UNIVERSIDAD DE LAS PALMAS DE GRAN CANARIA

DEPARTAMENTO DE SEÑALES Y COMUNICACIONES

TESIS DOCTORAL

ESTRATEGIAS PARA COMBINACIÓN DE CLASIFICADORES DE SENCUENCIAS TEMPORALES

ITZIAR ALONSO GONZÁLEZ

Las Palmas de Gran Canaria, Mayo de 2000

40/1999-00

UNIVERSIDAD DE LAS PALMAS DE GRAN CANARIA
UNIDAD DE TERCER CICLO Y POSTGRADO

Reunido el día de la fecha, el Tribunal nombrado por el Excmo. Sr. Rector Magfco. de esta Universidad, el/a aspirante expuso esta TESIS DOCTORAL.
Terminada la lectura y contestadas por el/a Doctorando/a las objeciones formuladas por los señores miembros del Tribunal, éste calificó dicho trabajo con la nota de SOBRESALIENTE CUM LAUDE (POR UNANIMIDAD, 5 votos)

Las Palmas de Gran Canaria, a 12 de mayo de 2000.

El/a Presidente/a: Dr.D. Pedro Saavedra Santana,

El/a Secretario/a: Dr.D. Alvaro Suárez Carmiento,

El/a Vocal: Dr.D. Juan Luis Navarro Mesa,

El/a Vocal: Dr.D. Francisco Javier Hernando Pericás,

El/a Vocal: Dr.D. ~~Luis Hernández Gómez~~ Fernando Díaz de María,

La Doctoranda: Dª. Itziar Goretti Alonso González,,

Universidad de Las Palmas de Gran Canaria

Tesis Doctoral

Estrategias para combinación de clasificadores de secuencias temporales

AUTORA:	Itziar G. Alonso González
DIRECTOR:	Miguel Ángel Ferrer Ballester
CO-DIRECTOR:	Aníbal Figueiras Vidal
FECHA:	Mayo 2000

Universidad de Las Palmas de Gran Canaria

Tesis Doctoral

Estrategias para combinación de clasificadores de secuencias temporales

PRESIDENTE:

SECRETARIO:

VOCAL:

VOCAL:

VOCAL:

CALIFICACIÓN:

DIRECTOR: Miguel Ángel Ferrer Ballester

CO-DIRECTOR: Aníbal Figueiras Vidal

AUTORA: Itziar G. Alonso González

a la memoria de mi padre

y a mi madre

Agradecimientos

Ha llegado el momento y lo mejor para empezar es el dar las gracias a todos los que han estado conmigo estos años, animándome y apoyándome. Especialmente

A Miguel Ángel Ferrer, por su constante paciencia, y ánimo. A Aníbal Figueiras, por orientarnos al comienzo de este trabajo. A Carlos Travieso, que gracias a su dedicación se han hecho muchas cosas en el grupo.

A mis compañeros del aula Magna, Carlos Ley, Alvaro, Miguel, Carlos Ramírez, Francis, a Domingo, y a Elsa, muchos momentos son los vividos y muchos los que nos quedan por vivir.

A Carmen, que además de ser la compañera de despacho, es la compañera de trabajo, de música, de los ratos del chocolate, del cortado, de los bombones, del té, la manzanilla, etc. Nos ha dado tiempo para probar casi de todo y no engordar demasiado con el fin de darnos ánimos en el día a día.

A mi familia, a mi madre, y a mis hermanos, ahora ya les puedo contestar, que ya casi he terminado.

Y a ti, Juan, siempre has creído que lo consiguiría, por tu ayuda, y tu comprensión. A partir de ahora, espero poder disfrutar juntos, un poco más de la vida.

Yo también me atrevo y un poco más, a Alejandro S.

