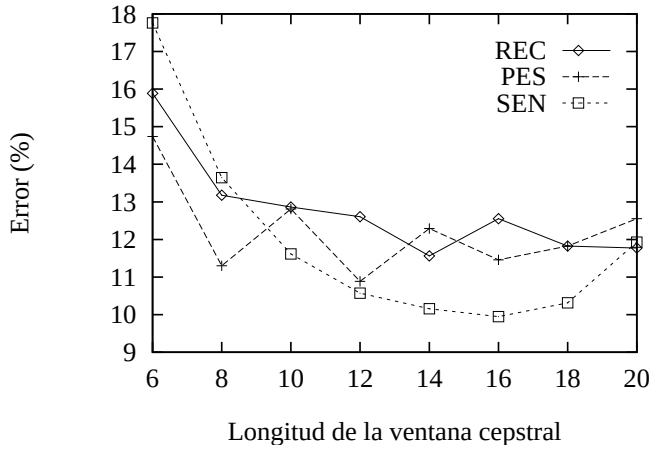


a)

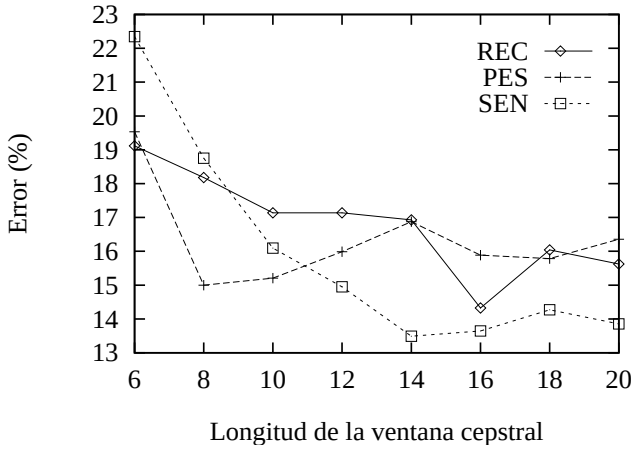


b)

Figura 5.4: *Efecto de la aplicación de ventanas en el dominio de la cuefrencia: a) Espectro LPC original, b) Espectro correspondiente a liftering con seno remontado.*



(a) *Experiencia MULTI*



(b) *Experiencia LOCI*

Figura 5.5: *Error en función de la longitud de ventana para las ventanas rectangular (REC), pesado estadístico (PES) y seno remontado (SEN).*

5.3 Nuevas características

Para enriquecer la caracterización de las tramas de análisis es posible introducir nuevos parámetros en los vectores que las representan, además de los coeficientes cepstrales tratados en el apartado anterior. Los parámetros cepstrales sólo contienen información *instantánea* de la señal de voz. Furui [Furui86] sugiere el uso de características *dinámicas*, para añadir al vector de características información *transicional* de la señal, es decir, añadir información sobre el ritmo de variación temporal de las propiedades de la señal.

La aproximación al problema realizada por Furui [Furui81] incorpora al vector cepstral estudiado anteriormente los coeficientes segundos obtenidos del desarrollo en serie mediante polinomios ortogonales de cada coeficiente cepstral c_n, considerado como una función del tiempo $c_n(t)$,

$$\Delta c_n(t) = \frac{\sum_{k=-K}^{k=K} k c_n(t+k)}{\sum_{k=-K}^{k=K} k^2} \qquad (n = 1, ..., L) \tag{5.8}$$

Como se puede observar, estos nuevos coeficientes pueden ser considerados como la pendiente de la tangente a $c_n(t)$ en el punto t, obtenida mediante el análisis de regresión lineal de los valores de c_n contenidos en un entorno temporal de centro t y radio K, y, por tanto,

$$\frac{\partial c_n(t)}{\partial t} \simeq \Delta c_n(t) \tag{5.9}$$

por lo que la señal $\{\Delta c_n(t); n = 1, \ldots, L\}$ es conocida como *Delta Cepstrum*, y sus muestras como *coeficientes delta cepstrales*. Es posible escribir la primera derivada respecto al tiempo del espectro LPC logarítmico en función de los coeficientes delta cepstrales,

$$\frac{\partial log H(\omega, t)}{\partial t} \simeq \sum_{n=-\infty}^{\infty} \Delta c_n(t) e^{jn\omega} \tag{5.10}$$

También es posible construir una distancia tipo euclídeo $d_{\Delta c}$ entre dos vectores de coeficientes delta cepstrales, con un significado análogo al de las ecuaciones (5.1) y (5.3), sustituyendo $log H(\omega)$ por $\partial log H(\omega, t)/\partial t$. Si calculamos los coeficientes delta cepstrales a partir de coeficientes cepstrales con liftering [Rabiner89], esta

distancia será de la siguiente forma,

$$d_{\Delta c}(\Delta\tilde{\mathbf{c}}_t, \Delta\tilde{\mathbf{c}}_r) = \sum_{n=1}^{L} \left(\Delta\tilde{c}_t(n) - \Delta\tilde{c}_r(n)\right)^2 = \|\Delta\tilde{\mathbf{c}}_t - \Delta\tilde{\mathbf{c}}_r\|^2 \tag{5.11}$$

Otra posibilidad para enriquecer el vector de características es la de introducir información sobre la energía de la señal. Diversos trabajos [Brown82, Rabiner84] han tratado sobre la introducción de la energía de la trama de señal, definida como,

$$E(t) = 10 \cdot log_{10}(R_t(0)) \tag{5.12}$$

donde $R_t(0)$ es el coeficiente de autocorrelación de orden cero de la señal, y t es el instante de tiempo. La ecuación (5.12) expresa la energía en escala decibélica. Este parámetro presenta el problema de ser excesivamente dependiente del nivel de grabación de la señal, por lo que se suele normalizar al pico de energía de la secuencia temporal considerada,

$$E(t) = 10 \cdot log_{10}(R_t(0)) - 10 \cdot log_{10}\left[\max_{1\leq l\leq T}\left(R_l(0)\right)\right] \tag{5.13}$$

Los trabajos mencionados encontraron que dicha energía normalizada disminuía el error en sistemas de palabras aisladas. Además, en [Rabiner84] se muestra que existe cierta correlación entre la energía de la señal y el espectro LPC, lo cual permite incorporar ambas características en un solo diccionario de cuantización, con el consiguiente ahorro en número total de centros para un sistema basado en HMMs discretos.

Otros trabajos [Furui86, Peinado90] tratan de incluir la derivada primera de la energía respecto al tiempo, conocida como *Delta Energía*, en el vector de características. La delta energía se define como,

$$\Delta E(t) = \frac{\sum_{k=-K}^{k=K} kE(t+k)}{\sum_{k=-K}^{k=K} k^2} \tag{5.14}$$

Lee [Lee89] demostró que la delta energía es más útil en reconocimiento que la propia energía.

En lo que sigue se considerará un vector de análisis formado por coeficientes

cepstrales, delta cepstrales, energía y delta energía,

$$\mathbf{x} = \{\tilde{\mathbf{c}}, \Delta\tilde{\mathbf{c}}, E, \Delta E\} \tag{5.15}$$

5.4 Inclusión de nuevas características

Una vez decididos qué parámetros caracterizarán las tramas de señal, el problema que se plantea es el de cómo nuestro sistema ha de manejar la información heterogénea contenida en los vectores de análisis. En la bibliografía se han propuesto, al menos, tres posibilidades:

a) Los primeros esfuerzos para enriquecer el vector de análisis con características dinámicas no se basaron en el uso de derivadas, como acabamos de ver en apartado anterior, sino en la concatenación de vectores adyacentes [Nadas81]. Esto aumentaba en sobremanera la dimensionalidad del problema, por lo que se requería el uso de alguna técnica de reducción del número de dimensiones de los vectores, tales como el *Análisis de Componentes Principales* o el *Análisis Discriminante* [Tou74, Fukunaga90], consistentes, básicamente, en la transformación del espacio de representación. Diversos trabajos han hecho uso de este tipo de técnicas con éxito [Bocchieri86, Lleida90]. Estas transformaciones espaciales suelen implicar una distancia distinta de la cepstral y delta cepstral anteriormente descritas, por lo que no serán consideradas en este trabajo.

b) Furui [Furui86] y trabajos posteriores [Peinado90, Rabiner89, Segura91] proponen con éxito el uso de una *Distancia Multicaracterística Pesada* (DMP), compuesta de las distancias euclídeas sobre los distintos tipos de características, pesadas entre sí de forma experimental. La forma de esta distancia, entre un vector de test $\mathbf{x}_t$ y otro de referencia $\mathbf{x}_r$, es,

$$\begin{aligned} d(\mathbf{x}_t, \mathbf{x}_r) \;=\; & p_c \frac{d_c(\tilde{\mathbf{c}}_t, \tilde{\mathbf{c}}_r)}{\sigma_c^2} + p_{\Delta c} \frac{d_{\Delta c}(\Delta\tilde{\mathbf{c}}_t, \Delta\tilde{\mathbf{c}}_r)}{\sigma_{\Delta c}^2} + \\ & p_E \frac{d_E(E_t, E_r)}{\sigma_E^2} + p_{\Delta E} \frac{d_{\Delta E}(\Delta E_t, \Delta E_r)}{\sigma_{\Delta E}^2} \end{aligned} \tag{5.16}$$

donde σ_E^2 y $\sigma_{\Delta E}^2$ son las varianzas de la energía y delta energía, respectivamente, y σ_c^2 y $\sigma_{\Delta c}^2$ son de la forma,

$$\sigma_c^2 = \sum_{n=1}^{L} \sigma_c(n)^2 \tag{5.17}$$

$$\sigma_{\Delta c}^2 = \sum_{n=1}^{L} \sigma_{\Delta c}(n)^2 \tag{5.18}$$

y actúan de factores de normalización, haciendo que las distancias d_c, $d_{\Delta c}$, d_E y $d_{\Delta E}$ tengan similares valores esperados. Los parámetros p_c, $p_{\Delta c}$, p_E y $p_{\Delta E}$ son los pesos experimentales que informan de la importancia de un determinado tipo de características. Las distancias d_c y $d_{\Delta c}$ tienen las formas dadas en las ecuaciones (5.3) y (5.11). Las distancias d_E y $d_{\Delta E}$ corresponden a,

$$d_E(E_t, E_r) = (E_t - E_r)^2 \tag{5.19}$$

$$d_{\Delta E}(\Delta E_t, \Delta E_r) = (\Delta E_t - \Delta E_r)^2 \tag{5.20}$$

c) Otra posibilidad es la de construir diccionarios independientes para cada clase de parámetros, usando en cada uno de ellos distancias euclídeas [Gupta87, Lee89]. Este método es descartado por el elevado coste computacional que introduce.

En lo sucesivo se empleará únicamente la distancia DMP dada en (5.16). A continuación se explica el ajuste de sus pesos experimentales.

5.4.1 Ajuste de los Pesos Experimentales

En esta sección, se ajustarán experimentalmente los valores del conjunto de pesos de la distancia dada por (5.16) de forma que proporcionen un error mínimo para las experiencias MULTI y LOCI.

Ajuste del peso del Delta Cepstrum

Para estudiar la importancia relativa del delta cepstrum frente al cepstrum se propone la siguiente distancia,

$$d(\mathbf{x}_t, \mathbf{x}_r) = (1-\mu) d_c^*(\tilde{\mathbf{c}}_t, \tilde{\mathbf{c}}_r) + \mu d_{\Delta c}^*(\Delta\tilde{\mathbf{c}}_t, \Delta\tilde{\mathbf{c}}_r) \tag{5.21}$$

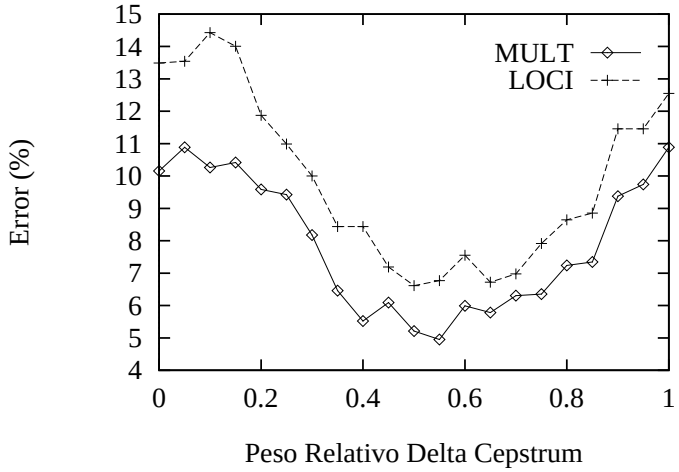


Figura 5.6: *Inclusión del delta cepstrum.*

donde se ha usado la notación $d_c^* = d_c/\sigma_c^2$ y $d_{\Delta c}^* = d_{\Delta c}/\sigma_{\Delta c}^2$, y μ representa el peso relativo del delta cepstrum, que varía entre 0 (no delta cepstrum) y 1 (no cepstrum). El intervalo temporal para el cómputo de los coeficientes delta cepstrales es de ± 3 tramas.

En la gráfica 5.6 se puede observar cómo varía el error MULTI y LOCI en función del peso relativo μ. Se tomará como valor óptimo $\mu = 0.55$, para el cual se obtienen los valores de error de 4.95% y 6.77% para los casos MULTI y LOCI, respectivamente. Esto indica que cepstrum y delta cepstrum tienen una importancia relativa similar, con una ligera ventaja para el delta cepstrum (esta conclusión es bastante similar a la encontrada por Furui en [Furui86]). Este resultado permite fijar $p_c = 1$ y $p_{\Delta c} = \mu/(1-\mu) = 1.222$.

Ajuste del peso de la Delta Energía

Siguiendo el método establecido anteriormente, se usará como medida de distancia,

$$d(\mathbf{x}_t, \mathbf{x}_r) = (1-\mu)\left[p_c d_c^*(\tilde{\mathbf{c}}_t, \tilde{\mathbf{c}}_r) + p_{\Delta c} d_{\Delta c}^*(\Delta\tilde{\mathbf{c}}_t, \Delta\tilde{\mathbf{c}}_r)\right] + \mu d_{\Delta E}^*(\Delta E_t, \Delta E_r) \tag{5.22}$$

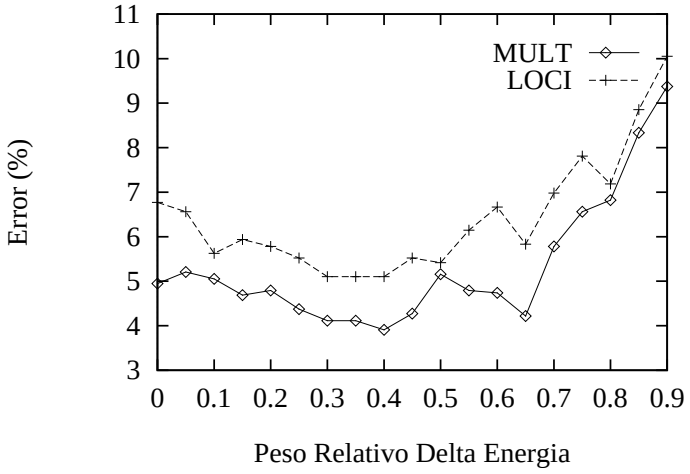


Figura 5.7: *Inclusión de la delta energía.*

con $d^*_{\Delta E} = d_{\Delta E}/\sigma^2_{\Delta E}$. Nuevamente, μ varía entre 0 (no delta energía) y 0.9 (predomina delta energía). El intervalo temporal para el cómputo de la delta energía es de ± 3 tramas. El valor óptimo ha sido hallado para $\mu = 0.35$ (fig. 5.7), por lo que se asigna a la delta energía un peso $p_{\Delta E} = 0.538$, manteniendo p_c y $p_{\Delta c}$ a los valores calculados en el apartado anterior. Los valores del error alcanzados por $\mu = 0.35$ son 4.11% para MULTI y 5.10% para LOCI.

Ajuste del peso de la Energía

Análogamente, estudiamos la utilidad de la energía mediante,

$$\begin{aligned} d(\mathbf{x}_t, \mathbf{x}_r) &= (1-\mu)\,[p_c d^*_c(\tilde{\mathbf{c}}_t, \tilde{\mathbf{c}}_r) + p_{\Delta c} d^*_{\Delta c}(\Delta\tilde{\mathbf{c}}_t, \Delta\tilde{\mathbf{c}}_r) + \\ &\quad p_{\Delta E} d^*_{\Delta E}(\Delta E_t, \Delta E_r)] + \mu d^*_E(E_t, E_r) \end{aligned} \tag{5.23}$$

con $d^*_E = d_E/\sigma^2_E$. Se ha realizado una experiencia similar, variando μ entre 0 y 0.9 (ver fig. 5.8). Las gráficas presentan un comportamiento bastante irregular, sin que se observe una mejora clara sobre el error, por lo que se asignará el peso $p_E = 0$ (no uso de la energía).

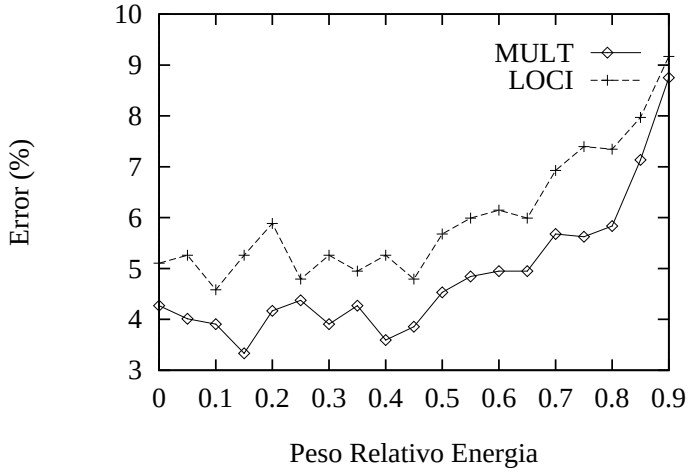


Figura 5.8: *Inclusión de la energía.*

$p_c = 1.0$
$p_{\Delta c} = 1.222$
$p_E = 0.0$
$p_{\Delta E} = 0.538$

Tabla 5.1: *Conjunto de pesos óptimo.*

5.4.2 Validez de la distancia DMP

Después de los ajustes realizados, el conjunto de pesos óptimo es el dado en la tabla 5.1. Por tanto, la distancia DMP a usar será de la forma,

$$d(\mathbf{x}_t, \mathbf{x}_r) = p_c d_c^*(\tilde{\mathbf{c}}_t, \tilde{\mathbf{c}}_r) + p_{\Delta c} d_{\Delta c}^*(\Delta\tilde{\mathbf{c}}_t, \Delta\tilde{\mathbf{c}}_r) + p_{\Delta E} d_{\Delta E}^*(\Delta E_t, \Delta E_r) \quad (5.24)$$

La forma óptima de esta distancia ha sido obtenida usando un diccionario de tamaño 64. En concreto, cabe preguntarse si el conjunto de pesos óptimos de la tabla 5.1 sigue siendo válido para tamaños mayores, es decir, si son *pesos universales*. Para ello, las proporciones entre d_c^*, $d_{\Delta c}^*$ y $d_{\Delta E}^*$ deberían mantenerse siempre constantes

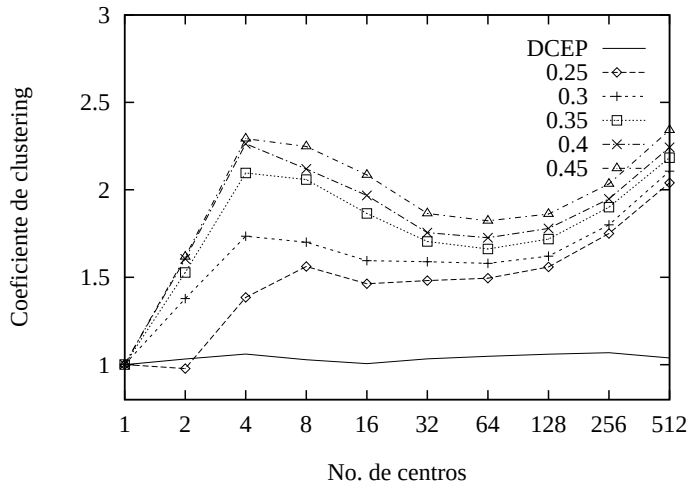


Figura 5.9: *Coeficientes de clustering, $K_{\Delta c}$ y $K_{\Delta E}$, para varios tamaños de diccionario.*

para cualquier tamaño, es decir,

$$\frac{\bar{d}^*_{\Delta c}}{\bar{d}^*_c} = K_{\Delta c} \qquad\qquad \frac{\bar{d}^*_{\Delta E}}{\bar{d}^*_c} = K_{\Delta E} \tag{5.25}$$

siendo $K_{\Delta c}$ y $K_{\Delta E}$ constantes sea cual sea el tamaño del diccionario, y donde $\bar{d}^*_c$, $\bar{d}^*_{\Delta c}$ y $\bar{d}^*_{\Delta E}$ son los valores medios de las distancias d^*_c, $d^*_{\Delta c}$ y $d^*_{\Delta E}$. Además, dado que para el caso de 1 centro $\bar{d}^*_c = \bar{d}^*_{\Delta c} = \bar{d}^*_{\Delta E} = 1$, debería cumplirse que esas constantes valiesen 1.

Se han trazado las curvas de $K_{\Delta c}$ y $K_{\Delta E}$, que serán denominados *coeficientes de clustering*, en la figura 5.9 (modo LOCI), para tamaños de diccionario $N = 1, 2, 4, 8, \ldots, 512$. La curva de $K_{\Delta c}$ (etiquetada como DCEP) se ha dibujado para los pesos de la tabla 5.1, y puede comprobarse que se ajusta bastante bien a la condición $K_{\Delta c} = 1$. Así, $p_{\Delta c}$ puede ser considerado como peso universal (junto con p_c, cuyo valor fue fijado arbitrariamente a 1). Las demás curvas de 5.9 corresponden a $K_{\Delta E}$, y fueron obtenidas con los pesos universales p_c y $p_{\Delta c}$, y varios valores de $\mu_{\Delta E}$ ($p_{\Delta E} = \mu_{\Delta E}/(1 - \mu_{\Delta E})$), incluyendo $\mu_{\Delta E} = 0.35$ (óptimo para 64 centros). Queda claro que $K_{\Delta E}$ no es constante cuando se varía el número

de centros, y que también depende del valor de $\mu_{\Delta E}$ empleado. Esto quiere decir que el peso óptimo de ΔE depende, en general, del tamaño N del diccionario empleado, por lo que en lo que sigue se notará dicho peso óptimo como $p^N_{\Delta E}$ (o $\mu^N_{\Delta E}$). Dado que este peso fue optimizado para 64 centros, debería cumplirse (para que se mantengan las contribuciones de las distancias parciales),

$$\frac{p^N_{\Delta E}\bar{d}^*_{\Delta E}(N, p^N_{\Delta E})}{p_c\bar{d}^*_c(N) + \bar{d}^*_{\Delta c}(N)} = \frac{p^{64}_{\Delta E}\bar{d}^*_{\Delta E}(64, p^{64}_{\Delta E})}{p_c\bar{d}^*_c(64) + \bar{d}^*_{\Delta c}(64)} \tag{5.26}$$

de donde se obtiene la condición,

$$p^N_{\Delta E}K_{\Delta E}(N, \mu^N_{\Delta E}) = A \tag{5.27}$$

donde $A = p^{64}_{\Delta E}K_{\Delta E}(64, \mu^{64}_{\Delta E}) = 0.895$. Para poder extraer el conjunto de valores $p^N_{\Delta E}$ óptimos, a partir de la ecuación (5.27), sería necesario conocer una forma paramétrica para la función $K_{\Delta E}(N, \mu_{\Delta E})$. Desgraciadamente, sólo se tiene un conocimiento experimental de esta función, dado en las curvas de la figura 5.9. Para una estimación aproximada, se puede considerar la siguiente parametrización para el intervalo 64-512 centros,

$$K_{\Delta E}(N, \mu_{\Delta E}) = R(\mu_{\Delta E}) + f(N) \tag{5.28}$$

basada en la regularidad de $K_{\Delta E}$ en dicho intervalo. En efecto, estas curvas presentan una variación del mismo tipo, dada por $f(N)$, pero desplazadas una cierta cantidad $R(\mu_{\Delta E})$ entre sí. Esta última función se ha obtenido mediante una regresión lineal de los puntos $K_{\Delta E}(64, \mu_{\Delta E})$, tal como se muestra en la gráfica 5.10. La curva $f(N)$ se ha obtenido promediando las distintas $K_{\Delta E}(N, \mu_{\Delta E})$ desplazadas al origen, obteniéndose los valores dados en la tabla 5.2. Ahora sí es posible obtener, aunque de forma aproximada, un conjunto de pesos óptimos, operando en la ecuación (5.27). Los resultados también se muestran en la tabla 5.2. Como puede comprobarse, la variación obtenida respecto al peso original $\mu^{64}_{\Delta E} = 0.35$, para los distintos tamaños de diccionario, no es lo suficientemente importante como salir del mínimo amplio observado en las curvas de inclusión de la ΔE (ver figura 5.7). Por tanto, es posible seguir usando el peso $\mu_{\Delta E} = 0.35$ ($p_{\Delta E} = 0.538$) como universal, en el rango de 64 a 512 centros. Respecto al rango 4-64 centros, también se puede seguir usando el mismo peso como universal, ya que las variaciones de $K_{\Delta E}$ en este rango no son superiores a las del rango 64-512.

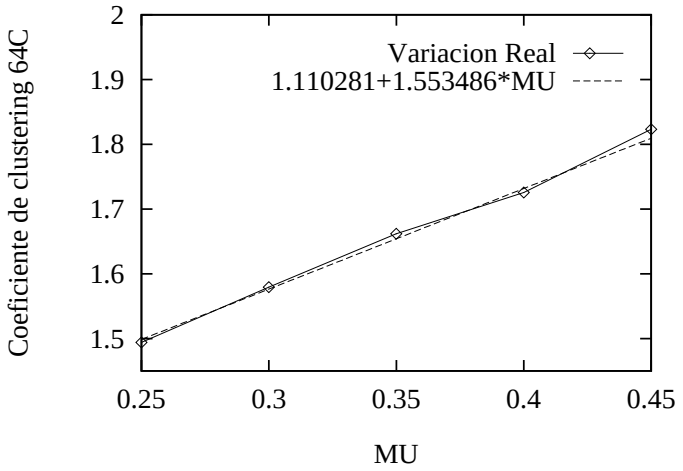


Figura 5.10: *Ajuste lineal de la función* $R(\mu_{\Delta E})$.

#Centros (N)	$f(N)$	$\mu^N_{\Delta E}$	$p^N_{\Delta E}$
64	0.00	0.350	0.538
128	0.05	0.344	0.524
256	0.23	0.325	0.481
512	0.52	0.298	0.424

Tabla 5.2: *Pesos óptimos de* ΔE *para varios tamaños de diccionario.*

En lo sucesivo se usará siempre la distancia pesada dada en (5.24) con el conjunto de pesos óptimos dado en la tabla 5.1. Para simplificar notaciones y desarrollos posteriores se considerará, equivalentemente, una distancia Euclídea entre vectores construídos de la siguiente forma:

$$\mathbf{x} = \{w_c\tilde{\mathbf{c}}, w_{\Delta c}\Delta\tilde{\mathbf{c}}, w_{\Delta E}\Delta E\} \tag{5.29}$$

donde $w_x = \sqrt{p_x/\sigma_x^2}$ con $x = c, \Delta c, \Delta E$. El número de dimensiones será $p = 29$ correspondientes a 14 coeficientes cepstrales, 14 delta cepstrales y la delta energía.

5.5 Ajuste del número de estados óptimo

Hasta ahora se han realizado todas las pruebas con $N = 10$ estados. Fijada la distancia (con sus pesos óptimos) en el apartado anterior, se busca ahora el número de estados óptimo, es decir, el que minimiza el error del sistema. La realización de este ajuste está motivada por el hecho de que trabajos básicos [Rabiner83, Rabiner89a] sobre reconocimiento con HMMs indican que la curva de error en función del número de estados no es monótonamente decreciente.

Para ajustar N, se han realizado nuevamente sendas experiencias MULTI y LOCI (ver figura 5.11). Se puede observar la tendencia a un mínimo en la zona de 9 a 15 estados (con un comportamiento bastante irregular). Para no aumentar excesivamente la complejidad de nuestro sistema, se seguirá usando el valor $N = 10$.

5.6 Estimaciones ML y MMI y Tamaño del diccionario

Hasta el momento, sólo se ha considerado la reestimación ML de los parámetros de los modelos HMM. Pero como ya se comentó en el capítulo 3, el método ML se apoya sobre una serie de suposiciones básicamente falsas. Por el contrario, el método MMI no necesita dichas suposiciones, pretendiendo, además, una disminución del error del sistema. Para comprobar la veracidad de lo anterior, se ha observado cómo evoluciona el error del sistema diseñado en los apartados precedentes, tanto en entrenamiento como en test, para un diccionario universal de 64 centros, usando el método MMI para modelos HMM discretos descrito en el capítulo 3 (ecuaciones (3.35)). Como punto de partida para el método iterativo del gradiente se toman los modelos obtenidos mediante Baum-Welch, desarrollando

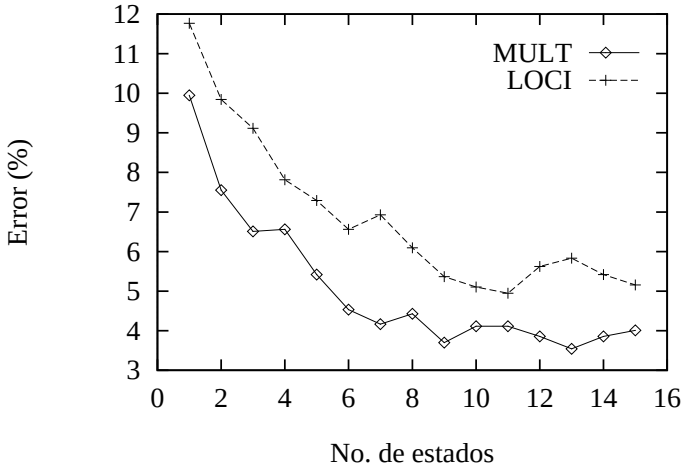
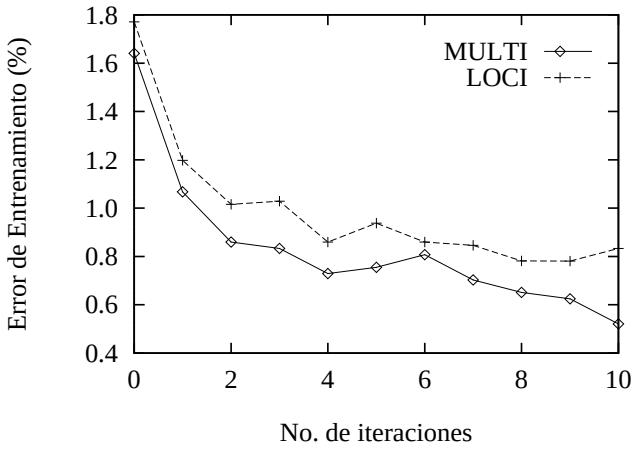


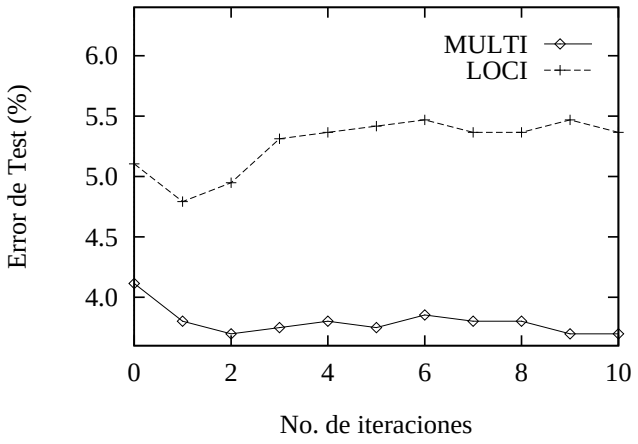
Figura 5.11: *Selección del número de estados óptimo.*

hasta 10 iteraciones de MMI. Como parámetro de convergencia se ha escogido experimentalmente $\eta = 0.1$. Los resultados de esta experiencia se muestran en la figura 5.12 (0 iteraciones corresponden a los modelos Baum-Welch originales). Puede comprobarse que el método MMI produce mejoras sustanciosas en la fase de entrenamiento, pero no así en reconocimiento, para el que sólo se observa una ligera mejora (sobre 0.5%) en modo MULTI, e, incluso, cierta degradación en modo LOCI.

Finalmente, se han desarrollado pruebas extensivas de reconocimiento para diccionarios con 64, 128, 256 y 512 centros, comparando el método ML y el MMI descrito anteriormente, obteniéndose los resultados mostrados en la tabla 5.3 (también en la gráfica 5.14 se proporciona la curva correspondiente para la estimación ML en modelos DHMM). Como cabía esperar, un aumento del número de centros se traduce en una mejora del rendimiento para ambos métodos. La conclusión final sobre el dilema ML/MMI, en modelos HMM discretos, es que el método MMI no aporta mejoras claras. Como causa plausible de este efecto cabe apuntar la "rigidez" que introduce la discretización mediante VQ del espacio de referencia.



(a) *Experiencias de entrenamiento*



(b) *Experiencias de test*

Figura 5.12: *Convergencia del método MMI.*

Modo	#Centros	Error (%) DHMM-ML	Error (%) DHMM-MMI	Error (%) SCHMM
MULTI	64	4.11	3.75	2.96
	128	2.55	2.76	1.92
	256	2.39	2.34	1.61
	512	1.40	1.35	0.78
LOCI	64	5.10	5.41	3.48
	128	4.63	4.58	2.76
	256	3.69	3.48	1.87
	512	3.17	3.22	1.66

Tabla 5.3: *Error de test para los modelados DHMM (estimaciones ML y MMI) y SCHMM para los modos MULTI y LOCI.*

5.7 Modelado semicontinuo

También se ha construído un sistema análogo basado en modelos SCHMM, usando los mismos valores para los parámetros característicos del sistema. El diccionario VQ universal es convertido en un conjunto de densidades gausianas de probabilidad (ver apartado 2.4 del capítulo 2), cuyas matrices de covarianza son supuestas diagonales, para simplificar el cálculo. Los parámetros HMM (matrices A y B) son estimados independientemente usando el procedimiento de Baum-Welch (su modificación para modelos semicontinuos es detallada en el capítulo 7). En la gráfica 5.13 se muestra la variación del error del sistema en función del número de candidatos, usando un diccionario de 64 centros. Puede observarse que el error disminuye hasta 4 candidatos, presentando un comportamiento irregular a partir de ese punto. En lo sucesivo se considerarán sólo los 4 mejores candidatos. Cabe esperar que para un tamaño de diccionario superior a 64 centros, el número de candidatos principales (que contribuyen a una disminución del error) sea mayor, aunque no se ha considerado aquí por razones de volumen de cálculo.

La figura 5.14 muestra una comparación del error de reconocimiento para DHMM y SCHMM (los valores de error correspondientes se proporcionan en la tabla 5.3). Se observa una clara mejora debida al uso de SCHMMs, aunque, claro está, a costa de un aumento de la complejidad computacional del sistema.

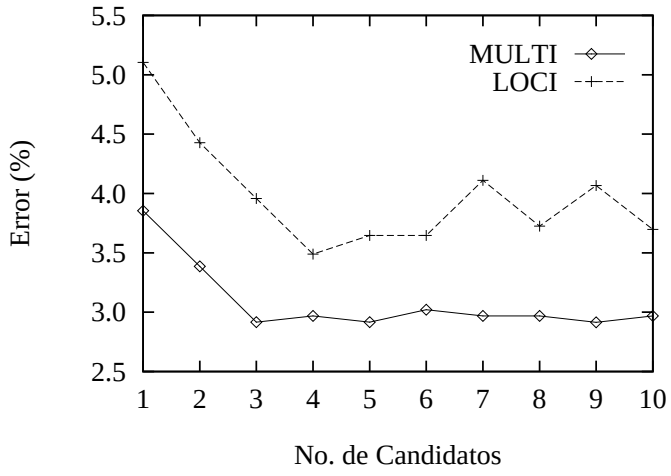
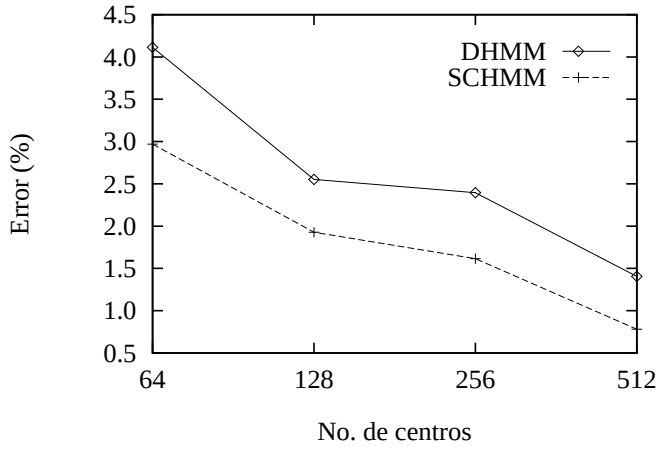
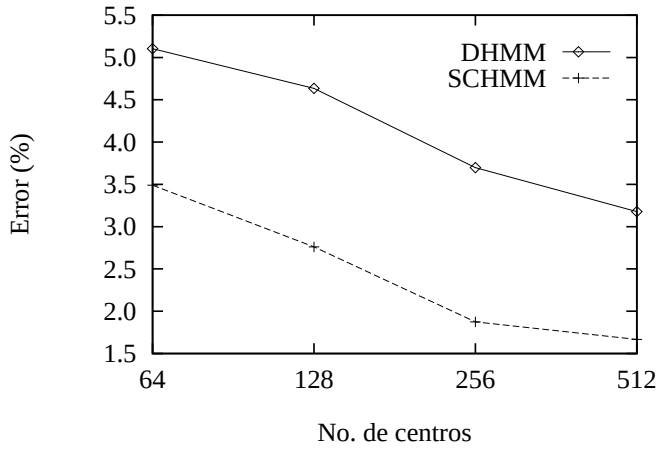


Figura 5.13: *Variación del error con el número de candidatos de cuantización en modelos SCHMM en los modos MULTI y LOCI.*



(a) *Experiencia MULTI*



(b) *Experiencia LOCI*

Figura 5.14: *Comparación de modelos DHMM y SCHMM en función del número de centros.*

Capítulo 6

ESTIMACIÓN DE PARÁMETROS EN MODELOS MVQ

En este capítulo se abordará el problema de la estimación de los parámetros de un sistema de reconocimiento de voz mediante modelos MVQHMM. Los parámetros básicos del sistema serán los mismos determinados en el capítulo anterior. En primer lugar, se establece la forma general de realizar la estimación ML de modelos MVQ. Posteriormente, se buscará una forma conveniente para las matrices de covarianza de las densidades de probabilidad asociadas a los centros de cada uno de los diccionarios, haciendo uso de la estimación ML descrita en el capítulo 3. Posteriormente, se realizará una estudio de estimaciones discriminativas tipo MMI y MCE sobre modelos MVQ.

6.1 Estimación ML de modelos MVQ

En el capítulo 2 se estableció que el modelado MVQ usaba la siguiente forma para las densidades de probabilidad de producción de observaciones,

$$b_i(\mathbf{x}) = P(\mathbf{x}|o,\lambda)b_i(o) \tag{6.1a}$$

$$o = \max_{v_j \in V(\lambda)}{}^{-1} \left[P(\mathbf{x}|v_j,\lambda)\right] \tag{6.1b}$$

También se estableció que la probabilidad de una secuencia dado un modelo MVQ podía descomponerse en el producto de las probabilidades de cuantización y de generación (ver ecuación (2.77)). Así, el proceso de reconocimiento corresponde al mostrado en la figura 6.1, donde cada modelo MVQHMM se compone de un diccionario DIC (proceso VQ) y un modelo HMM discreto (proceso de evaluación de la secuencia de observaciones discretas O).

Si consideramos que el conjunto de parámetros del modelo también se puede descomponer como $\lambda = (\theta, \phi)$, donde θ representa a los parámetros del conjunto de densidades $P(\mathbf{x}|v_j,\lambda)$ (asociadas al diccionario de λ) y $\phi = (A, B, \Pi)$ a los parámetros del modelo discreto, el problema de la estimación ML para una secuencia de vectores $X = x_1, \ldots, x_T$, cuya representación cuantizada por diccionario θ es $O = o_1, \ldots, o_T$, puede formularse como,

$$\max_{\lambda} P(X|\lambda) = \max_{\lambda} \left[P(X|O,\lambda)P(O|\lambda)\right] = \left[\max_{\theta} P(X|O,\theta)\right]\left[\max_{\phi} P(O|\phi)\right] \tag{6.2}$$

Por tanto, la estimación ML de λ consiste en una estimación ML independiente de los parámetros de θ y de ϕ. Los primeros pueden obtenerse aplicando el algoritmo LBG para construir un diccionario VQ y asociando a cada centro una densidad de probabilidad de tipo gausiano (esto constituye una aproximación a una estimación ML, como ya se mencionó en el capítulo 2). Los segundos pueden obtenerse aplicando el algoritmo Baum-Welch (una vez obtenido el diccionario VQ y las cuantizaciones correspondientes de la secuencia de entrenamiento), tratando los parámetros de ϕ de forma análoga a los de un modelo discreto. Una demostración rigurosa de la independencia de las estimaciones para cuantización y generación en un modelo MVQ será dada en el próximo capítulo.