A todos, gracias

Índice General

1 Introducción **1**

1.1 Antecedentes 1
1.2 Objetivos 3
1.3 Evolución de los clasificadores aplicados a voz 5
1.4 Estructura de la memoria 9

2 Estado del Arte. Clasificadores de Patrones **11**

2.1 Introducción a los clasificadores 11
2.2 Clasificación de secuencias 13
2.3 Clasificadores de patrones 13
2.3.1 Cuantificador vectorial 14
2.4 Técnicas de alineamiento temporal de patrones, (DTW) 18
2.4.1 Principios de la programación dinámica 21
2.5 Árboles de decisión 22
2.6 Aplicación de las redes neuronales 24
2.6.1 Introducción 24
2.6.2 Descripción general de las redes neuronales 25
2.6.3 Redes estáticas 27
2.6.4 Redes dinámicas 39
2.7 Modelos ocultos de Markov, HMM 41
2.7.1 Introducción 41
2.7.2 Proceso discreto de Markov 42
2.7.3 Elementos de un HMM 43
2.7.4 Los 3 problemas básicos de un HMM 44
2.7.5 Tipos de HMMs 45
2.8 Conclusión 45

3 Bases de datos **47**

3.1 Introducción 47
3.2 La señal de voz 48
3.2.1 Diseño de la base de datos de voz 49

3.2.2 Pre-procesado de la señal 52
3.2.3 Etiquetador 56
3.3 Diseño de la base de datos de texto manuscrito 57
3.3.1 Preprocesado de la señal 57
3.3.2 Extracción de parámetros 57
3.4 Diseño de la base de datos de firmas 59

4 Mejora de la capacidad de la generalización de los HMMs **61**
4.1 Introducción 61
4.2 Capacidad de generalización 63
4.3 Estructura de los modelos HMMs 64
4.3.1 Símbolos de observación 65
4.3.2 Número de estados 67
4.4 Inicialización de los modelos HMM 72
4.4.1 Métodos de inicialización 73
4.4.2 Inicialización de igual ocupación de estados 75
4.5 Criterios de parada 78
4.5.1 Experimentos y resultados 80
4.5.2 Texto manuscrito 81
4.5.3 Firmas 81
4.6 Dependencia con los datos de entrenamiento 82
4.6.1 Validación cruzada aleatoria 84
4.6.2 Resultados y experimentos 84
4.7 Estudio del cálculo computacional/tasa de reconocimiento 88
4.7.1 Experimentos y resultados 88

5 Sistemas Híbridos: HMM/NN **91**
5.1 Introdución 91
5.2 Modelo híbrido 92
5.2.1 Modelos ocultos de Markov del modelo híbrido 93
5.2.2 Estructura neuronal del modelo híbrido 94
5.3 Híbrido HMM con estructura neuronal MLP ACON 95
5.3.1 Experimentos y resultados 97
5.4 Híbrido HMM con estructura neuronal RBF ACON 100
5.4.1 Experimentos y resultados 101
5.5 Híbrido con comités de redes neuronales 104
5.5.1 Híbrido HMM con comité de MLPs ACON 105
5.5.2 Experimentos y resultados 106
5.6 Híbrido HMM con estructura neuronal modular MLP OCON 108
5.6.1 Experimentos y resultados 109

5.7 Híbrido HMM con estructura neuronal modular de redes polinómicas, GMDH . . 112
5.7.1 Experimentos y resultados 117
5.8 Híbrido HMM con estructura neuronal polinómica 120
5.8.1 Resultados y experimentos 123
5.9 Combinación de HMMs con el método de *voting model* 125
5.9.1 Experimentos y resultados 126
5.10 Estudio del cálculo computacional/tasa de reconocimiento 128

6 Conclusiones y Líneas Futuras **129**
6.1 Conclusiones 129
6.2 Líneas Futuras 132

Índice de Figuras

2.1 Diagrama de bloques de un cuantificador vectorial. 14
2.2 División del espacio de dimensión 2 en 8 celdas. 15
2.3 División del espacio de dimensión 1 en 8 regiones 16
2.4 Función *warping* y ajuste de la ventana. 20
2.5 Red de 7 puntos. Hay que encontrar el camino óptimo entre el nodo 1 y el 7. 21
2.6 Red de 7 puntos. Hay que encontrar el camino óptimo entre el nodo 1 y el 7. 23
2.7 Nodo de una Red Neuronal. 25
2.8 Estructura típica de una Red neuronal. 26
2.9 Estructura típica de una MLP. 28
2.10 Estructura típica de una RBF. 31
2.11 Nodo o elemento de proceso. 35
2.12 Formación de la primera capa de la Red GMDH. 36
2.13 Red GMDH para 4 entradas. 37
2.14 Ejemplo de una Red básica TDNN. 40
2.15 Red TDNN para un reconocedor de fonemas. 40
2.16 Modelos de Ocultos de Markov. 41
2.17 Gráfico de estados de los HMM 42