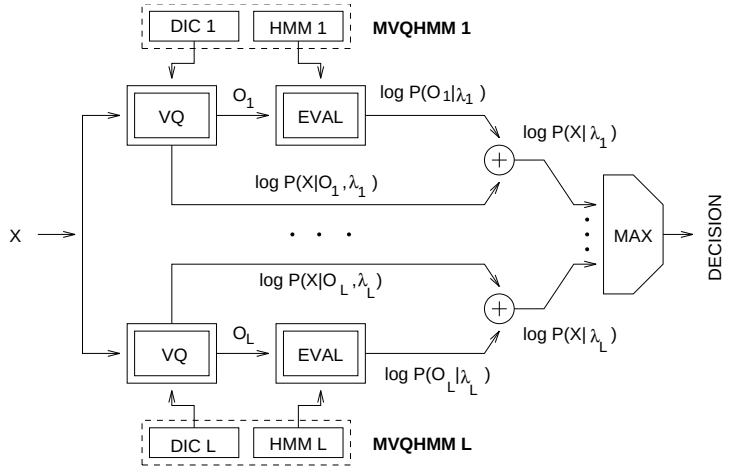


Figura 6.1: *Sistema de reconocimiento basado en modelos MVQ.*

6.2 Matrices de covarianza de los centros

Como se acaba de establecer, la probabilidad de que el vector de entrada x corresponda a clase $o_j \in V(\lambda)$ dado el modelo λ corresponde a una densidad tipo gausiano, con vector media $\mathbf{y}_j$ y cuya matriz de covarianza $\Sigma_j = \left\{\sigma^2_{jik}\right\}$ se supondrá diagonal (no correlación entre los parámetros de características),

$$\sigma^2_{jik} = \begin{cases} \sigma^2_{ji} \neq 0 & i = k \\ 0 & i \neq k \end{cases} \tag{6.3}$$

con lo que puede escribirse,

$$P(\mathbf{x}|o_j, \lambda) = \prod_{i=1}^{p} \frac{1}{\sqrt{2\pi\sigma^2_{ji}}} \exp\left\{-\frac{1}{2}\frac{(x_i - y_{ji})^2}{\sigma^2_{ji}}\right\} \tag{6.4}$$

donde p representa el número de parámetros del vector de características.

Hay que realizar dos observaciones respecto a la forma de la matriz de covarianza Σ_j:

1) Para que la medida de probabilidad dada por (6.4) sea coherente con la distancia pesada descrita en el capítulo anterior, es necesario que todos los elementos de la diagonal principal sean iguales entre sí, e iguales para todos

los centros. De esta forma se puede asegurar que, dado un vector de entrada, el centro más probable es también el más próximo.

2) La estimación de las matrices de covarianza completas implica la estimación de un número elevado de parámetros, por lo que el problema de entrenamiento insuficiente de dichos parámetros podría adquirir cierta importancia.

Los dos puntos que acabamos de enunciar nos llevan a plantear las siguientes experiencias:

EXP1: Se utiliza una matriz de covarianza por centro Σ_j (tal como esta indicado en (6.4))

EXP2: Sólo se utiliza una matriz de covarianza por diccionario Σ_λ. Cada elemento de la diagonal principal es la distorsión promedio del parámetro correspondiente en el diccionario λ.

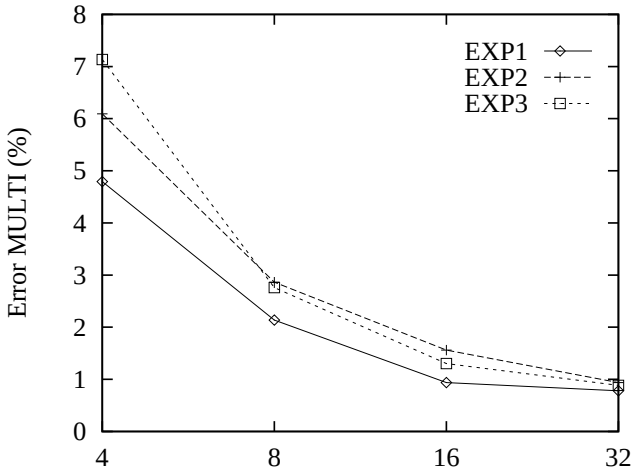
EXP3: Se utiliza una sola matriz de covarianza por diccionario $\Sigma_\lambda = \sigma_\lambda^2 I$ (I es la matriz identidad) con todos los elementos de la diagonal principal iguales entre sí e iguales a la distorsión media por parámetro del diccionario σ_λ^2.

La figura 6.2 muestra una comparación de esta experiencias en los modos MULTI y LOCI para distintos tamaños de los diccionarios. Queda clara la poca utilidad de EXP2 frente a las demás. EXP1 tiene el mejor comportamiento en modo MULTI, aunque EXP3 consigue resultados similares para 32 centros. En modo LOCI, EXP3 sólo es superada por EXP1 para el caso de 4 centros. La degradación del sistema para EXP1-2 (respecto a EXP3) conforme crece el número de centros confirma la segunda de las observaciones hechas al inicio del apartado, ya que también es mayor el número de parámetros a estimar. Además, la experiencia EXP3 es la única, de las tres realizadas, coherente con la distancia pesada empleada en este trabajo (observación primera). En lo sucesivo se empleará la forma descrita en EXP3 para la matriz de covarianza, lo cual implica que,

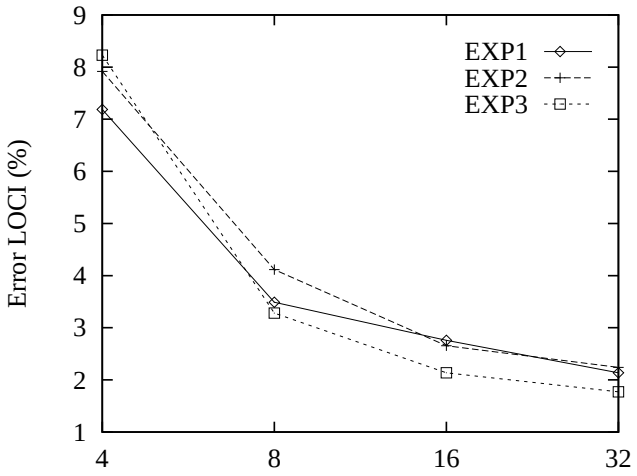
$$P(\mathbf{x}|o_j,\lambda) = (2\pi\sigma_\lambda^2)^{-p/2} \exp\left\{-\frac{1}{2\sigma_\lambda^2}\|\mathbf{x}-\mathbf{y}_j\|^2\right\} \tag{6.5}$$

y la probabilidad de cuantización de una secuencia X será de la forma:

$$P(X|O,\lambda) = \prod_{t=1}^{T} P(\mathbf{x}_t|o_t,\lambda) = \prod_{t=1}^{T} (2\pi\sigma_\lambda^2)^{-p/2} \exp\left\{-\frac{1}{2\sigma_\lambda^2}\|\mathbf{x}_t-\mathbf{y}_t\|^2\right\} \tag{6.6}$$



(a) *Experiencia MULTI*



(b) *Experiencia LOCI*

Figura 6.2: *Error en función del número de centros para las experiencias EXP1, EXP2 y EXP3.*

y en forma logarítmica,

$$\log P(X|O,\lambda) = -\frac{pT}{2}\left[\log(2\pi\bar{D}_\lambda/p) + \frac{D_\lambda(X)}{\bar{D}_\lambda}\right] \tag{6.7}$$

donde $\bar{D}_\lambda = p\sigma_\lambda^2$ es la distorsión media del diccionario del modelo λ y $D_\lambda(X)$ es la distorsión media de la secuencia X en dicho diccionario, definida como,

$$D_\lambda(X) = \frac{1}{T}\sum_{t=1}^{T}\|\mathbf{x}_t - \mathbf{y}_t\|^2 \tag{6.8}$$

por lo que, como se puede ver, la probabilidad logarítmica de cuantización en el modelo λ está linealmente relacionada con la distorsión media de la secuencia X en el modelo correspondiente.

6.3 Composición de probabilidades

En las experiencias precedentes se ha usado como medida de evaluación ó puntuación de secuencias la expresión la suma de las probabilidades logarítmicas de cuantización y de generación,

$$\log P(X|\lambda) = \log P(X|O,\lambda) + \log P(O|\lambda) \tag{6.9}$$

donde la probabilidad de cuantización logarítmica tiene una relación lineal con la distorsión de la secuencia (ecuación (6.7)), mientras que la de generación se corresponde con la de un modelo discreto. De hecho, la motivación del modelado MVQ es la de incorporar a la puntuación tradicional proporcionada por el modelado DHMM, otra relacionada con la distorsión de la secuencia (la idea de usar las distorsiones promedio de una señal de entrada en un conjunto de diccionarios, prescindiendo del alineamiento temporal, ya ha sido empleada con éxito anteriormente por Burton et al [Burton85]). Se plantea aquí el interrogante de si esa incorporación está óptimamente realizada. Por ejemplo, un problema de tipo práctico, que se presenta realmente, es el de cómo responde la puntuación total dada en (6.9) cuando se añade al vector de características algún nuevo parámetro cuya varianza sea muy pequeña respecto a las de los demás. La respuesta es que modificará poco la distorsión media del diccionario $\bar{D}_\lambda$ y de la secuencia $D_\lambda(X)$, pero el valor p se ve incrementado en una unidad, lo cual hace que se incremente el valor de la puntuación de cuantización respecto a la de generación en la misma

proporción, como se desprende de la expresión (6.7), dándole más importancia a la primera. La consecuencia es clara: en las experiencias realizadas hasta el momento no se sabe si la composición de ambas puntuaciones es óptima.

La discusión anterior plantea la necesidad de experimentar nuevamente el sistema construído con la siguiente expresión para evaluación,

$$\log P(X|\lambda) = \alpha \log P(X|O,\lambda) + \log P(O|\lambda) \tag{6.10}$$

con

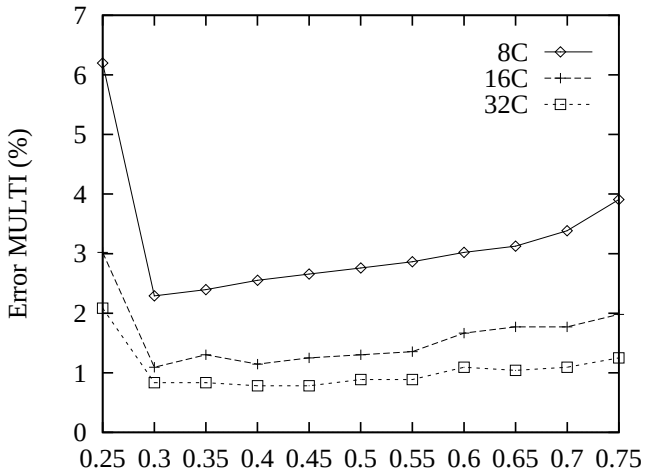
$$\alpha = \frac{\mu}{1-\mu} \tag{6.11}$$

siendo μ el factor de composición, variable desde 0 (solo probabilidad de generación) hasta 1 (sólo probabilidad de cuantización). La figura 6.3 muestra el resultado de esta experiencia para distintos valores de μ (desde 0.25 a 0.75) y número de centros por diccionario (8,16 y 32). Se observa un mínimo acusado en la zona $\mu = 0.3, 0.35$, por lo que en lo sucesivo se usará $\mu = 0.35$ ($\alpha = 0.538$).

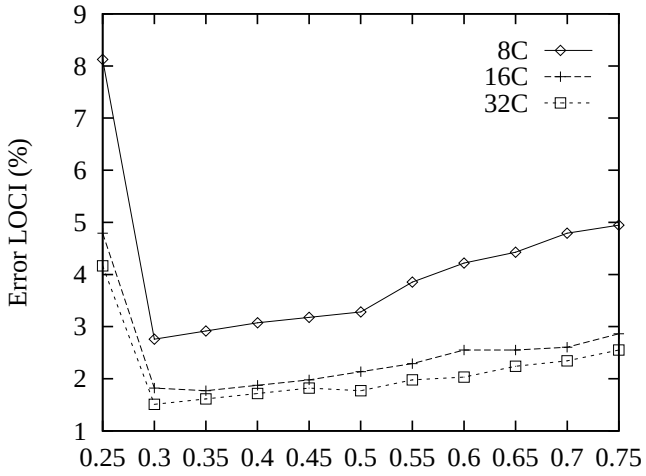
Otra observación importante debe ser hecha en las gráficas 6.3. Antes de llegar al mínimo de error, es decir, cuando predomina el reconocimiento con probabilidades de generación, el error decrece muy rápidamente hacia dicho mínimo. Una vez superado el valor crítico, cuando ya predomina el reconocimiento con probabilidades de cuantización, la pendiente de la curva de error es bastante más suave. Esto es una prueba de que el peso más importante en el proceso de reconocimiento con modelos MVQ recae principalmente sobre las probabilidades de cuantización [Segura91].

6.4 Rendimiento del modelado MVQ

En la figura 6.4 se comparan los resultados de reconocimiento en modo MULTI y LOCI de los modelos MVQ con la composición de probabilidades dada en la expresión (6.10) (con $\alpha = 0.538$), y los correspondientes a modelos DHMMs y SCHMMs (los mismos estimados en el capítulo anterior). Ya que el número de unidades es 16 (en este caso, el número de palabras del vocabulario), un conjunto de 16 diccionarios con N centros (caso MVQ) es equivalente (en cuanto a número total de centros) a un diccionario universal con 16*N centros (caso DHMM y SCHMM). Por ello, se hace uso de las etiquetas "4/64", "8/128", "16/256" y "32/512" en dicha gráfica. En cuanto a cantidad de cálculo en reconocimiento, los modelos MVQ



(a) *Experiencia MULTI*



(b) *Experiencia LOCI*

Figura 6.3: *Variación del error en función del peso* μ.

serán directamente comparables con los DHMM. En entrenamiento, el uso de modelos MVQ supone un considerable ahorro de cálculo respecto a los DHMM, ya que la estimación de 16 diccionarios de N centros es menos costosa que la estimación de un único diccionario de 16*N centros al ser la complejidad del algoritmo LBG exponencialmente creciente con el número de centros. También, la construcción de modelos HMM con un alfabeto de símbolos 16 veces inferior es claramente más barata. Los modelos SCHMM no son comparables en reconocimiento, y mucho menos en entrenamiento, a los dos anteriores, por manejar varios candidatos de cuantización por vector de entrada al sistema, pero se ha considerado conveniente comprobar su comportamiento frente a los MVQ.

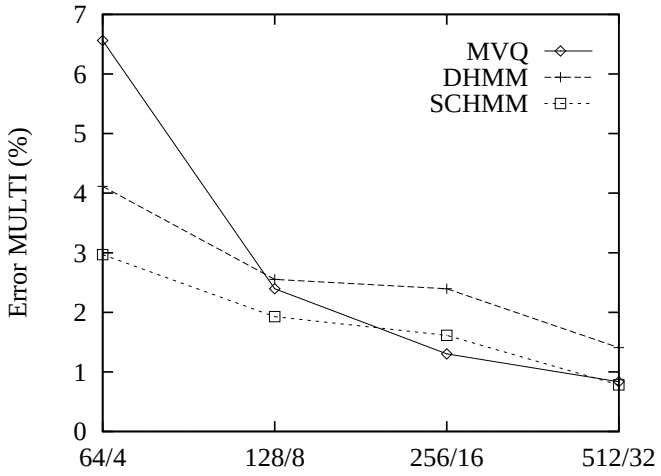
En el caso de 4/64 centros, queda claro que los modelos DHMM, aún perdiendo información sobre la señal en el proceso VQ, son muy superiores a los MVQ, dado que todos los modelos comparten un mismo diccionario de 64 centros, mientras que los diccionarios MVQ de 4 centros no pueden modelar correctamente la variedad acústica de las palabras del vocabulario. Esta incapacidad frente a los DHMM queda claramente superada si se usan más de 8/128 centros por diccionario, tanto para MULTI como para LOCI. Los modelos MVQ muestran, incluso, un rendimiento similar (8/128 centros) o ligeramente superior (desde 16/256 centros) a los SCHMM. Estos resultados resaltan la potencia del nuevo modelado MVQ, tanto en error como en volumen de cálculo, frente a los modelados clásicos DHMM o SCHMM.

En lo sucesivo, y dado el bajo nivel de error alcanzado, se omitirán las experiencias Multilocutor, concentrando la atención en un sistema Independiente del locutor, para el que el margen de error remanente permite esperar aún mejoras apreciables.

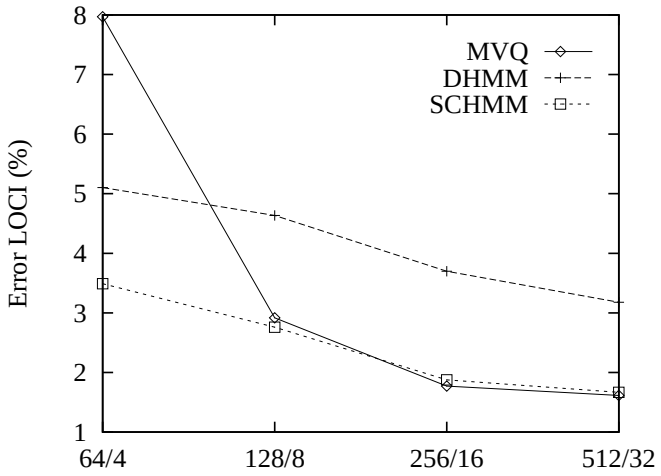
6.5 Estimación MMI en modelos MVQHMM

Según se estudió en el capítulo 3, la estimación MMI de un conjunto de modelos MVQ $\Lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_L\}$, entrenados por una secuencia X y para el caso de un modelo de lenguaje equiprobable, podrá obtenerse mediante la maximización de la función,

$$I_m(\Lambda) = \log P_m(\Lambda) = \log \frac{P(X|\lambda_m)}{\sum_{l=1}^{L} P(X|\lambda_l)} \tag{6.12}$$



(a) *Experiencia MULTI*



(b) *Experiencia LOCI*

Figura 6.4: *Variación del error en función del número de centros para MVQ, DHMM y SCHMM.*

Dado que la probabilidad $P(X|\lambda)$ se puede descomponer como producto de la probabilidad de cuantización $P(X|O,\lambda)$ y la probabilidad de generación $P(O|\lambda)$, y la primera depende sólo de los parámetros del diccionario, y la segunda sólo de los del modelo DHMM asociado, se tiene que las derivadas de I_m respecto a los parámetros del DHMM serán similares a las que se dedujeron el capítulo 3 (ecuaciones (3.35)), con la diferencia de que las probabilidades P_s incluyen ahora también probabilidades de cuantización, dado que tienen la siguiente expresión,

$$P_s(\Lambda) = \frac{P(X|\lambda_s)}{\sum_{l=1}^{L} P(X|\lambda_l)} \tag{6.13}$$

La diferencia fundamental respecto a los modelos discretos es que ahora la estimación discriminativa se extiende a los diccionarios VQ. La derivada de I_m respecto a la componente k de un centro $\mathbf{y}_j^s$ viene dada por,

$$\frac{\partial I_m}{\partial y_j^s(k)} = \frac{\partial I_m}{\partial P(X|\lambda_s)} \frac{\partial P(X|\lambda_s)}{\partial y_j^s(k)} \tag{6.14}$$

donde,

$$\frac{\partial I_m}{\partial P(X|\lambda_s)} = \frac{\delta_{m,s} - P_m}{P(X|\lambda_m)} \tag{6.15}$$

$$\frac{\partial P(X|\lambda_s)}{\partial y_j^s(k)} = \sum_{t=1}^{T} \left[\prod_{\tau \neq t} P(\mathbf{x}_\tau|\lambda_s) \frac{\partial P(\mathbf{x}_t|\lambda_s)}{\partial y_j^s(k)} \right] \tag{6.16}$$

$$\frac{\partial P(\mathbf{x}_t|\lambda_s)}{\partial y_j^s(k)} = P(\mathbf{x}_t|\lambda_s) \frac{x_t(k) - y_j^s(k)}{\sigma_\lambda^2} \delta_{o_t,v_j} \tag{6.17}$$

Finalmente, puede escribirse la siguiente expresión,

$$\frac{\partial I_m}{\partial y_j^s(k)} = (\delta_{m,s} - P_s) \sum_{t=1}^{T} \frac{x_t(k) - y_j^s(k)}{\sigma_\lambda^2} \delta_{o_t,v_j} \tag{6.18}$$

También es posible derivar respecto a los parámetros σ_λ^2 para obtener la reestimación correspondiente,

$$\frac{\partial I_m}{\partial \sigma_{\lambda_s}^2} = (\delta_{m,s} - P_s) \frac{Tp}{2\sigma_{\lambda_s}^2} \left[\frac{D_{\lambda_s}(X)}{\bar{D}_{\lambda_s}} - 1 \right] \tag{6.19}$$

En el caso de múltiple secuencia de entrenamiento, basta sumar las contribuciones de cada una de esas secuencias, tal como se indica en la ecuación (3.25). La estimación MMI de los modelos MVQ no es una estimación totalmente desacoplada como en el caso de aplicar una estimación ML, ya que en las probabilidades P_s intervienen probabilidades de cuantización y probabilidades de generación.

6.6 Diseño MMI de diccionarios MVQ

En el presente trabajo sólo se hará uso de la estimación MMI para realizar el diseño de los diccionarios VQ. Varias razones apoyan esta decisión:

1) Los resultados de estimación MMI en modelos DHMM no arrojaron mejoras significativas respecto a la estimación ML, y los HMMs usados en modelado MVQ, sin considerar la parte VQ, son de hecho discretos.

2) La estimación MMI de los HMMs en un sistema MVQ incrementaría considerablemente la complejidad del entrenamiento, ya que es necesario obtener en cada iteración del procedimiento las versiones cuantizadas de cada una de las secuencias de entrenamiento en cada uno de los diccionarios.

3) La mayor contribución a la disminución del error en un sistema MVQ, tal como se ha visto anteriormente, se debe a las probabilidades de cuantización. Si el método MMI produce alguna mejora apreciable, ésta debe producirse en la parte VQ.

4) Dado que se manejan diccionarios de tamaño reducido, el reconocimiento con sólo probabilidades de generación no es demasiado preciso, y la recuperación de errores en esta fase no es sencillo. En la gráfica 6.5 se puede ver cómo evoluciona el error para entrenamientos MMI (realizados para experiencia LOCI_1) de los modelos HMM (solo matrices Π, A y B) en los modelados DHMM y MVQHMM (en ambos casos se usan diccionarios ML). Se puede observar que, efectivamente, la recuperación de errores en el caso MVQ es despreciable.

Atendiendo a los cuatro puntos anteriores, se ha considerado conveniente el realizar únicamente un *diseño MMI* para los diccionarios VQ, usando la expresión (6.18) para reestimación de sus centros. Los parámetros σ^2_λ siguen siendo estimados como distorsiones medias de los diccionarios, y no serán reestimados mediante

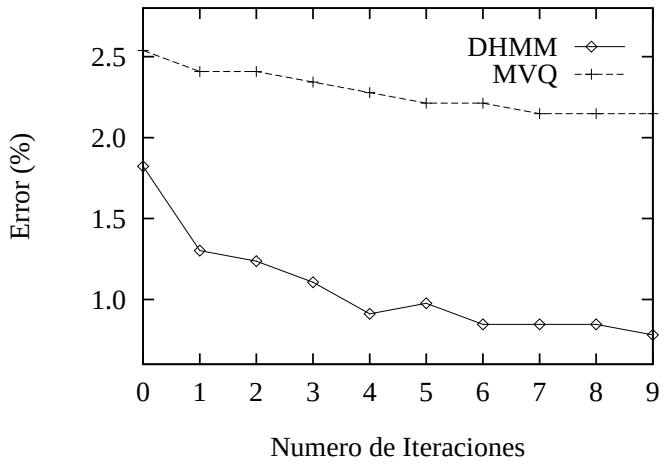


Figura 6.5: *Evolución del error de entrenamiento MMI (matrices* Π*,* A *y* B*) sobre modelos DHMM (64 centros) y MVQ (8 centros) usando diccionarios LBG para la experiencia LOCI_1.*

(6.19) (este punto será discutido en un subapartado posterior). Para que este diseño, al que se hará referencia como *MMI-MVQ*, sea totalmente independiente de la estimación de los parámetros de las matrices Π, A y B, las probabilidades P_s deben incluir sólo probabilidades de cuantización,

$$P_s(\Theta) = \frac{P(X|O_s, \lambda_s)}{\sum_{l=1}^{L} P(X|O_l, \lambda_l)} = \frac{P(X|\theta_s)}{\sum_{l=1}^{L} P(X|\theta_l)} \tag{6.20}$$

donde $\Theta = \{\theta_1, \ldots, \theta_L\}$ es el subconjunto de Λ que contiene únicamente los parámetros de los diccionarios VQ, siendo $\theta_i \subset \lambda_i$ los parámetros del diccionario del modelo λ_i.

La construcción del sistema se puede resumir en cuatro puntos:

1) Construcción de un conjunto de diccionarios iniciales mediante uso del algoritmo LBG con bipartición.

2) Reestimación de los diccionarios usando la fórmula (6.18) (con la forma de P_s dada en (6.20)) y aplicando un número suficiente de iteraciones (diseño MMI-MVQ). Los parámetros σ_λ^2 siguen estimándose como distorsiones medias.

3) Estimación de los HMM discretos (matrices Π, A y B) usando el algoritmo de Baum-Welch.

4) Reconocimiento mediante la expresión (6.10).

Para ilustrar cómo se actúa sobre los diccionarios en el paso 2 antes descrito, se ha efectuado la siguiente experiencia: se dispone de un conjunto de secuencias de entrenamiento, cada una de las cuales está formada por 2 vectores bidimensionales. Estas secuencias son,

$$\text{Clase 1 (W1)} \begin{cases} X_1^1 = (0.70, 1.30)(2.60, 3.00) \\ X_2^1 = (0.70, 0.70)(3.00, 3.49) \\ X_3^1 = (1.30, 1.30)(3.00, 2.51) \\ X_4^1 = (1.30, 0.70)(3.40, 3.00) \end{cases}$$

$$\text{Clase 2 (W2)} \begin{cases} X_1^2 = (1.20, 1.00)(3.00, 2.30) \\ X_2^2 = (1.50, 1.40)(3.00, 1.70) \\ X_3^2 = (1.50, 0.60)(2.60, 2.00) \\ X_4^2 = (1.80, 1.00)(3.40, 2.00) \end{cases}$$

a las que les corresponden 2 diccionarios LBG de 2 centros,

$$\theta_1 \text{ (C1)} \begin{cases} y_1^1 = (1.00, 1.00) \\ y_2^1 = (3.00, 3.00) \end{cases} \quad \theta_2 \text{ (C2)} \begin{cases} y_1^2 = (1.50, 1.00) \\ y_2^2 = (3.00, 2.00) \end{cases}$$

Este esquema presenta un error para la secuencia X_3^1 si se usan las probabilidades $P(X|\theta)$ de cuantización para clasificación. Se ha supuesto $\sigma_{\lambda_1}^2 = \sigma_{\lambda_2}^2 = 1$, y se han aplicado 10 iteraciones de la expresión (6.18), con parámetro de convergencia $\eta = 1.0$. La figura 6.6 muestra cómo evolucionan los centros de cada diccionario. Al cabo de 5 iteraciones, el error en X_3^1 desaparece. El efecto principal del proceso es alejar los centros de vectores incorrectos, sin perjudicar a los correctos. Es evidente la gran diferencia entre los diccionarios libres de error (de la iteración 6 en adelante) y los LBG originales.

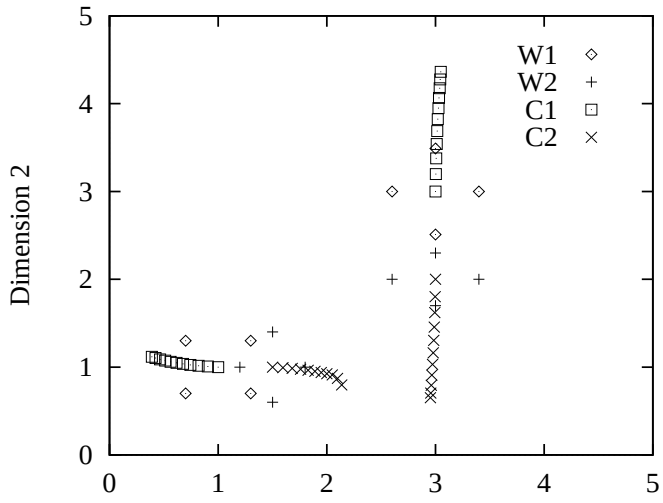


Figura 6.6: *Evolución de diccionarios MVQ mediante diseño MMI-MVQ.*

6.6.1 Normalización temporal y aproximación al error empírico

En la expresión (6.18), el peso de la corrección a que da lugar una secuencia X viene dado por el factor $(\delta_{m,s} - P_s)$. A su vez, la probabilidad P_s responde a la ecuación (6.20), dependiendo directamente de las probabilidades $P(X|\theta_l)$ $(l = 1, \ldots, L)$. Cada una de estas probabilidades se desarrolla según (6.6). De esta expresión se deduce que el valor de $P(X|\theta_l)$ va haciéndose paulatinamente más pequeño conforme crece la duración T. De esta manera, la probabilidad P_s y, por tanto, el factor $(\delta_{m,s} - P_s)$ presentan una dependencia acusada con T. Ya que las $P(X|\theta_l)$ se obtienen como un producto de T probabilidades, la raíz T-sima proporciona una normalización temporal que elimina esa dependencia, haciendo que el factor de corrección $(\delta_{m,s} - P_s)$ no dependa de la longitud de la secuencia. Las probabilidades P_s deben ser expresadas ahora como,

$$P_s(\Theta) = \frac{P(X|\theta_s)^{1/T}}{\sum_{l=1}^{L} P(X|\theta_l)^{1/T}} \tag{6.21}$$

El efecto de esta normalización puede entenderse como sigue: si se elevan las probabilidades $P(X|\theta_l)^{1/T}$ a la potencia T en la ecuación (6.21) (es decir, volver a la P_s original), lo que se hace, dada la no linealidad de la operación, es primar a la mayor (o mayores) de esas $P(X|\theta_l)^{1/T}$, es decir, aumentar la importancia del modelo (o modelos) mas probable. Así puede entenderse que en el límite $T \to \infty$ se tiene $P_s \to 1$, si $P(X|\theta_s)$ es la máxima entre las $P(X|\theta_l)$, ó $P_s \to 0$ en caso contrario. Además, en dicho límite ($T \to \infty$), $|\delta_{m,s} - P_s|$ coincidirá con el error empírico,

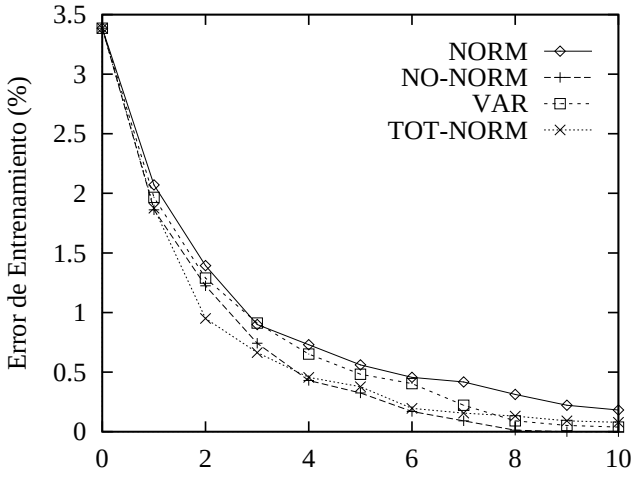
$$|\delta_{m,s} - P_s| \longrightarrow \begin{cases} 0 & m = s \\ 1 & m \neq s \end{cases} \tag{6.22}$$

si $P(X|\theta_s)^{1/T}$ es el máximo, o

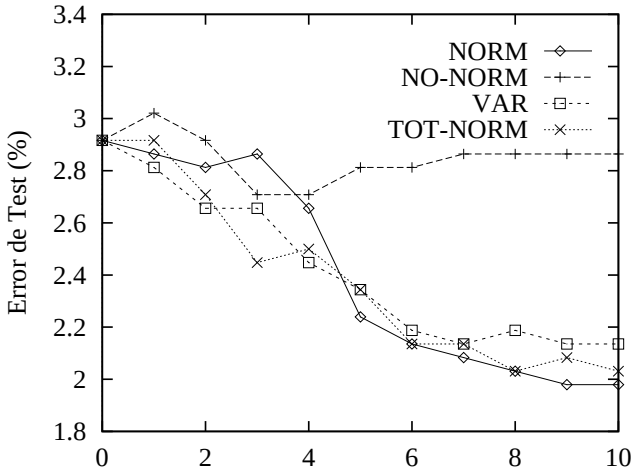
$$|\delta_{m,s} - P_s| \longrightarrow \begin{cases} 1 & m = s \\ 0 & m \neq s \end{cases} \tag{6.23}$$

en caso contrario.

Se puede concluir que el efecto de esta normalización temporal es el de suavizar $|\delta_{m,s} - P_s|$, haciendo que aumente la influencia de los modelos menos probables. Como ya se vió en el capítulo 3, el suavizado del error empírico es necesario para obtener funciones de buen comportamiento, aptas para diferenciación. Para comprobar el efecto de la normalización temporal sobre el error del sistema, se han realizado pruebas sobre un sistema MVQ con 8 centros por diccionario en entrenamiento y test, para probabilidades normalizadas y sin normalizar, empleando el entrenamiento descrito en el apartado anterior, con 10 iteraciones MMI (modo LOCI) y con parámetro de convergencia $\eta = 0.2$. Los resultados se muestran en la figura 6.7. El error de entrenamiento se obtiene usando sólo probabilidades de cuantización. Para el error de test se usa la puntuación dada en (6.10). En la fase de entrenamiento no se detectan grandes diferencias, aunque la curva correspondiente a no normalización (NO-NORM) converge de una forma ligeramente más rápida. En cambio, la normalización temporal (curva NORM) se muestra claramente más potente durante el test. Una explicación plausible a este efecto es que la no normalización hace más importantes las correcciones debidas a errores detectados durante el entrenamiento, mientras que la normalización aumenta la importancia de los casi-errores. Así, la corrección de los casi-errores de entrenamiento parece ser mucho más relevante en la corrección de errores de test que la de los propios errores de entrenamiento. El uso de probabilidades normalizadas también ha sido



(a) *Evolución del Error de Entrenamiento*



(b) *Evolución del Error de Test*

Figura 6.7: *Evolución del error del sistema con y sin normalización temporal. Efecto de la estimación de* σ^2_λ.

experimentado sobre modelos DHMM sin que produzcan ninguna mejora.

6.6.2 Estimación de σ_λ^2

Como se indicó anteriormente, los parámetros σ_λ^2 conservan, en el diseño MMI-MVQ propuesto, el sentido de distorsiones medias de los diccionarios, en lugar de ser reestimados mediante (6.19). Para justificar esta decisión se ha añadido en las gráficas 6.7 unas nuevas curvas (VAR) que corresponden a un sistema similar al de las curvas NORM, pero en el que además se realiza una estimación de σ_λ^2 según (6.19). Puede observarse que dicha estimación produce en entrenamiento una convergencia ligeramente más rápida que la correspondiente al caso NORM. La explicación es sencilla: hay un parámetro más que es entrenado discriminativamente, y ello contribuye a una reducción más efectiva del error, ya que sólo se reconoce con probabilidades de cuantización. Esto no es así para las curvas en modo de test. En las primeras iteraciones, la curva VAR presenta un mejor comportamiento, tal como sucedía en entrenamiento. Esta tendencia se invierte a partir de la quinta iteración. Una explicación plausible a este comportamiento es que el parámetro σ_λ^2 actúa como factor de normalización de la distancia medida para una secuencia en la puntuación dada por (6.7). Cuando es entrenado de una forma discriminativa, pierde ese papel, y la composición de probabilidades de cuantización y generación deja de ser óptima. Por esta razón se ha optado por mantener σ_λ^2 con el significado de distorsión media.

6.6.3 Rendimiento del diseño MMI-MVQ

Los resultados expuestos en el apartado anterior sugieren el uso de la siguiente expresión para P_s,

$$P_s(\Theta) = \frac{P(X|\theta_s)^{\beta/T}}{\sum_{l=1}^{L} P(X|\theta_l)^{\beta/T}} \tag{6.24}$$

donde el factor β ayuda a controlar la aproximación de $|\delta_{m,s} - P_s|$ al error empírico. En lo sucesivo se hará referencia a él como *factor de suavizado*. Con todas estas modificaciones, la estimación de parámetros resultante no es tipo MMI propiamente, sino que está más orientada al objetivo de minimización del error de test del sistema. Aún así se seguirá hablando de diseño MMI para los diccionarios MVQ.

En la figura 6.8 se muestran las curvas de evolución del error en función del

número de iteraciones del diseño MMI-MVQ (hasta 15) aplicadas en el procedimiento de entrenamiento descrito al comienzo del apartado, con la forma dada en la ecuación (6.24) para P_s. Las pruebas se han desarrollado para 4, 8, 16 y 32 centros por diccionario y para $\beta = 0.5, 1.0, 2.0$. Con 0 iteraciones se hace referencia a los sistemas originales MVQ. El parámetro de convergencia usado es $\eta = 0.2$.

En el caso de 4 centros, el método propuesto puede reducir el error a menos de la mitad, tal como se observa en la gráfica correspondiente. La mejor convergencia tiene lugar para $\beta = 2.0$, destacándose claramente a partir de 5 iteraciones. El valor $\beta = 0.5$ produce cierta disminución del error hasta 3 iteraciones, aumentándolo rápidamente después. En la gráfica de 8 centros, el valor $\beta = 1.0$ arroja resultados claramente superiores a los de $\beta = 2.0$ (especialmete a partir de 4 iteraciones). A diferencia del caso de 4 centros, $\beta = 0.5$ tiene un comportamiento aceptable, aunque todavía inferior a los otros dos. Para 16 centros, el valor $\beta = 0.5$ se consolida como una buena opción y produce una disminución más rápida que $\beta = 1.0$, aunque los resultados de ambas curvas son similares a partir de 10 iteraciones (presentando $\beta = 1.0$ un comportamiento más estable). La curva para $\beta = 2.0$ no aporta ya ninguna mejora. Finalmente, en el caso de 32 centros, se observa que $\beta = 2.0$ presenta ya un comportamiento inaceptable, mientras que $\beta = 0.5, 1.0$ tienden a disminuir lentamente el error ($\beta = 0.5$ es superior a partir de 10 iteraciones).

De todo lo anterior, se pueden extraer las siguientes conclusiones:

1) Los parámetros de los diccionarios VQ son apropiados para una estimación MMI, ayudando a la disminución del error de test del sistema, a diferencia de la insensibilidad que muestran los parámetros del modelo discreto.

2) Conforme se aumenta el número de centros es conveniente aumentar también el peso de los casi-errores de entrenamiento, disminuyendo el de los errores, lo cual se consigue disminuyendo el valor del factor de suavizado.

3) El método del diseño MMI-MVQ descrito produce reducciones del error de test para todos los tamaños de diccionario empleados, aunque se muestra tanto más efectivo cuanto menor es el tamaño. Esto guarda relación con la discusión ML/MMI realizada en el capítulo 3, según la cual el método MMI pierde efectividad conforme aumenta la calidad del modelado.

4) Es importante usar en cada situación un número apropiado de iteraciones, ya

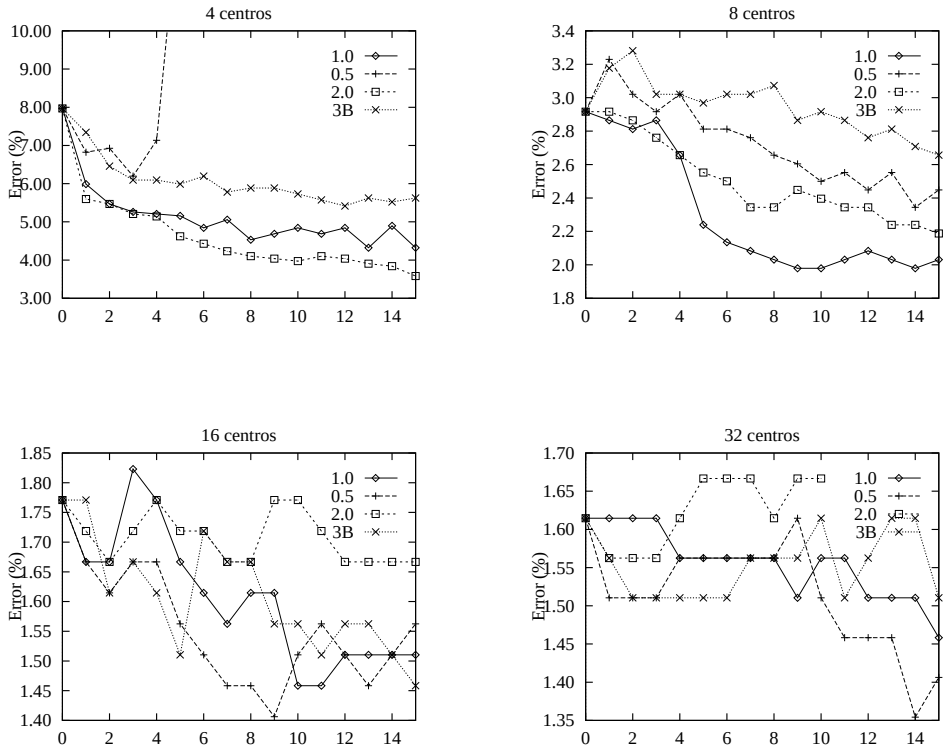


Figura 6.8: *Evolución del error de test para 4, 8, 16 y 32 centros y* $\beta = 0.5, 1.0, 2.0$ *en función del número de iteraciones (método MMI-MVQ).*

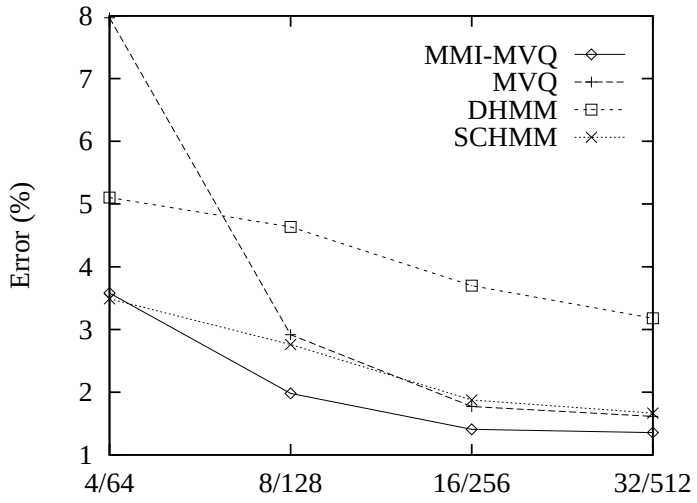


Figura 6.9: *Comparación de las técnicas de entrenamiento MMI-MVQ, MVQ, DHMM y SCHMM.*

que un número excesivamente corto puede no producir mejoras apreciables, mientras que demasiadas pueden llegar a degradar los modelos, de forma que ya no sean útiles para reconocimiento. En este sentido, también será crucial la selección de un valor apropiado del parámetro de convergencia η.