3.1 Forma de onda de un fichero de grabación. 51
3.2 Diagrama de bloques del procesado de la señal. 53
3.3 Función de transferencia del filtro de pre-énfasis. 54
3.4 Etapa de preprocesado de imagen. 58
3.5 Trayectoria. Ejemplo de vectorización 59
3.6 Esqueletización de una firma. 60
3.7 Ejemplo de hojas de firmas. 60

4.1 Tasa de reconocimiento de voz en función de la biblioteca del Cuantificador Vectorial. 66
4.2 Varianza de la tasa de reconocimiento de voz en función de la biblioteca del Cuantificador Vectorial. 67
4.3 Ejemplo de vectorización de una trayectoria de un trazo. 68
4.4 Media de la tasa de reconocimento de voz para la secuencia de test en función del número de estados. 69

4.5 Varianza de la tasa de reconocimento para la secuencia de test en función del número de estados. 70
4.6 Media de la tasa de reconocimento de firmas para la secuencia de test en función del número de estados. 71
4.7 Media de la tasa de reconocimento de firmas para la secuencia de test en función del número de estados. 72
4.8 Tasa de reconocimiento/Carga computacional para distintos tipos de inicialización y criterios de parada 89
4.9 Tasa de reconocimiento/Carga computacional para modelos inicializados con el método igual ocupación de estados.) 90

5.1 Modelo Híbrido: HMM y Estructura Neuronal. 92
5.2 Modelos ocultos de Markov vistos como una estructura neuronal. 94
5.3 Función de transferencia sigmoidal 96
5.4 Modelo Híbrido: HMM/MLP 97
5.5 Modelo Híbrido: HMM/Comités MLP 107
5.6 Modelo Híbrido: HMM con Estructura Neuronal Modular MLP OCON 109
5.7 Nodo o elemento de proceso 114
5.8 Error Cuadrático medio por nodos, capa 1 116
5.9 MSE para la segunda capa 116
5.10 MSE para la tercera capa 117
5.11 Estructura de la red GMDH para el dígito 0 118
5.12 Nodo o elemento de proceso 121
5.13 Estructura de la Red GMDH 122

Índice de Tablas

2.1 Algoritmo de entrenamiento del VQ 17
2.2 Variables del algoritmo de aprendizaje. 29
2.3 Algoritmo de Backpropagation. 32
2.4 Algoritmo LMS para la capa de salida de la RBF. 34
2.5 Algoritmo de cálculo de los coeficientes de la red GMDH. 38
2.6 Procedimiento de poda de la red GMDH. 38

3.1 Parámetros de grabación de una tarjeta de sonido SOUND BLASTER 16 ASP 50

4.1 Tasa de Reconocimiento para distintas Bibliotecas de tamaño M para la secuencia de test. 65
4.2 Tasa de reconocimiento de voz para distintas bibliotecas de tamaño M y distintos estados N para la secuencia de test. 69
4.3 Comparación de los resultados obtenidos con un reconocedor de voz de la inicialización con diferentes métodos. 74
4.4 Algoritmo de inicialización. 75
4.5 Algoritmo de inicialización Igual Ocupación de Estados. 76
4.6 Resultados obtenidos con el método de inicialización Igual Ocupación de Estados (IOE). 76
4.7 Comparación de los métodos de inicialización en el reconocimiento de voz. 77
4.8 Comparación de los métodos de inicialización para la aplicación de texto manuscrito 78
4.9 Comparación de los métodos de inicialización para la aplicación de firmas. 78
4.10 Tasa de reconocimiento de voz aplicando diferentes criterios de parada (los HMM están inicializados aleatoriamente). 80
4.11 Tasa de reconocimiento de voz aplicando diferentes criterios de parada, (los HMMs están inicialización con el método igual ocupación de estados (IOE)). 80
4.12 Tasa de reconocimiento de texto manuscrito aplicando diferentes criterios de parada, (HMMs están inicializados aleatoriamente). 81
4.13 Tasa de reconocimiento de texto manuscrito aplicando diferentes criterios de parada, (HMMs están inicializados con el método IOE). 81
4.14 Tasa de reconocimiento de firmas aplicando diferentes criterios de parada, (HMMs están inicializados aleatoriamente). 82
4.15 Tasa de reconocimiento de texto manuscrito aplicando diferentes criterios de parada, (HMMs están inicializados con el método IOE. 82