En la figura 6.9 se muestra una comparación de los mejores valores de error obtenidos en las curvas 6.8 con los experimentos realizados con MVQ, DHMM y SCHMM. Los resultados obtenidos por el método MMI-MVQ son siempre superiores a los demás, salvo para 4 centros (los SCHMM dan un error ligeramente inferior). Aparte de las mejoras en las tasas de error, otra observación, más importante, si cabe, debe ser hecha. Esta se refiere a la reducción de la complejidad computacional necesaria para obtener un sistema de alto rendimiento. Efectivamente, usando MMI-MVQ la tasa de error para 16 centros es muy similar a la obtenida con 32. Además, con 8 centros es posible una tasa de error en torno al 2%. Incluso, con sólo 4 centros para caracterizar los distintos sonidos que intervienen en una palabra, se puede obtener un error similar al obtenido con un sistema SCHMM de 64 centros, siendo el método MVQ computacionalmente más eficiente en reconocimiento.

6.7 Estimación MCE de diccionarios MVQ

De los apartados anteriores se deduce que el uso de un método discriminativo para el diseño de los diccionarios de un sistema MVQ puede reducir de una forma efectiva el error de test del sistema. Por esta razón se ha considerado conveniente la aplicación de una estimación MCE sobre dichos diccionarios, de una forma similar a la desarrollada para el método MMI. A diferencia del diseño MMI-MVQ, que usa un descenso en gradiente para realizar la maximización de la información mutua, se usará el algoritmo GPD, descrito en el capítulo 3, para la minimización del error de clasificación, con el objetivo de actuar sobre el valor esperado del error en lugar de directamente sobre el error, tal como se propone en [Juang92].

Como ya fue expuesto en el capítulo 3, el método se fundamenta sobre tres pasos. Para la estimación de diccionarios, siguiendo las pautas marcadas en los apartados anteriores, estos pasos son los siguientes:

1) Como funciones discriminantes se escogen las probabilidades de cuantización logarítmicas,

$$g_k(X,\Lambda) = g_k(X,\lambda_k) = \log P(X|O,\lambda_k) = \log P(X|\theta_k) \qquad (6.25)$$

2) Medidas de error:

$$d_k(X) = -g_k(X,\lambda_k) + \log\left[\frac{1}{L-1}\sum_{j\neq k} e^{\beta g_j(X,\lambda_j)}\right]^{1/\beta} \qquad (6.26)$$

3) Funciones de coste l_k tipo sigmoide. Se considera $\alpha = 1$ para simplificar el cálculo.

La minimización del Coste Esperado (expresión (3.48)) puede realizarse mediante la aplicación del teorema GPD (expresiones (3.50) y (3.51)) (en este caso los parámetros de Λ reestimados son sólo los de los diccionarios). Es, por tanto, fundamental la evaluación del gradiente $\bigtriangledown l_m(X,\Lambda)$ (donde X es de la clase W_m). Este gradiente incluye 2 tipos de derivadas, según se derive respecto a un parámetro del diccionario de la clase correcta W_m, o respecto a otro de una clase incorrecta W_s ($s \neq m$). En general, para la componente k de un centro $\mathbf{y}_j^s$, de cualquier

diccionario de parámetros θ_s, se tiene,

$$\frac{\partial l_m(X,\Lambda)}{\partial y_j^s(k)} = \frac{\partial l_m}{\partial d_m}\frac{\partial d_m}{\partial y_j^s(k)} \tag{6.27}$$

donde,

$$\frac{\partial l_m}{\partial d_m} = l_m(X,\Lambda)\left[1 - l_m(X,\Lambda)\right] \tag{6.28}$$

Para el caso $s = m$,

$$\frac{\partial d_m}{\partial y_j^m(k)} = -\frac{\partial g_m}{\partial y_j^m(k)} = -\frac{1}{\sigma_{\lambda_m}^2}\sum_{t=1}^{T}\left(x_t(k) - y_j^m(k)\right)\delta_{o_t,v_j} \tag{6.29}$$

Si $s \neq m$,

$$\frac{\partial d_m}{\partial y_j^s(k)} = \frac{e^{\beta g_s}}{\sum\limits_{l\neq m} e^{\beta g_l}}\frac{\partial g_s}{\partial y_j^s(k)} = \frac{e^{\beta g_s}}{\sum\limits_{l\neq m} e^{\beta g_l}}\frac{1}{\sigma_{\lambda_s}^2}\sum_{t=1}^{T}\left(x_t(k) - y_j^s(k)\right)\delta_{o_t,v_j} \tag{6.30}$$

Se pueden definir unos factores de corrección $\nu_m(X)$ y $\phi_{sm}(X)$ con expresiones análogas a las (3.60) y (3.61), por lo que las derivadas parciales para los centros de los diccionarios pueden escribirse como,

$$\frac{\partial l_m(X,\Lambda)}{\partial y_j^m(k)} = -\nu_m(X)\frac{\sum\limits_{t=1}^{T}\left(x_t(k) - y_j^m(k)\right)\delta_{o_t,v_j}}{\sigma_{\lambda_m}^2} \tag{6.31a}$$

$$\frac{\partial l_m(X,\Lambda)}{\partial y_j^s(k)} = \nu_m(X)\phi_{sm}(X)\frac{\sum\limits_{t=1}^{T}\left(x_t(k) - y_j^s(k)\right)\delta_{o_t,v_j}}{\sigma_{\lambda_s}^2} \qquad s \neq m \tag{6.31b}$$

También pueden obtenerse las derivadas parciales correspondientes a los parámetros σ_λ^2,

$$\frac{\partial l_m(X,\Lambda)}{\partial \sigma_{\lambda_m}^2} = -\nu_m(X)\frac{\partial g_m}{\partial \sigma_{\lambda_m}^2} \tag{6.32a}$$

$$\frac{\partial l_m(X,\Lambda)}{\partial \sigma_{\lambda_s}^2} = \nu_m(X)\phi_{sm}(X)\frac{\partial g_m}{\partial \sigma_{\lambda_s}^2} \qquad s \neq m \tag{6.32b}$$

donde , para cualquier clase de modelos λ se tiene,

$$\frac{\partial g_m}{\partial \sigma_\lambda^2} = \frac{T}{2\sigma_\lambda^2}\left[\frac{D_\lambda(X)}{\sigma_\lambda^2} - p\right] \tag{6.33}$$

donde $D_\lambda(X)$ responde a la expresión (6.8).

Lo mismo que sucedía para la estimación de modelos DHMM, hay una gran similitud entre los gradientes obtenidos por los métodos MMI y MCE para el diseño VQ. La única diferencia reside en los factores de corrección aplicados. También cabe destacar aquí los paralelismos existentes entre el método propuesto y el entrenamiento LVQ descrito en el capítulo 2, dada la secuencialidad de ambos, la similitud de los algoritmos de corrección, y el uso de una ventana de selección de casi-errores (en este caso representada por $\nu_m(X)$).

6.7.1 Rendimiento de la estimación MCE de diccionarios MVQ

En la implementación de la estimación MCE-GPD sobre diccionarios MVQ se ha considerado una matriz $\mathbf{U}$ unidad. El parámetro de convergencia ha sido tomado como $\eta = 0.01$ (constante) para una convergencia apropiada. Para la reestimación de centros se ha hecho uso de las expresiones (6.31). Dado que el método GPD es secuencial, calcular σ_λ^2 como distorsión media (al estilo de como se hizo para el caso MMI-MVQ) a la llegada de cada secuencia de entrenamiento sería altamente costoso en cuanto a cómputo, por lo que, en primera aproximación, no se realiza reestimación alguna de σ_λ^2 (se mantiene el valor original del diccionario LBG). Los factores de corrección ν_m y ϕ_{sm} son obtenidos a partir de funciones de discriminación normalizadas en duración (siguiendo la misma filosofía del diseño MMI-MVQ),

$$g_k(X, \lambda_k) = \frac{1}{T}\log P(X|O, \lambda_k) \tag{6.34}$$

donde la probabilidad logarítmica de cuantización responde a la expresión (6.7).

En la figura 6.10 puede observarse la evolución del error de test del método propuesto para 4, 8, 16 y 32 centros por diccionario y para $\beta = 2.0, 4.0, 8.0$ en función del número de iteraciones. En este caso cada iteración se refiere cada presentación completa del conjunto de entrenamiento. Con el fin de comparar con el método MMI-MVQ, las gráficas incorporan también el mejor resultado de este último método para cada uno de los tamaños de diccionario ($\beta = 2.0$ para 4 centros, $\beta = 1.0$ para 8 centros, y $\beta = 0.5$ para 16 y 32 centros). Puede observarse

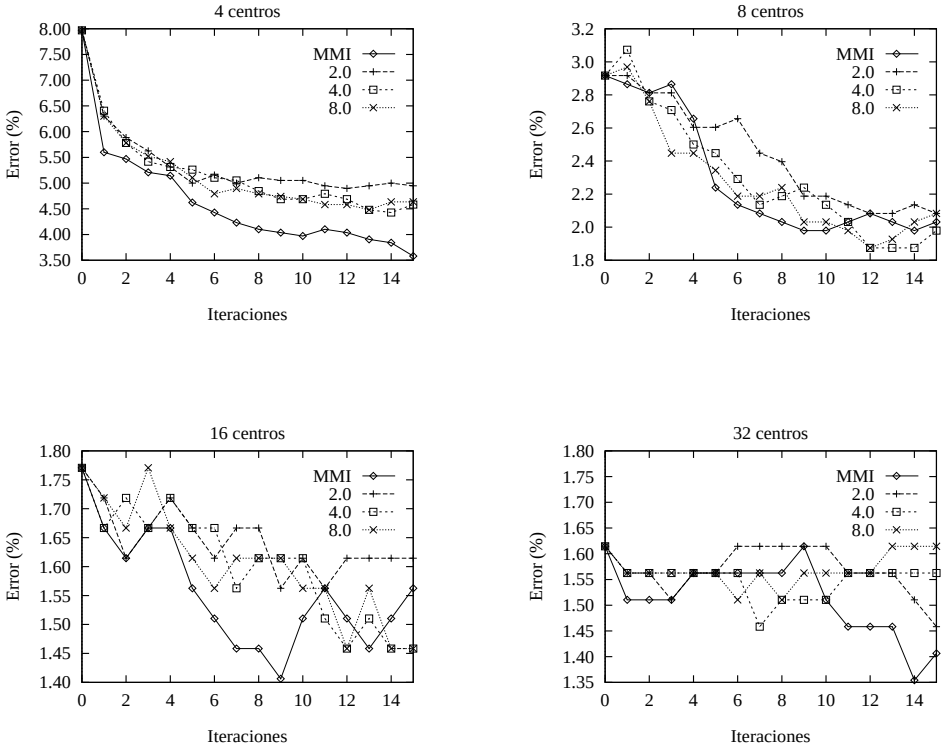


Figura 6.10: *Evolución del error de test para 4, 8, 16 y 32 centros y* $\beta = 2.0, 4.0, 8.0$ *en función del número de iteraciones en el método MCE-GPD, y comparación con los mejores casos del diseño MMI-MVQ.*

que el valor $\beta = 2.0$ produce siempre (salvo para 32 centros) los peores resultados. Por tanto, una mayor aproximación al error empírico, es decir, un mayor valor de β ($= 4.0, 8.0$), se adecúa mejor a la tarea de reconocimiento. Este resultado parece estar en contradicción con el obtenido en el método MMI-MVQ, para el que una disminución del factor de suavizado β parecía razonable conforme se aumentaba el tamaño del diccionario. Debe recordarse, sin embargo, que la disminución de dicho factor era el único mecanismo de introducción de casi-errores posible en dicho método, mientras que la estimación MCE dispone siempre de dicho mecanismo a través del factor $\nu_m(X)$.

Tomando el valor $\beta = 4.0$ como aceptable, se han realizado dos experiencias más que aparecen en las gráficas 6.11. Ambas aparecen comparadas con los diseños MMI-MVQ y MCE-GPD. La primera (notada como DG) consiste en la aplicación de un descenso en gradiente común al coste empírico promedio de la expresión (3.47). Al utilizarse un método de optimización similar al del MMI-MVQ, los parámetros σ_λ^2 pueden reestimarse fácilmente como distorsiones medias en cada iteración. La experiencia notada como GPDR pretende ser un refinamiento a la MCE-GPD (en las curvas aparece notada simplemente como GPD) estimando σ_λ^2 como distorsión media del diccionario correspondiente después de cada presentación completa de todas las secuencias de entrenamiento (una iteración). No se observan diferencias apreciables con el método GDP originalmente aplicado. Dos conclusiones son rápidamente extraídas de estos resultados:

1) No hay gran diferencia entre la minimización del coste empírico promedio y del coste esperado, como se desprende del hecho de que las curvas GPD y DG presenten resultados similares.

2) El método MCE no produce desviaciones importantes en la composición óptima de probabilidades de cuantización y de generación, como se deduce del hecho de que GPD, DG y GPDR reestimen los parámetros σ_λ^2 de distintas formas y arrojen todos ellos resultados similares.

Finalmente, hay que señalar que, aunque el método MCE está mejor adaptado a la disminución del error del sistema que el MMI-MVQ, no se ha conseguido (salvo quizás para 8 centros) una mejora apreciable sobre este último.

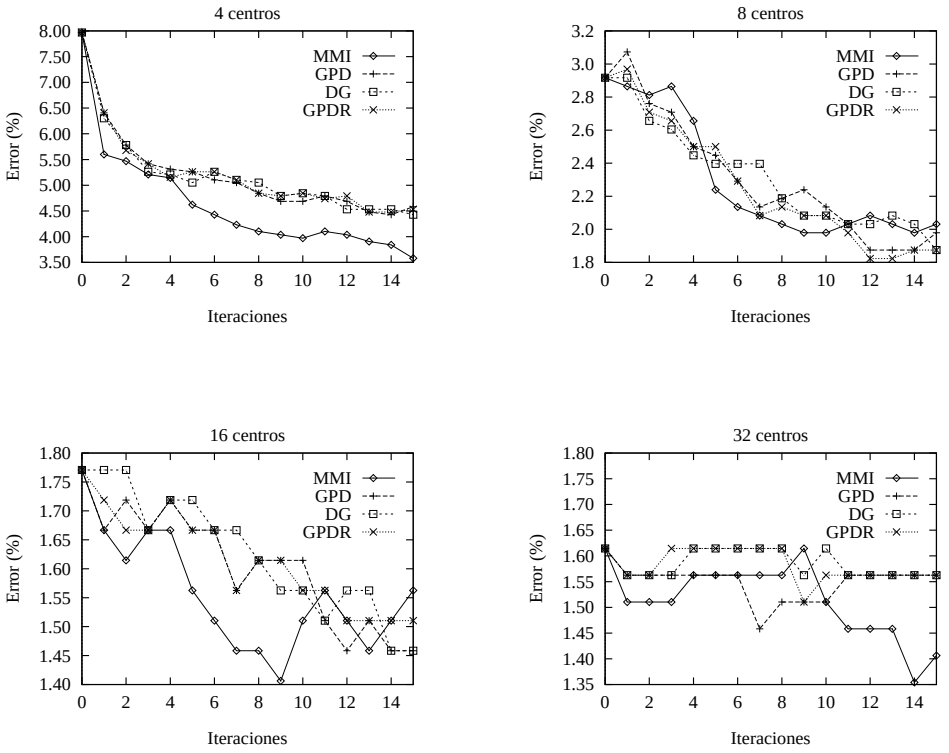


Figura 6.11: *Evolución del error de test para 4, 8, 16 y 32 centros en función del número de iteraciones para las experiencias MMI, GPD, DG y GPDR (las tres últimas con* $\beta = 4.0$*.*

#Centros	DHMM	SCHMM	MVQ	MMI-MVQ	MCE-GPD
64/4	5.10	3.48	7.96	3.58	4.42
128/8	4.63	2.76	2.91	1.97	1.87
256/16	3.69	1.87	1.77	1.40	1.45
512/32	3.17	1.66	1.61	1.35	1.45

Tabla 6.1: *Error de test para los modelados DHMM, SCHMM, MVQ con estimación ML y MVQ con diseños VQ tipo MMI-MVQ y MCE-GPD.*

6.8 Conclusión sobre diseño VQ discriminativo

Los resultados obtenidos en los apartados precedentes apoyan la adecuación de la técnica de Cuantización Vectorial Múltiple con entrenamiento discriminativo a la tarea de reconocimiento. En la tabla 6.1 se muestra un resumen comparativo de los resultados obtenidos. Los valores de error para DHMM, SCHMM, MVQ y MMI-MVQ corresponden a los de la gráfica 6.9. Los valores del método MCE-GPD han sido seleccionados de las curvas 6.10 con $\beta = 4.0$, para un número suficiente de iteraciones. Se observa cómo el uso de diccionarios discriminativos puede aproximar el rendimiento de un sistema MVQ al de un SCHMM en el caso de 4 centros, y proporcionar los mejores resultados para 8, 16 y 32 centros.

Los métodos empleados en este capítulo sugieren el uso de un algoritmo de corrección de centros VQ que, en su forma más general, tendría la siguiente expresión,

$$\mathbf{y}_k = \mathbf{y}_k + \eta f(X, \Lambda) \sum_{t=1}^{T} (\mathbf{x}_t - \mathbf{y}_k)\, \delta_{o_t, v_k} \tag{6.35}$$

donde $X = \mathbf{x}_1, \ldots, \mathbf{x}_T$ es una secuencia de entrenamiento y $\mathbf{y}_k$ es un centro del diccionario θ asociado a la clase W. El factor $f(X, \Lambda)$ y el parámetro η deben seguir las siguientes pautas:

1) $f(X, \Lambda)$ es positiva si $X \in W$ y negativa en caso contrario, con el objetivo de aproximar los centros a vectores de secuencias correctas y alejarlos de las incorrectas.

2) $|f(X, \Lambda)|$ debe depender del grado de confusión existente en la clasificación de X y debe aumentar la importancia de los casi-errores conforme aumenta el tamaño del diccionario empleado.

3) El factor η es un número real positivo cuyo valor debe escogerse apropiadamente (suficientemente pequeño) para una correcta convergencia.

El algoritmo (6.35) propone, en realidad, toda una gama de métodos de diseño VQ discriminativo basados en la selección de una función $f(X, \Lambda)$ apropiada.

Capítulo 7

MODELADO MVQ SEMICONTÍNUO

En este capítulo se presenta una nueva variante de modelado de Markov que incorpora características de los modelos SCHMM y MVQ, que han demostrado mejorar el rendimiento de los sistemas de reconocimiento. Así surge un modelo híbrido que será denominado SCMVQ. Previamente a su introducción, se realizará un estudio sobre la estimación ML de todos los parámetros que componen un modelo HMM semicontinuo. Posteriormente, se describirán los detalles de implementación de modelos HMM tipo SCMVQ y se realizará una comparación de su rendimiento con el de los SCHMM. Se finalizará con el estudio de la aplicación de diccionarios entrenados de forma discriminativa, como se vio en el capítulo anterior, al modelado SCMVQ.

7.1 Estimación ML de modelos semicontinuos

Como ya se vió en el capítulo 2, la diferencia fundamental entre los modelos SCHMM y DHMM es el uso de varios candidatos de cuantización, lo cual requiere pasar de probabilidades de producción $b_i(o)$ a densidades del tipo,

$$b_i(\mathbf{x}) = \sum_{v_k \in V} P(\mathbf{x}|v_k) b_i(v_k) \simeq \sum_{k=1}^{C} P(\mathbf{x}|v_k) b_i(v_k) \tag{7.1}$$

donde se ha aproximado la suma extendida a todos los centros del diccionario V por otra limitada sólo a los C mejores candidatos (con $P(\mathbf{x}|v_k)$ mayor). Las densidades de probabilidad se suponen gausianas con matriz de covarianza diagonal, por lo que pueden escribirse como,

$$P(\mathbf{x}|v_k) = \prod_{i=1}^{p} \frac{1}{\sqrt{2\pi\sigma_{ki}^2}} \exp\left\{ -\frac{1}{2} \frac{(x(i) - y_k(i))^2}{\sigma_{ki}^2} \right\} \tag{7.2}$$

De manera similar que para modelos DHMM se pueden definir probabilidades adelante y atrás,

$$\alpha_t(i) = P(\mathbf{x}_1 \mathbf{x}_2 \cdots \mathbf{x}_t, q_t = s_i|\lambda) \tag{7.3}$$

$$\beta_t(i) = P(\mathbf{x}_{t+1} \mathbf{x}_{t+2} \cdots \mathbf{x}_T | q_t = s_i, \lambda) \tag{7.4}$$

que se calculan con el mismo procedimiento iterativo descrito en el capítulo segundo. También se definen las probabilidades,

$$\xi_t(i,j) = P(q_t = s_i, q_{t+1} = s_j|X, \lambda) = \frac{\alpha_t(i) a_{ij} b_j(\mathbf{x}_{t+1}) \beta_{t+1}(j)}{P(X|\lambda)} \tag{7.5a}$$

$$\gamma_t(i) = P(q_t = s_i|X, \lambda) = \sum_{j=1}^{N} \xi_t(i,j) = \frac{\alpha_t(i)\beta_t(j)}{P(X|\lambda)} \tag{7.5b}$$

$$\begin{aligned} \mathcal{X}_t(i,j,k) &= P(q_{t-1} = s_i, q_t = s_j, o_t = v_k|X, \lambda) = \\ &\quad \frac{\alpha_{t-1}(i) a_{ij} P(\mathbf{x}_t|v_k) b_j(v_k) \beta_t(j)}{P(X|\lambda)} \end{aligned} \tag{7.5c}$$

$$\mathcal{F}_t(i,k) = P(q_t = s_i, o_t = v_k|X, \lambda) = \sum_{s=1}^{N} \mathcal{X}_t(s,i,k) =$$

$$\frac{\left[\sum_{s=1}^{N} \alpha_{t-1}(i) a_{si}\right] P(\mathbf{x}_t|v_k) b_i(v_k) \beta_t(i)}{P(X|\lambda)} \tag{7.5d}$$

$$\mathcal{S}_t(k) = P(o_t = v_k|X, \lambda) = \sum_{i=1}^{N} \mathcal{F}_t(i,k) \tag{7.5e}$$

Además se verifican las relaciones,

$$\xi_t(i,j) = \sum_{k=1}^{M} \mathcal{X}_t(i,j,k) \tag{7.6}$$

$$\gamma_t(i) = \sum_{s=1}^{N}\sum_{k=1}^{M} \xi_t(s,i,k) = \sum_{k=1}^{M} \mathcal{F}_t(i,k) \tag{7.7}$$

Las fórmulas para la reestimación de las matrices(Π, A, B) se obtienen de forma similar a la desarrollada en el capítulo 2 para modelos discretos. Mientras que las expresiones resultantes son idénticas a las (3.11) para las matrices Π y A, para las matrices de producción B se ve ligeramente modificada como,

$$\hat{b}_j(v_k) = \frac{\sum_{l=1}^{S}\sum_{t=1}^{T^l}\sum_{s=1}^{N} \mathcal{X}_t^l(s,j,k)}{\sum_{l=1}^{S}\sum_{t=1}^{T^l} \gamma_t^l(j)} \tag{7.8}$$

Para la implementación práctica de las fórmulas de reestimación de A y B es conveniente usar las siguientes expresiones,

$$\hat{a}_{ij} = \frac{\sum_{l=1}^{S}\sum_{t=1}^{T^l-1} \tilde{\alpha}_t^l(i) a_{ij} b_j(o_{t+1}^l) \tilde{\beta}_{t+1}^l(j)}{\sum_{l=1}^{S}\sum_{t=1}^{T^l-1}\sum_{k=1}^{M} \mathcal{F}_t^l(i,k)} \tag{7.9a}$$

$$\hat{b}_j(k) \frac{\sum_{l=1}^{S}\sum_{t=1}^{T^l} \mathcal{F}_t^l(j,k)}{\sum_{l=1}^{S}\sum_{t=1}^{T^l}\sum_{k=1}^{M} \mathcal{F}_t^l(j,k)} \tag{7.9b}$$

con

$$\mathcal{F}_t^l(i,k) = \left[\sum_{s=1}^{N} \tilde{\alpha}_{t-1}(i) a_{si}\right] P(\mathbf{x}_t|v_k) b_i(v_k) \tilde{\beta}_t(i) \tag{7.10}$$

Una diferencia fundamental con el modelado DHMM es que los parámetros del diccionario universal, centros y matrices de covarianza, también deben ser reestimados para que los modelos sean realmente óptimos desde el punto de vista de máxima semejanza [Huang89]. Por ejemplo, para reestimar la componente i del centro $\mathbf{y}_k$, $y_k(i)$, entrenando con una única secuencia $X = \mathbf{x}_1\mathbf{x}_2\cdots\mathbf{x}_T$, se procederá de la siguiente manera,

$$\begin{aligned}
\frac{\partial P(X|\lambda)}{\partial y_k(i)} &= \sum_t \sum_r \frac{\partial P(X|\lambda)}{\partial \alpha_t(r)} \frac{\partial \alpha_t(r)}{\partial y_k(i)} = \\
&\sum_t \sum_r \beta_t(r) \left[\sum_s \alpha_{t-1}(s) a_{sr}\right] \frac{\partial b_r(\mathbf{x}_t)}{\partial y_k(i)} = \\
&\sum_t \frac{x_t(k) - y_k(i)}{\sigma_{ki}^2} \sum_r \sum_s \alpha_{t-1}(s) a_{sr} P(\mathbf{x}_t|v_k) b_r(v_k) \beta_t(r) = \\
&P(X|\lambda) \sum_t \frac{x_t(k) - y_k(i)}{\sigma_{ki}^2} \mathcal{S}_t(k) = 0
\end{aligned} \tag{7.11}$$

Despejando y generalizando para múltiples secuencias de entrenamiento, se obtienen las siguientes fórmulas de reestimación,

$$\hat{y}_k(i) = \frac{\displaystyle\sum_{X\in\mathcal{T}} \sum_{t=1}^{T^X} \mathcal{S}_t^X(k) x_t^X(i)}{\displaystyle\sum_{X\in\mathcal{T}} \sum_{t=1}^{T^X} \mathcal{S}_t^X(k)} \tag{7.12a}$$

$$\hat{\sigma}_{ki}^2 = \frac{\displaystyle\sum_{X\in\mathcal{T}} \sum_{t=1}^{T^X} \mathcal{S}_t^X(k) \left(x_t^X(i) - \hat{y}_k(i)\right)^2}{\displaystyle\sum_{X\in\mathcal{T}} \sum_{t=1}^{T^X} \mathcal{S}_t^X(k)} \tag{7.12b}$$

donde queda patente que los parámetros del diccionario se reestiman conjuntamente con los de las matrices (Π, A, B). Obsérvese que las sumatorias se ha extendido al conjunto $\mathcal{T}$ que incluye todas las secuencias de entrenamiento, en lugar de restringirse al conjunto T (de cardinal S) que incluye sólo secuencias de la clase

que está siendo entrenada (como se hace para las matrices (Π, A, B)), ya que se trata de un diccionario universal.

7.2 Resultados experimentales con modelos SCHMM

Se han experimentado tres tipos distintos de modelos SCHMM:

REF Es el mismo sistema implementado en el capítulo 5, que reestima únicamente las matrices A y B mediante las expresiones (7.9).

TOT Se utiliza la reestimación conjunta dada por las expresiones (7.9) y (7.12).

SIN Es similar a TOT, pero no se reestiman las matrices de covarianza. Según Huang [Huang90], la no reestimación de estas matrices conduce a un mayor rendimiento del sistema, lo que indica que los parámetros acústicos no pueden ser correctamente modelados por la *fdp* empleada.

En la figura 7.1 se comparan los resultados de error obtenidos para los tres tipos de sistemas descritos, para 64, 128, 252 y 516 centros, y usando 4 candidatos (valores numéricos en la tabla 7.1). Tal como sugiere Huang en [Huang90], la superioridad de SIN sobre TOT, en todos los casos, indica la conveniencia de no reestimar las matrices de covarianza. También se comprueba que el sistema REF presenta resultados similares a SIN para 64 y 128 centros (ligeramente superiores para 64 centros y ligeramente inferiores para 128), pero más claramente superiores para 256 y 512 centros. La explicación a este comportamiento puede buscarse en dos problemas apuntados ya en el capítulo anterior (apartado 6.2): la no coherencia entre la medida de probabilidad y la distancia pesada empleada, y el problema de la reestimación de un gran número de parámetros al usar matrices de covarianza diagonales completas. Todos estos resultados confirman la dificultad, detectada por Huang, de modelar los parámetros acústicos con las *fdp*s gausianas multivariadas empleadas.

7.3 Modelos de Markov tipo SCMVQ

En el capítulo 2 se estableció que, en la forma más general, las densidades $b_i(\mathbf{x})$ debían escribirse como,

$$b_i(\mathbf{x}) = P(\mathbf{x}|s_i, \lambda) = \sum_{v_k \in V(s_i,\lambda)} P(\mathbf{x}|v_k, s_i, \lambda) P(v_k|s_i, \lambda) \tag{7.13}$$

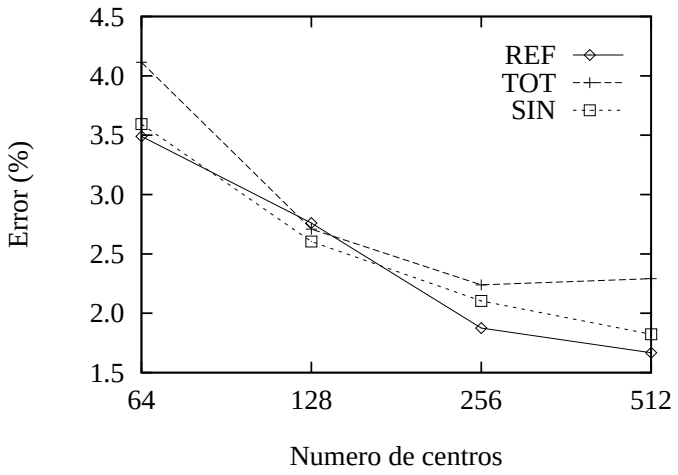


Figura 7.1: *Comparación de sistemas SCHMM entrenados con y sin reestimación conjunta.*

#Centros	REF	TOT	SIN
64/4	3.48	4.11	3.59
128/8	2.76	2.70	2.60
256/16	1.87	2.23	2.10
512/32	1.66	2.29	1.82

Tabla 7.1: *Error de test para los sistemas REF, TOT y SIN.*

que es el caso CHMM. Este tipo de modelado presenta, como ya se señaló en los capítulos 1 y 2, varios problemas relacionados con su entrenamiento: complejidad de cálculo, necesidad de una gran cantidad de datos de entrenamiento, etc., de donde surge el interés por modelos HMM simplificados:

SCHMM Para todo estado s_i de cualquier modelo λ, $V(s_i, \lambda) = V$.

DHMM Similar al anterior pero con únicamente el mejor candidato de cuantización.

MVQ Para todo estado s_i de un cierto modelo λ, $V(s_i, \lambda) = V(\lambda)$, conservando únicamente el mejor candidato de cuantización.

En los capítulos precedentes se ha comprobado la potencia y posibilidades discriminativas de los modelos MVQ frente a las otras dos variantes, además de aportar un importante ahorro en la cantidad de cálculo (dependiendo si se trata de entrenamiento o test). Estos resultados han sugerido la implementación de un nuevo tipo de modelado que generalice el modelado MVQ para varios candidatos de cuantización, en la misma manera que el modelado SCHMM generaliza al DHMM. Esta nueva variante será denominada *Modelado HMM Semicontinuo con Cuantización Vectorial Múltiple* (SCMVQHMM o, simplemente, SCMVQ). La expresión de las densidades $b_i(\mathbf{x})$ se establece de la siguiente manera,

$$b_i(\mathbf{x}) = P(\mathbf{x}|s_i, \lambda) = \sum_{v_k \in V(\lambda)} P(\mathbf{x}|v_k, \lambda) P(v_k|s_i, \lambda) \simeq \sum_{k=1}^{C} P(\mathbf{x}|v_k, \lambda) b_i(v_k) \tag{7.14}$$

Aunque el modelado SCMVQ queda formalmente descrito por la expresión (7.14), queda por determinar la forma de las densidades $P(\mathbf{x}|v_k, \lambda)$. Dado que esta nueva variante pretende ser una generalización del modelado MVQ, es decir, que coincida con el caso MVQ para 1 sólo candidato de cuantización, las densidades deben tener una forma análoga a la dada por la expresión (6.5). Sin embargo, no debe olvidarse la introducción de un peso α para las probabilidades de cuantización en la expresión (6.10). De aquí que deban emplearse densidades con la siguiente expresión,

$$P(\mathbf{x}|o_k, \lambda) = (2\pi\sigma_\lambda^2)^{-p\alpha/2} \exp\left\{-\frac{\alpha}{2\sigma_\lambda^2}\|\mathbf{x} - \mathbf{y}_k\|^2\right\} \tag{7.15}$$

que es una función densidad no normalizada, debido a la introducción del factor α (este factor quedó fijado en el capítulo anterior a $\alpha = 0.538$).

En cuanto a la estimación de los parámetros de los modelos SCMVQ, pueden emplearse las mismas fórmulas de reestimación (7.9) para las matrices Π, A y B, teniendo en cuenta que cada modelo usa las cuantizaciones obtenidas de su propio diccionario. Respecto a la estimación de los parámetros de los diccionarios, también se procede de forma similar al caso SCHMM, pero ahora cada diccionario es entrenado exclusivamente con el conjunto T de secuencias de su clase, por lo que utilizando las densidades (7.15), se tienen las siguientes fórmulas de reestimación,

$$\hat{y}_k(i) = \frac{\sum_{l=1}^{S}\sum_{t=1}^{T^l} \mathcal{S}_t^l(k) x_t^l(i)}{\sum_{l=1}^{S}\sum_{t=1}^{T^l} \mathcal{S}_t^l(k)} \tag{7.16a}$$

$$\hat{\sigma}_\lambda^2 = \frac{1}{p} \frac{\sum_{l=1}^{S}\sum_{t=1}^{T^l}\sum_{k=1}^{M} \mathcal{S}_t^l(k) \|x_t^l - \hat{y}_k\|^2}{\sum_{l=1}^{S}\sum_{t=1}^{T^l}\sum_{k=1}^{M} \mathcal{S}_t^l(k)} \tag{7.16b}$$

Como punto de partida para realizar esta reestimación, pueden usarse los diccionarios LBG empleados en los modelos MVQ.

Es fácil de comprobar que en el caso de 1 sólo candidato de cuantización,

$$\mathcal{S}_t(k) = \delta_{o_t, v_k} \tag{7.17}$$

por lo que las fórmulas de reestimación ya no dependen de los parámetros (Π, A, B), y coinciden con las de cálculo del centroide del algoritmo LBG. Dado que dicho algoritmo ya ha sido llevado hasta la convergencia previamente, para usarlo como diccionario inicial, no es necesario iterarlo más, y se puede omitir la reestimación de los parámetros del diccionario mediante las ecuaciones (7.16). De esta forma, el método SCMVQ queda reducido al MVQ en el caso de un sólo candidato de cuantización. Además, este resultado ratifica la coincidencia del método de entrenamiento, expuesto en el capítulo anterior, para los modelos MVQ con una estimación ML (ver apartado 6.1).

Entre todas las variantes propuestas, el modelado SCMVQ es el más próximo formalmente al modelado continuo CHMM, ya que la única variación respecto a este es la de hacer que todos los estados del modelo compartan el mismo conjunto

No. Cands.	1	2	3	4	5	6	7	8
8C EXP1	2.91	3.02	3.33	3.07	3.17	3.22	3.22	3.22
8C EXP2	-	2.70	2.96	2.81	2.81	2.81	2.81	2.81
16C EXP1	1.77	1.56	1.45	1.51	1.56	1.61	1.61	1.56
16C EXP2	-	1.45	1.51	1.56	1.61	1.66	1.61	1.61
32C EXP1	1.61	1.25	1.40	1.35	1.40	1.40	1.45	1.45
32C EXP2	-	1.25	1.14	1.09	1.09	1.09	1.19	1.14

Tabla 7.2: *Valores de error para modelado SCMVQ con 1-8 candidatos para 8, 16 y 32 centros por diccionarios. Experiencias EXP1 y EXP2.*

de densidades (diccionario VQ). A pesar de esta proximidad al modelado CHMM, si se considera (lo mismo que en el capítulo anterior) que el tamaño de los diccionarios MVQ multiplicado por el número de modelos (esto es el número total de centros entre todos los diccionarios) es igual al tamaño del diccionario universal usado en SCHMM, resulta que el modelado SCMVQ tiene una complejidad en reconocimiento similar al SCHMM, pero, lo mismo que ocurría con el MVQ frente al DHMM, presenta un sensible ahorro computacional en entrenamiento por la simplificación que supone el uso de diccionarios pequeños, tanto en el propio entrenamiento de los diccionarios VQ como en la posterior estimación de las matrices Π, A y B.

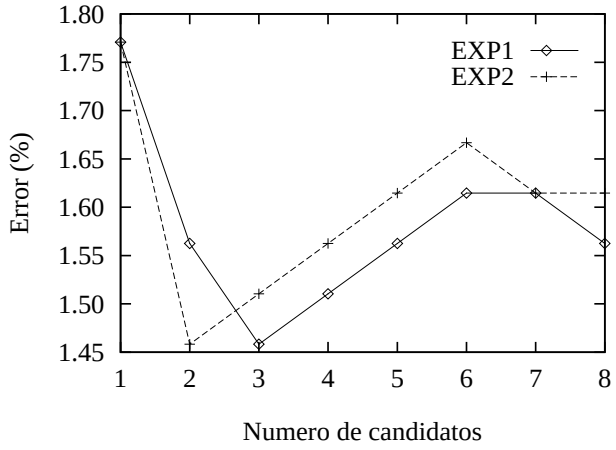
7.4 Resultados Experimentales con modelos SCMVQ

Se han llevado a cabo 2 tipos de experiencias de reconocimiento con los modelos SCMVQ descritos en el apartado anterior:

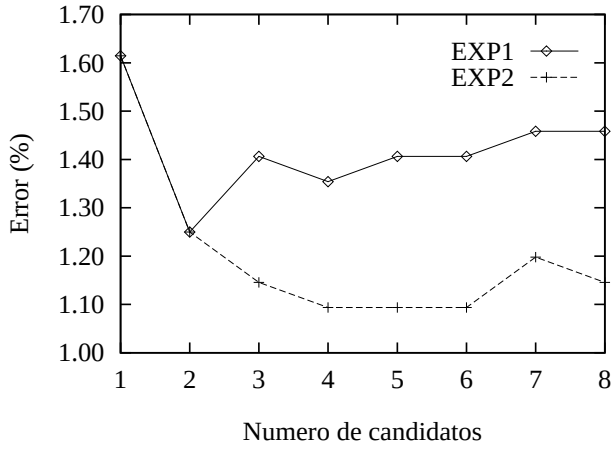
EXP1 Se reestiman únicamente las matrices A y B (inicializadas con segmentación lineal) con las expresiones dadas en (7.9), y se usan los mismos diccionarios LBG utilizados en los modelos MVQ.

EXP2 Es similar a la anterior, pero incluyendo la reestimación de los parámetros de los diccionarios (fórmulas (7.16)) (inicializados con los diccionarios LBG del caso anterior). Es decir, se realiza una estimación ML completa de todos los parámetros del sistema.

Las experiencias se han llevado a cabo para 8, 16 y 32 centros (no se considera el caso de 4 centros por no proporcionar una densidad de centros suficientemente



(a) *16 centros por diccionario*



(b) *32 centros por diccionario*

Figura 7.2: *Variación del error con el número de candidatos de cuantización para 16 y 32 centros por diccionario en un sistema SCMVQ.*

alta en el espacio de representación). Los resultados de error para un número de candidatos desde 1 a 8 aparecen en la tabla 7.2. El caso de un sólo candidato corresponde exactamente al modelado MVQ. El efecto de la inclusión de un mayor número de candidatos de cuantización depende del tamaño del diccionario:

1) En el caso de 8 centros (8C), la inclusión de más candidatos degrada claramente el sistema en la experiencia EXP1. La aplicación de EXP2 puede aportar una pequeña mejora de 0.2% para 2 candidatos, degradándose o estabilizándose el error para más candidatos.

2) Para 16 centros (16C) (ver figura 7.2.a), EXP1 reduce de una forma efectiva el error para 2-4 candidatos. La inclusión de más candidatos puede empeorar el rendimiento, aunque obteniéndose siempre resultados mejores que con un sólo candidato (sistema MVQ). EXP2 ofrece un comportamiento similar a EXP1.

3) Para 32 centros (32C) (ver figura 7.2.b), EXP1 presenta presenta de nuevo un mínimo de error, en este caso para 2 candidatos. Pero, además, ahora se observa que la reestimación de los diccionarios (EXP2) contribuye a reducir aún más el error del sistema.

De todo lo anterior se extraen 2 conclusiones:

a) Que los modelos SCMVQ necesitan una densidad suficientemente alta de centros en el espacio de representación para aportar mejoras significativas. Sólo la existencia de varios centros próximos a un vector de entrada dado puede contribuir a una mejor representación de dicho vector.

b) Es importante "enseñar" al sistema a que también pueden contribuir otros centros, distintos del vecino más próximo para representar a un vector dado mediante un entrenamiento conjunto de las matrices A y B y los diccionarios VQ (EXP2). Así, es posible impedir una degradación excesiva en el caso de 8 centros, y obtener mejoras importantes si la concentración de centros es alta, como en el caso de 32 centros.

Por último, en la tabla 7.3 se comparan los resultados obtenidos con modelos DHMM (estimación ML), MVQ (estimación ML), SCHMM (sólo estimación ML para las matrices A y B, sistema REF) y SCMVQ (estimación ML completa). En el caso de modelos SCMVQ, se muestran los resultados para 2 candidatos de

#Centros	DHMM	SCHMM	MVQ	SCMVQ	SCHMM1
64/4	5.10	3.48	7.96	-	4.01
128/8	4.63	2.76	2.91	2.70	2.34
256/16	3.69	1.87	1.77	1.45	1.87
512/32	3.17	1.66	1.61	1.09	1.40

Tabla 7.3: *Error de test para los modelados DHMM, SCHMM, MVQ y SCMVQ.*

cuantización para 8 y 16 centros, y 4 candidatos para 32 centros. La complejidad de cálculo en reconocimiento será siempre inferior para los SCMVQ, debido a la simplificación introducida en las matrices de covarianza de los centros, respecto a los SCHMM. Esta reducción es aún más notoria para los casos de 8 y 16 centros, en los que se usan sólo 2 candidatos de cuantización, en lugar de los 4 que usan los SCHMM. Como puede comprobarse, los modelos MVQ y SCMVQ son siempre superiores cuando se usan 16 y 32 centros por diccionario. Los SCMVQ arrojan, además, un resultado ligeramente superior a los SCHMM para 8 centros (no así si se usan los SCHMM tipo TOT o SIN). Finalmente, se ha realizado la experiencia referenciada como SCHMM1 (última columna de la tabla 7.3), en la que se ha desarrollado un sistema tipo SCHMM, con reestimación conjunta ML completa, usando las mismas *fdp*s (expresión (7.15)) que para los modelos SCMVQ. Es interesante observar que los modelos SCHMM1 pueden obtener resultados iguales o superiores a los SCHMM estandard (que usan gausianas multivariadas), aunque solo superiores a los SCMVQ para 8/128 centros. Este resultados ratifican el potencial que encierra el uso de *fdp*s similares a las de (7.15).

Capítulo 8

CONCLUSIONES Y TRABAJO FUTURO

En este capítulo se hace un repaso de los contenidos expuestos en los capítulos anteriores, extrayendo las principales conclusiones de los mismos. También se comentan las perspectivas y líneas de trabajo futuras que abre este trabajo, así como las posibilidades de aplicación en el ámbito del reconocimiento de voz continua.

8.1 Conclusiones

1) La inclusión de información sobre la energía de la señal contribuye a una reducción del error del sistema. En concreto, la derivada de la energía respecto al tiempo, o delta energía, ha mostrado ser más efectiva que la energía. La comparación de los vectores formados por las características seleccionadas puede llevarse a cabo mediante una Distancia Multicaracterística Pesada (DMP) que no incrementa la complejidad de los procesos posteriores.

2) Los pesos de la distancia DMP pueden considerarse universales siempre que las condiciones de "clustering" no varíen drásticamente.

3) La estimación MMI de modelos DHMM, aún disminuyendo el error de entrenamiento, no proporciona mejoras apreciables sobre el error de test del sistema estudiado en el presente trabajo. Este comportamiento es achacable a la eliminación de información discriminativa en el proceso VQ.

4) La estimación ML de modelos MVQ conduce a dos procesos independientes de entrenamiento: un entrenamiento LBG para la etapa VQ y una estimación Baum-Welch de la etapa HMM discreta.

5) Los problemas de entrenamiento insuficiente y la coherencia con la distancia DMP aconsejan el uso de matrices de covarianza, en las densidades de probabilidad asociadas a los centros, diagonales, y con todos los elementos de la diagonal principal iguales a la distorsión promedio por característica del diccionario considerado, para el modelado MVQ-HMM. El resultado es que a la puntuación clásica obtenida de un modelo HMM discreto, hay que añadir otra nueva relacionada con la distorsión resultante en el diccionario correspondiente. La composición óptima de ambas puntuaciones puede obtenerse de forma experimental. Los resultados obtenidos indican que el modelado MVQ puede superar a los modelados DHMM y SCHMM en cuanto a tasas de reconocimiento. Respecto a la cantidad de cómputo requerida, los modelos MVQ son siempre menos costosos en entrenamiento. En reconocimiento, igualan a los DHMM y mejoran a los SCHMM.

6) A partir de una estimación MMI puede obtenerse un método discriminativo de diseño de diccionarios MVQ totalmente independiente del proceso HMM discreto. Este diseño tipo MMI conduce a una reducción efectiva del

error de test del sistema (a diferencia de lo que ocurría en el caso DHMM), proporcionando resultados claramente superiores a todos los considerados anteriormente. Esta conclusión está en consonancia con la afirmación (conclusión 3) de que la información discriminativa debe ser utilizada antes del proceso VQ.

7) La efectividad del diseño VQ tipo MMI es menor conforme aumenta la bondad del modelado empleado, en nuestro caso, cuando se aumenta el número de centros. Este efecto puede ser aprovechado para obtener sistemas de alto rendimiento con una menor complejidad computacional (reduciendo el número de centros).

8) El método MCE-GPD, especialmente ideado para la disminución del error, proporciona un mecanismo similar al MMI para el diseño discriminativo de diccionarios MVQ, aunque no se ha podido probar su superioridad respecto al segundo.

9) La Cuantización Vectorial Múltiple y el modelado HMM semicontinuo pueden ser combinados para generar un nuevo tipo de modelado que hemos denominado SCMVQ. Las mejoras que este método aporta quedan condicionadas a una densidad suficiente de centros en el espacio de representación, y a una reestimación conjunta con los parámetros VQ (a diferencia de la independencia de estimaciones ML del modelado MVQ). De hecho, este nuevo tipo de modelado ha presentado la tasa de error de reconocimiento más baja encontrada sobre la base de datos de trabajo (1.09%).

8.2 Líneas Futuras de Trabajo

Los buenos resultados obtenidos de la aplicación de las estimaciones MMI y MCE al diseño de diccionarios MVQ ponen de manifiesto la necesidad de un estudio en profundidad sobre el entrenamiento competitivo de diccionarios. Consideramos que un buen punto de partida es el algoritmo de corrección señalado en (6.35). Esta expresión contiene toda una familia de métodos de diseño VQ discriminativos generada por las posibles funciones de peso $f(X, \Lambda)$. Éstas podrán derivarse a

partir de una función criterio (como en los casos MMI o MCE) o de una forma empírica, al estilo de los algoritmos LVQ.

El presente trabajo deja también abierta la puerta para un entrenamiento discriminativo de diccionarios en el caso de modelos SCMVQ. Como ya se señaló en el capítulo 7, será necesario "enseñar" a los vectores de entrada qué centros deben usar para su representación, pero ahora de una forma discriminativa. Una solución directa sería proceder a una reestimación conjunta según los criterios MMI o MCE. Un claro inconveniente sería la ingente cantidad de cálculo necesaria para la misma. Sería, por tanto, interesante buscar alguna forma de independizar de nuevo los procesos VQ y HMM discreto.

Por supuesto, queda también pendiente la aplicación del trabajo aquí desarrollado a una tarea de voz continua. Aunque todas las técnicas desarrolladas tienen inmediata traslación a sistemas de voz continua, consideramos interesante remarcar, suponiendo un sistema basado en modelos MVQ y unidades inferiores a la palabra, las siguientes observaciones: a) las unidades inferiores a la palabra son altamente confundibles entre sí, por lo que un diseño discriminativo de sus diccionarios debe contribuir de forma efectiva a la reducción de la tasa de error del sistema, y b) el diccionario asociado a cada una de estas unidades tendría, presumiblemente, una concentración de centros suficiente como para justificar la aplicación del modelado SCMVQ.

Apéndice A

ESCALAMIENTO EN EL ALGORITMO ADELANTE-ATRÁS

Las probabilidades adelante escaladas son de la forma,

$$\tilde{\alpha}_t(i) = K_t \alpha_t(i) \qquad (i = 1, \ldots, N) \tag{A.1}$$

El algoritmo adelante-atrás modificado es el siguiente:

1) Se calculan las probabilidades $\alpha_1(j)$,

$$\alpha_1(j) = \pi_j b_j(o_1) \tag{A.2}$$

y el coeficiente $K_1 = c_1$, donde,

$$c_1 = \left[\sum_{j=1}^{N} \alpha_1(j) \right]^{-1} \tag{A.3}$$

Las probabilidades escaladas son,

$$\tilde{\alpha}_1(j) = c_1 \alpha_1(j) = K_1 \alpha_1(j) \tag{A.4}$$

2) Para $t = 2, \ldots, T$ se calculan unas α_t^* como,

$$\begin{aligned} \alpha_t^*(j) &= \left[\sum_{i=1}^{N} \tilde{\alpha}_{t-1}(i) a_{ij}\right] b_j(o_t) = \\ & \left[\prod_{s=1}^{t-1} c_s\right] \left[\sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij}\right] b_j(o_t) = \\ & \left[\prod_{s=1}^{t-1} c_s\right] \alpha_t(j) \end{aligned} \tag{A.5}$$

El coeficiente c_t se calcula como,

$$c_t = \left[\sum_{j=1}^{N} \alpha_t^*(j)\right]^{-1} \tag{A.6}$$

y las probabilidades escaladas son,

$$\tilde{\alpha}_t(j) = c_t \alpha_t^*(j) = \left[\prod_{s=1}^{t} c_s\right] \alpha_t(j) = K_t \alpha_t(i) \tag{A.7}$$

con,

$$K_t = \left[\prod_{s=1}^{t} c_s\right] \tag{A.8}$$

3) Queda por calcular $P(O|\lambda)$. Para ello, se tiene en cuenta que,

$$\sum_{j=1}^{N} \tilde{\alpha}_t(j) = 1 \tag{A.9}$$

como fácilmente se desprende de (A.6) y (A.7). En particular,

$$\sum_{j=1}^{N} \tilde{\alpha}_T(j) = K_T \sum_{j=1}^{N} \alpha_T(j) = \left[\prod_{s=1}^{T} c_s\right] \sum_{j=1}^{N} \alpha_T(j) = 1 \tag{A.10}$$

de donde, teniendo en cuenta (2.46), se obtiene que,

$$P(O|\lambda) = 1/K_T = \left[\prod_{s=1}^{T} c_s\right]^{-1} \tag{A.11}$$

Esta expresión puede dar problemas de cálculo, por lo que es preferible el uso de logaritmos en (A.11),

$$\log P(O|\lambda) = -\sum_{s=1}^{T} \log c_s \tag{A.12}$$

Apéndice B

FÓRMULAS DE BAUM-WELCH

Las fórmulas de reestimación de Baum [Levinson83] pueden ser obtenidas a partir de los siguientes lemas:

Lema 1: dada una secuencia $\{u_i, i = 1, \ldots, N\}$ de números reales positivos, y otra $\{v_i, i = 1, \ldots, N\}$ de números reales no negativos (con, al menos, un elemento no nulo), es posible comprobar, a partir de la concavidad de la función logarítmo, que,

$$\log\left(\frac{\sum_i v_i}{\sum_i u_i}\right) \geq \frac{1}{\sum_k u_k}\left[\sum_i (u_i \log v_i - u_i \log u_i)\right] \tag{B.1}$$

Lema 2: dada la secuencia no negativa $\{c_i, i = 1, \ldots, N\}$, la función

$$F(\mathbf{x}) = \sum_{i=1}^{N} c_i \log x_i \tag{B.2}$$

con la condición $\sum_i x_i = 1$ tiene un único máximo global para

$$x_i = \frac{c_i}{\sum_i c_i} \tag{B.3}$$

como puede comprobarse fácilmente aplicando el método de Lagrange.

El primer lema se aplica considerando cada u_i como la probabilidad conjunta $P(Q_i, O|\lambda)$ de un cierto camino Q_i y la secuencia de observación O dado el modelo λ. Cada v_i tiene el mismo significado, pero para otro modelo $\hat{\lambda}$. En este caso,

$$\begin{aligned} \sum_i u_i &= P(O|\lambda) \\ \sum_i v_i &= P(O|\hat{\lambda}) \end{aligned} \tag{B.4}$$

y aplicando el lema,

$$\log \frac{P(O|\hat{\lambda})}{P(O|\lambda)} \geq \mathcal{Q}(\lambda, \hat{\lambda}) - \mathcal{Q}(\lambda, \lambda) \tag{B.5}$$

donde se ha definido la función,

$$\mathcal{Q}(\lambda, \hat{\lambda}) \equiv \sum_Q P(Q|O, \lambda) \log P(Q, O|\hat{\lambda}) \tag{B.6}$$

normalmente denominada *función auxiliar de Baum* (la sumatoria se extiende a todos los caminos Q posibles). La inecuación (B.5) indica que si es posible obtener que la diferencia del segundo miembro sea positiva, entonces se estará obteniendo un modelo $\hat{\lambda}$ que aumenta la probabilidad total, tal como es el objetivo de la estimación ML. La forma de asegurar lo anterior es maximizar $\mathcal{Q}(\lambda, \hat{\lambda})$ en (B.5), respecto al conjunto de parámetros $\hat{\lambda}$. Para ello se tiene que,

$$\log P(Q, O|\hat{\lambda}) = \log \hat{\pi}_{q_1} + \sum_{t=1}^{T-1} \log \hat{a}_{q_t q_{t+1}} + \sum_{t=1}^{T} \log \hat{b}_{q_t}(o_t) \tag{B.7}$$

y la función auxiliar (B.6) puede expresarse como,

$$\mathcal{Q}(\lambda, \hat{\lambda}) = \sum_{i=1}^{N} \sum_{j=1}^{N} c_{ij} \log \hat{a}_{ij} + \sum_{j=1}^{N} \sum_{k=1}^{M} d_{jk} \log \hat{b}_j(v_k) + \sum_{i=1}^{N} e_i \log \hat{\pi}_i \tag{B.8}$$

con,

$$c_{ij} = \sum_{Q} P(Q|O,\lambda) n_{ij}(Q) \tag{B.9a}$$

$$d_{jk} = \sum_{Q} P(Q|O,\lambda) m_{jk}(Q) \tag{B.9b}$$

$$r_i = \sum_{Q} P(Q|O,\lambda) r_i(Q) \tag{B.9c}$$

donde se ha hecho uso de la siguiente notación,

$$\begin{aligned} n_{ij}(Q) &= \text{número de transiciones del estado } s_i \text{ al } s_j \\ m_{jk}(Q) &= \text{número de veces símbolo } v_k \text{ en el estado } s_j \\ r_i(Q) &= \begin{cases} 1 & \text{si } s_i \text{ es el estado inicial} \\ 0 & \text{en caso contrario} \end{cases} \end{aligned}$$

Obviamente, c_{ij}, d_{jk} y e_i son los valores esperados de n_{ij}, m_{jk} y r_i, dados el modelo λ y la secuencia O.

Como puede observarse, $\mathcal{Q}(\lambda, \hat{\lambda})$ es una función del mismo tipo que el expuesto en el lema 2, por lo que será minimizada por un modelo $\hat{\lambda}$ con los siguientes parámetros,

$$\hat{a}_{ij} = \frac{c_{ij}}{\sum_{l} c_{il}} \tag{B.10a}$$

$$\hat{b}_i(v_k) = \frac{d_{jk}}{\sum_{l} d_{jl}} \tag{B.10b}$$

$$\hat{\pi}_i = \frac{r_i}{\sum_{l} c_l} \tag{B.10c}$$

que corresponden a las fórmulas de reestimación dadas en (3.10).

Apéndice C

APROXIMACIÓN AL ERROR Y DESCENSO PROBABILÍSTICO

C.1 Aproximación al error del sistema

Se puede comprobar que el coste esperado definido en (3.48) es una aproximación a la probabilidad de error del sistema. Para ello, se define la función de error para la clase W_k como,

$$e_k(X,\Lambda) = \delta(X \in W_k)\delta\left[g_k(X,\Lambda) \neq \max_i g_i(X,\lambda)\right] \tag{C.1}$$

El error total puede obtenerse como el valor esperado de las $e_k(X,\Lambda)$,

$$\begin{aligned} E &= \sum_{k=1}^{L}\int_{\mathcal{X}} P(X,W_k)e_k(X,\Lambda)dX \\ &= \sum_{k=1}^{L}\int_{\mathcal{X}} P(X,W_k)\delta(X \in W_k)\delta\left[g_k(X,\Lambda) \neq \max_i g_i(X,\lambda)\right]dX \\ &\simeq \sum_{k=1}^{L}\int_{\mathcal{X}} P_\Lambda(X,W_k)\delta(X \in W_k)l_k(X,\Lambda)dX = L(\Lambda) \end{aligned} \tag{C.2}$$

La bondad de la aproximación puede ser controlada mediante los parámetros α y β, introducidos en la definición de las funciones $l_k(d_k)$. Por tanto, la minimización de $L(\Lambda)$ está directamente relacionada con la minimización de la probabilidad de error del sistema, lo cual puede realizarse mediante un descenso probabilístico.

C.2 Descenso Probabilístico Generalizado

El descenso probabilístico surge de la idea de entrenar un clasificador basado en funciones discriminantes tipo lineal [Amari67]. Su extensión a clasificadores de cualquier tipo da lugar al *descenso probabilístico generalizado* (GPD). Se basa en la aplicación de una regla de aprendizaje adaptativa, en la que cada vez que llega un objeto de entrada X (en el instante t), el conjunto de parámetros del clasificador en ese instante, Λ_t, se modifica mediante,

$$\Lambda_{t+1} = \Lambda_t + \delta\Lambda_t \tag{C.3}$$

con el objetivo de minimizar el coste esperado, $L(\Lambda)$, de forma que no sea necesario almacenar los datos ya utilizados, ni hacer ningún tipo de suposición acerca de las probabilidades $P(W_k)$ y $P(X|W_k)$ (que son desconocidas).

Si la magnitud de la corrección es pequeña, es posible quedarse con la aproximación de primer orden,

$$L(\Lambda_{t+1}) \simeq L(\Lambda_t) + \delta\Lambda_t \bigtriangledown L(\Lambda)|_{\Lambda=\Lambda_t} \tag{C.4}$$

Tomando valores esperados,

$$E\left[L(\Lambda_{t+1}) - L(\Lambda_t)\right] = E\left[\delta L(\Lambda_t)\right] = E\left[\delta\Lambda_t\right] \bigtriangledown L(\Lambda_t) \tag{C.5}$$

El objetivo es conseguir una regla de aprendizaje que asegure que $E\left[\delta L(\Lambda_t)\right] \leq 0$. El siguiente lema da respuesta al problema.

Lema: dado $X \in W_k$, si la regla de ajuste es,

$$\delta\Lambda(X, W_k, \Lambda) = -\epsilon \mathbf{U} \bigtriangledown l_k(X, \Lambda) \tag{C.6}$$

donde ϵ es un número positivo pequeño y $\mathbf{U}$ una matriz de elementos positivos,

entonces

$$E\left[\delta L(\Lambda_t)\right] \leq 0 \tag{C.7}$$

prevaleciendo la igualdad sólo en los mínimos de $L(\Lambda)$).

Demostración: calculando el valor esperado de la corrección propuesta en el lema anterior,

$$E\left[\delta\Lambda\right] = -\epsilon \mathbf{U} E\left[\bigtriangledown l_k(X,\Lambda)\right] = -\epsilon \mathbf{U} \bigtriangledown L(\Lambda) \tag{C.8}$$

Sustituyendo en (C.5),

$$E\left[\delta L(\Lambda)\right] = -\epsilon \mathbf{U}| \bigtriangledown L(\Lambda)|^2 \leq 0 \tag{C.9}$$

dado que $\epsilon > 0$ y $\mathbf{U}$ es definida positiva.

El valor asignado a ϵ controla la validez de la aproximación de primer orden realizada. Dado que $\delta L(\Lambda)$ es negativo sólo en promedio, este método puede ser denominado *descenso probabilístico*.

Bibliografía

[Ainsworth76] W. A. Ainsworth. *Mechanisms Of Speech Recognition*. Pergamon Press, 1976.

[Amari67] S. Amari. "A Theory of Adaptive Pattern Classifiers,". *IEEE Trans. on Electronic Computers*, 16(3):299–307, Junio 1967.

[Bahl80] L.R. Bahl, R. Bakis, P.S. Cohen, A.G. Cole, F. Jelinek, B.L. Lewis, and R.L. Mercer. "Further Results on a Continuously Read Natural Corpus,". In *Proc. of ICASSP-80*, pages 872–875, Denver, Abril 1980.

[Bahl83] L.R. Bahl, F. Jelinek, and R.L. Mercer. "A Maximum Likelihood approach to continuous speech recognition,". *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 5:179–190, Marzo 1983.

[Bahl86] L.R. Bahl, P.F. Brown, P.V. de Souza, and R.L. Mercer. "Maximum Mutual Information Estimation of Hidden Markov Models Parameters for Speech Recognition,". In *Proc. of ICASSP-86*, pages 49–52, Tokyo, 1986.

[Bahl87] L.R. Bahl, P.F. Brown, P.V. de Souza, and R.L. Mercer. "Estimating Hidden Markov Model Parameters so as to Maximize Recognition Accuracy,". *Computer Science*, 1987.

[Baker75] J.K. Baker. "The DRAGON system - an overview,". *IEEE Trans. on ASSP*, 23:24–29, Febrero 1975.

[Bellman57] R. E. Bellman. *Dynamic Programming*. Princeton Univ. Press, 1957.

[Bocchieri86] E. L. Bocchieri and G. R. Doddington. "Frame-Specific Statistical Features for Speaker Independent Speech Recognition,". *IEEE Transaction on ASSP*, 34:755–764, Agosto 1986.

[Bogert63] B. P. Bogert, M. J. Healy, and W. Tukey. "The quefrency analysis of time series for echoes: Cepstrum, pseudo-autocovariance, cross-cepstrum and shape cracking,". In *Proc. Symp. Time Series Anal.*, pages 209–243. M. Rossenblatt, Ed. New York: Wiley, 1963.

[Bridle90] J.S. Bridle. "Alpha-Nets: a recurrent neural network architecture with a Hidden Markov Model Interpretation,". *Speech Communication*, 9:83–92, 1990.

[Brigham74] O. Brigham. *The Fast Fourier Transform*. Prentice-Hall, 1974.

[Brown82] M.K. Brown and L. R. Rabiner. "On the Use of Energy in LPC-Based Recognition of Isolated Words,". *Th Bell System Technical Journal*, 61:2971–2985, Diciembre 1982.

[Burton85] D.K. Burton and J.E. Shore J.T. Buck. "Isolated-Word Speech Recognition Using Multisection Vector Quantization Codebooks,". *IEEE Trans. on ASSP*, 33(4):837–849, Agosto 1985.

[Buzo80] A. Buzo, A.H. Gray, R.M. Gray, and J.D. Markel. "Speech Coding Based Upon Vector Quantization,". *IEEE Trans. on ASSP*, 28:562–574, Octubre 1980.

[Casacuberta87] F. Casacuberta and E. Vidal. *Reconocimiento Automático del Habla*. Marcombo, 1987.

[Casacuberta90] C. Puchol and F. Casacuberta. "Reconocimiento de Palabras Aisladas mediante Redes Neuronales,". In *IV Simposium Nacional de Reconocimiento de Formas y Análisis de Imágenes*, Septiembre 1990.

[Chang93] P.C. Chang and B.H. Juang. "Discriminative Training of Dynamic Programming Based Speech Recognizers,". *IEEE Trans. on Speech and Audio Processing*, 1(2):135–143, Abril 1993.

[Chou92] W. Chou, B.H. Juang, and C.H. Lee. "Segmental GPD Training of HMM Based Speech Recognizer,". In *Proc. of ICASSP-92*, pages I–473–476, 1992.

[Class86] F. Class, A. Kaltenmeier, and H. Katterfeldt. "Comparison of two connected word recognition principles: Hidden Markov Modeling and Dynamic Time Warping,". In *Proc. of EUSIPCO-86*. Elsevier Science Publishers, 1986.

[Derouault86] A.M. Derouault and B. Merialdo. "Natural Language Modeling for Phoneme-to-Text Transcriptions,". *IEEE Trans. on PAMI*, 5:742–749, 1986.

[Derouault87] A.M. Derouault. "Context-Dependent Phonetic Markov Models for Large Vocabulary Speech Recognition,". In *Proc. of ICASSP-87*, pages 360–363, Abril 1987.

[Diaz91] J.E. Díaz, A.M. Peinado, J.C. Segura, and M.C. Benítez, A.Rubio. "Recurrent Neural Networks for Speech Recognition,". In *Proc. of IWANN-91*, Granada, Septiembre 1991.

[Diaz91a] J.E. Díaz. *Redes Neuronales Recurrentes para Reconocimiento de Voz*. Tesis de Licenciatura,, Universidad de Granada, Nov 1991.

[Diaz93] J.E. Díaz, J.C. Segura, A.J. Rubio, A.M. Peinado, and J.L. Pérez. "A new neuron model for an alphanet-semicontinuous HMM,". In *Proc. of ICASSP-93*, volume 1, pages 529–532, 1993.

[Duda73] R.O. Duda and P.E. Hart. *Pattern Classification and Scene Analysis*. J. Wiley and Sons, 1973.

[Ephraim89] Y. Ephraim, A. Dembo, and L.R. Rabiner. "A Minimum Discrimination Information Approach for Hidden Markov Modeling,". *IEEE Trans. on Information Theory*, 35(5):1001–1013, Septiembre 1989.

[Ephraim90] Y. Ephraim and L.R. Rabiner. "On the relations between modeling approaches for speech recognition,". *IEEE Trans. on Information Theory*, 36(2):372–380, Marzo 1990.

[Flanagan72] J. L. Flanagan. *Speech Analysis, Synthesis and Perception*. Springer-Verlag, 1972.

[Fukunaga90] K. Fukunaga. *Introduction to Statistical Pattern Recognition*. Academic Press, 1990.

[Furui81] S. Furui. "Cepstral Analysis Technique for Automatic Speaker Verification,". *IEEE Trans. on ASSP*, 29:254–272, Abril 1981.

[Furui86] S. Furui. "Speaker-Independent Isolated Word Recognition Using Dynamic Features of Speech Spectrum,". *IEEE Trans. on ASSP*, 34:52–59, Febrero 1986.

[Garcia93] P. García, J.C. Segura, A.J. Rubio, and J. Díaz. "An efficient prunning algorithm for continuous speech recognition,". In *Proc. of NATO ASI 93*, Bubión, Junio-Julio 1993.

[Gersho91] A. Gersho and R.M. Gray. *Vector Quantization and Signal Compression*. Kluwer Academic Publishers, 1991.

[Gopalakrishnan89] P.S. Gopalakrishnan, D. Kanevsky, A. Nadas, and D. Nahamoo. "A Generalization of the Baum Algorithm to Rational Objective Functions,". In *Proc. of ICASSP-89*, pages 631–634, Glasgow (Scotland), Mayo 1989.

[Gray80] R.M. Gray, A. Buzo, A.H. Gray, and Y. Matsuyama. "Distortion Measures for Speech Processing,". *IEEE Trans. on ASSP*, 28(4), Agosto 1980.

[Gray84] R.M. Gray. "Vector Quantization,". *IEEE ASSSP Magazine*, Abril 1984.

[Group85] IBM Speech Recognition Group. "A Real-Time, Isolated-Word, Speech Recognition System for Dictation Transcription,". In *Proceedings of ICASSP-85*, Marzo 1985.

[Gupta87] V. N. Gupta, M. Lennig, and P. Mermelstein. "Integration of Acoustic Information in a Large Vocabulary Word Recognizer,". In *Proceedings of ICASSP-87*, Abril 1987.

[Huang89] X.D. Huang and M.A. Jack. "Unified Techniques for Vector Quantisation and Hidden Markov Modeling Using Semi-Continuous Models,". In *Proc. of ICASSP-89*, pages 639–642, Glasgow (Scotland), Mayo 1989.

[Huang90] X. Huang, K.F. Lee, and H.W Hon. "On Semi-Continuous Hidden Markov Modeling,". In *Proc. of ICASSP-90*, pages 689–692, 1990.

[Jelinek76] F. Jelinek. "Continuous Speech Recognition by Statistical Methods,". In *IEEE Proceedings*, volume 64, pages 532–556, Abril 1976.

[Juang82] B.H. Juang, D.Y. Womg, and A.H. Gray. "Distortion Performance of Vector Quantization for LPC Voice Coding,". *IEEE Trans. on ASSP*, 30(2), Abril 1982.

[Juang85] B.H. Juang and L.R. Rabiner. "Mixture Autorregressive Hidden Markov Models for Speech Signals,". *IEEE Trans. on ASSP*, 33(6):1404–1413, Diciembre 1985.

[Juang87] B. H. Juang, L. R. Rabiner, and J. G. Wilpon. "On the use of bandpass liftering in speech recognition,". *IEEE Trans. on ASSP*, 35(7):947–954, julio 1987.

[Juang92] B.H. Juang and S. Katagiri. "Discriminative Learning for Minimum Error Classification,". *IEEE Trans. on Signal Processing*, 40(12):3043–3054, Diciembre 1992.

[Junqua89] J.C. Junqua and H. Wakita. "A Comparative Study of Cepstral Lifters And Distance Measures for All Pole Models of Speech in Noise,". In *Proc. of ICASSP-89*, pages 476–479, 1989.

[Kirkpatrick83] S. Kirkpatrick, C.D. Gelatt, and M.P. Vacchi. "Optimization by Simulated Annealing,". *Science*, 220:671–680, 1983.

[Kohonen90] T. Kohonen. "The Self-Organizing Map,". *Proc. of IEEE*, 78(9):1464–1480, Septiembre 1990.

[Lee89] K. F. Lee. *Automatic Speech Recognition (The Development of the Sphinx System)*. Kluwer Academic Publishers, 1989.

[Lee90] K. F. Lee. "Context-Dependent Phonetic Hidden Markov Models for Speaker-Independent Continuous Speech Recognition,". *IEEE Trans. on ASSP*, 38:599–623, Abril 1990.

[Lee90a] K. F. Lee, H. W. Hon, and R. Reddy. "An Overview of the SPHINX Speech Recognition System,". *IEEE Trans. on ASSSP*, 38:35–45, Enero 1990.

[Lee90b] Y.T. Lee and D. Kahn. "Information-Theoretic Distorsion Measures for Speech Recognition: Theoretical Considerations and Experimental Results,". In *Proc. of ICASSP-90*, pages 785–788, 1990.

[Lee90c] C.H. Lee, L.R. Rabiner, R. Pieraccini, and J.G. Wilpon. "Acoustic modeling for large vocabulary speech recognition,". *Computer Speech and Language*, 4:127–165, 1990.

[Levinson83] S.E. Levinson, L.R. Rabiner, and M.M. Sondhi. "An Introduction to the Application of the Theory of Probabilistic Functions of a Markov Process to Automatic Speech Recognition,". *The Bell System Technical Journal*, 62(4):1035–1074, Abril 1983.

[Linde80] Y. Linde, A. Buzo, and R.M. Gray. "An Algorithm for Vector Quantizer Design,". *IEEE Trans. on Commun.*, 28, Enero 1980.

[Lippman87] R.P. Lippman. "A Introduction to Computing with Neural Nets,". *IEEE Trans. on ASSP*, 4(2):4–22, Abril 1987.

[Ljolje90] A. Ljolje, Y. Ephraim, and L.R. Rabiner. "Estimation of Hidden Markov Model Parameters by Minimizimg Empirical Error Rate,". In *Proc. of ICASSP-90*, pages 709–712, 1990.

[Lleida90] E. Lleida, C. Nadeu, and J. B. Mariño. "Statistical Feature Selection for Isolated Word Recognition,". In *Proceedings of ICASSP-90*, Alburquerque, Abril 1990.

[Makhoul75] J. Makhoul. "Linear Prediction: A Tutorial Review,". *Proceedings of IEEE*, 63:561–580, Abril 1975.

[Mariani89] J. Mariani. "Recent Advances in Speech Processing,". In *Proc. of ICASSP-89*, pages 429–440, 1989.

[Markel76] J.D. Markel and A.H. Gray. *Linear Prediction of Speech*. Springer-Verlag, 1976.

[Moreno90] P.J. Moreno, D.B. Roe, and P. Ramesh. "Rejection Techniques in Continuous Speech Recognition Using Hidden Markov Models,". In *Proc. of EUSIPCO-90*, volume 2, pages 1383–1386, Septiembre 1990.

[Myers81] C. S. Myers and L. R. Rabiner. "A Level Building Dynamic Time Warping Algorithm for Connected Word Recognition,". *IEEE Trans. on ASSP*, 29:284–297, Abril 1981.

[Nadas81] A. Nadas, R. L. Mercer, L. R. Bahl, R. Bakis, P. S. Cohen, A. G. Cole, F. Jelinek, and B. L. Lewis. "Continuous Speech Recognition with Automatically Selected Acoustic Prototypes Obtained by Either Bootstrapping or Clustering,". In *Proceedings of ICASSP-81*, Abril 1981.

[Nadas83] A. Nadas. "A decision theoretic formulation of a training problem in speech recognition and a comparison of training by unconditional versus conditional maximum likelihood,". *IEEE Trans. on ASSP*, 31(4):814–817, Agosto 1983.

[Nadas88] A. Nadas, D. Nahamoo, and M.A. Picheny. "On a Model-Robust Training Method for Speech Recognition,". *IEEE Trans. on ASSP*, 36(9):1432–1436, Septiembre 1988.

[Nocerino85] N. Nocerino, F.K. Soong, L.R. Rabiner, and D.H. Klatt. "Comparative Study of Several Distortion Measures for Speech Recognition,". In *Proc. of ICASSP-85*, 1985.

[Oppenheim75] A.V. Oppenheim and R.W. Schafer. *Digital Signal Processing*. Prentice-Hall, Inc., 1975.

[Paliwal82] K. K. Paliwal. "On the performance of the quefrency-weighted cepstral coefficients in vowel recognition,". *Speech Communication*, 1:151–154, Mayo 1982.

[Peinado89] Antonio M. Peinado. *Reconocimiento Monolocutor de Palabras Aisladas Usando Modelos Ocultos de Markov y Cuantización Vectorial para Control de un Robot mediante Voz*. Tesis de Licenciatura, Universidad de Granada, Abril 1989.

[Peinado90] A.M. Peinado, P. Ramesh, and D.B. Roe. "On the Use of Energy Information for Speech Recognition Using HMM,". In *Proceedings of EUSIPCO-90*, volume 2, pages 1243–1246, Barcelona, Sept. 1990.

[Peinado91] A.M. Peinado, J.M. López, V.E. Sánchez, J.C. Segura, and A.J. Rubio. "Improvements in HMM-based isolated word recognition system,". *IEE Proceedings-I*, 138(3):201–206, Junio 1991.

[Peinado91a] A.M. Peinado, A. Rubio, J.M. López, J.C. Segura, and V. Sánchez. "Including duration Information in a threshold-based rejector for HMM Speech Recognition,". In *Proc. of Congres International des Sciencies Phonetiques*, volume 4, pages 470–473, Aix-en-Provence, Agosto 1991.

[Peinado91b] A.M. Peinado, R. Román, A. Rubio, P. García, and J. Díaz. "Entropic Training for HMM Speech Recognition,". In *Proc. of EUROSPEECH 91*, volume 2, pages 651–654, Génova (Italia), Septiembre 1991.

[Peinado93] A.M. Peinado, J.C. Segura, A.J. Rubio, and V.E. Sánchez. *New Advances and Trends in Speech Recognition and Coding*, chapter A MMI Codebook Design for MVQHMM Speech Recognition. NATO ASI 93, 1993.

[Peinado94] A.M. Peinado, J.C. Segura, A.J. Rubio, and M.C. Benítez. "Using Multiple Vector Quantization and Semicontinuous Hidden Markov Models for Speech Recognition,". In *Proc. of ICASSP-94*, 1994.

[Prager87] R.W. Prager, T.D. Harrison, and F. Fallside. "A comparison of the Boltzmann Machine and the Backpropagation network as recognizers of static speech patterns,". *Computer, Speech and Language*, 2(3/4), Sept/Dic 1987.

[Quilis89] A. Quilis and J. A. Fernández. *Curso de Fonética y Fonología Españolas*. C.S.I.C., 1989.

[Rabiner75] L.R. Rabiner and M.R. Sambur. "An algorithm for detecting the endpoints of isolated utterances,". *Bell System Tech.*, 54:297–315, Febrero 1975.

[Rabiner78] L. R. Rabiner and R. W. Schafer. *Digital Processing of Speech Signals*. Prentice-Hall, 1978.

[Rabiner83] L. R. Rabiner, S. E. Levinson, and M. M. Sondhi. "On the Application of Vector Quantization and Hidden Markov Models to Speaker-

Independent, Isolated Word Recognition,". *The Bell System technical Journal*, 62(4):1075–1105, Abril 1983.

[Rabiner84] L. R. Rabiner, M. M. Sondhi, and S. E. Levinson. "A Vector Quantizer Incorporating Both LPC Shape and Energy,". In *Proc. of ICASSP 84*, 1984.

[Rabiner85] L.R. Rabiner, B.H. Juang, S.E. Levinson, and M.M. Sondhi. "Some Properties of Continuous Hidden Markov Model Representations,". *AT&T Technical Journal*, 64(6):1251–1269, Julio-Agosto 1985.

[Rabiner85a] L.R. Rabiner, B.H. Juang, S.E. Levinson, and M.M. Sondhi. "Recognition of Isolated Digits Using Hidden Markov Models With Continuous Mixture Densities,". *AT&T technical Journal*, 64(6):1211–1233, Julio-Agosto 1985.

[Rabiner86] L.R. Rabiner and B. H. Juang. "An Introduction to Hidden Markov Models,". *IEEE ASSP Magazine*, Enero 1986.

[Rabiner86a] L.R. Rabiner, J.G. Wilpon, and B.H. Juang. "A Segmental K-Means Training Procedure for Connected Word Recognition,". *AT&T Technical Journal*, 65(3):21–32, Mayo-Junio 1986.

[Rabiner89] L. R. Rabiner, J. G. Wilpon, and F. K. Soong. "High Performance Connected Digit Recognition Using Hidden Markov Models,". *IEEE Trans. on ASSP*, 37:1214–1225, Agosto 1989.

[Rabiner89a] L. R. Rabiner. "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition,". In *Proceedings of the IEEE*, volume 77, pages 257–285, Febrero 1989.

[Rabiner93] L.R. Rabiner and B.H. Juang. *Fundamentals of Speech Recognition*. Prentice-Hall, 1993.

[Reddy75] D.R. Reddy. *Speech Recognition*. Academic Press, 1975.

[Roe93] D.B. Roe and J.G. Wilpon. "Whiter Speech Recognition: The Next 25 Years,". *IEEE Communications Magazine*, 31(11):54–62, Noviembre 1993.

[Rosenblatt59] R. Rosenblatt. *Principles of Neurodynamics*. Spartan Books, New York, 1959.

[Sakoe79] H. Sakoe. "Two-Level DP Matching - A Dynamic Programming Based Pattern Matching Algorithm for Connected Word Recognition,". *IEEE Trans. on ASSP*, 27:588–595, Dic. 1979.

[Schalkoff92] R. Schalkoff. *Pattern Recognition (Statistical, Structural and Neural Approaches)*. J. Wiley & Sons, 1992.

[Schwartz85] R. Schwartz, Y. Chow, O. Kimball, S. Roucos, M. Krasner, and J. Makhoul. "Context-Dependent Modeling for Acoustic-Phonetic Recognition of Continuous Speech,". In *Proc. of ICASSP-85*, Abril 1985.

[Segura84] José C. Segura. *Cuatro Métodos de Codificación para el Reconocimiento de Palabras Aisladas*. Tesis de Licenciatura, Universidad de Granada, 1984.

[Segura91] J.C. Segura. "Modelos de Markov con Cuantización Dependiente para Reconocimiento de Voz,". *Tesis Doctoral*, Noviembre 1991.

[Segura94] J.C. Segura, A.J. Rubio, A.M. Peinado, P. García, and R. Román. "Multiple VQ Hidden Markov Modeling for Speech Recognition,". *Speech Communication (in press)*, 1994.

[Shikano91] K. Shikano and F. Itakura. "Spectrum Distance Measures for Speech Recognition,". In S. Furui and M.M. Sondhi, editors, *Advances in Speech Signal Processing*, chapter 14, pages 419–452. Marcel Dekker, Inc., 1991.

[Sugamura83] N. Sugamura, K. Shikano, and S. Furui. "Isolated Word Recognition Using Phoneme-Like Templates,". *IEEE ICASSP-83*, pages 723–726, 1983.

[Tohkura87] Y. Tohkura. "A weighted cepstral distance measure for speech recognition,". *IEEE Trans. on ASSP*, 35, no. 10:1414–1422, Octubre 1987.

[Tou74] J. T. Tou and R. C. González. *Pattern Recognition Principles*. Addison-Wesley Publishing Company, Inc., 1974.

[Vaisey88] J. Vaisey and A. Gersho. "Simulated Annealing and Codebook Design,". In *Proc. of ICASSP-88*, pages 1176–1179, Abril 1988.

[Vintsjuk68] T. K. Vintsjuk. "Recognition of Words of Oral Speech by Dynamic Programming,". *Kibernetica*, 4(1):81–88, 1968.

[Waibel89] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. Lang. "Phoneme Recognition using Time-Delay Neural Networks,". *IEEE Trans. on ASSP*, 37(3):328–339, Marzo 1989.

[Waibel91] A. Waibel. *Neural Network Approaches for Speech Recognition*, pages 555–596. Marcel Dekker, Inc., 1991.

[Wilpon85] J. G. Wilpon and L. R. Rabiner. "A modified k-means clustering algorithm for use in isolated word recognition,". *Transanctions on ASSP*, 33, Junio 1985.

[Wilpon90] J.G. Wilpon, L.R. Rabiner, C-H. Lee, and E.R. Goldman. "Automatic Recognition of Keywords in Unconstrained Speech Using Hidden Markov Models,". *IEEE Trans. on ASSP*, 38, no. 11:1870–1878, Noviembre 1990.

[Yair92] E. Yair, K. Zeger, and A. Gersho. "Competitive Learning and Soft Competition for vector Quantizer Design,". *IEEE Trans. on Signal Processing*, 40, no. 2, Febrero 1992.

Índice temático