4.16 Tasa de reconocimiento de voz con validación cruzada (VC) aplicando dos inicializaciones: aleatoria (A) e igual ocupación de estados (IOE); y criterios de parada: umbral y por máximo de iteraciones igual a 4. 85
4.17 Tasa de reconocimiento de voz con validación cruzada aleatoria (VCA) aplicando dos inicializaciones: aleatoria (A) e igual ocupación de estados,(IOE) ; y diferentes criterios de parada: umbral y por máximo de iteraciones igual a 4. 85
4.18 Comparación de las tasas de reconocimiento de voz con el método de inicialización, IOE, para entrenamientos que utilizan validación cruzada y validación cruzada aleatoria. 85
4.19 Tasa de reconocimiento de texto manuscrito aplicando validación cruzada (VC), con inicializaciones: aleatorio (A) e igual ocupación de estados (IOE); y con criterios de parada por umbral. 86
4.20 Tasa de reconocimiento de texto manuscrito aplicando validación cruzada aleatoria (VCA), con inicializaciones: aleatorio (A) e igual ocupación de estados (IOE); y con criterios de parada por umbral. 86
4.21 Comparación de las tasas de reconocimientode texto manuscrito con el método de inicialización, IOE, para entrenamientos que utilizan validación cruzada y validación cruzada aleatoria. 87
4.22 Tasa de reconocimiento de firmas aplicando validación cruzada (VC), con inicializaciones: aleatorio (A) e igual ocupación de estados (IOE); y con criterios de parada por umbral. 87
4.23 Tasa de reconocimiento de firmas aplicando validación cruzada aleatoria (VCA), con inicializaciones: aleatorio (A) e igual ocupación de estados (IOE); y con criterios de parada por: umbral. 87
4.24 Comparación de las tasas de reconocimiento con el método de inicialización, IOE, para entrenamientos que utilizan validación cruzada y validación cruzada aleatoria. . 87
4.25 Tabla comparativa del cálculo computacional con la tasa de reconocimiento para los distintos tipos de inicialización y criterios de parada. 88
4.26 Tabla comparatativa del cálculo computacional con la tasa de reconocimiento para los distintos criterios de parada con modelos inicializados con el método igual ocupación de estados (IOE). 89

5.1 Tasa de reconocimiento de voz del modelo híbrido HMM/MLP ACON. Los HMMs inicializados aleatoriamente). 98
5.2 Tasa de reconocimiento de voz del modelo híbrido HMM/MLP ACON. Los HMMs inicializados con el método de igual ocupación de estados, (IOE). 98
5.3 Tasa de reconocimiento de texto manuscrito del modelo híbrido HMM/MLP ACON. Los HMMs están inicializados aleatoriamente. 99
5.4 Tasa de reconocimiento de texto manuscrito del modelo híbrido HMM/MLP ACON. Los HMMs están inicializados con el método igual ocupación de estados. 99
5.5 Tasa de reconocimiento de firmas del modelo híbrido HMM/MLP ACON. Los HMMs inicializados aleatoriamente 100
5.6 Tasa de reconocimiento de firmas del modelo híbrido HMM/MLP ACON. Los HMMs inicializados con el método igual ocupación de estados 100

5.7 Tasa de reconocimiento de voz del modelo híbrido HMM/RBF. Los HMM han sido inicializados aleatoriamente. 102
5.8 Tasa de reconocimiento de voz del modelo híbrido HMM/RBF. Los HMMs han sido inicializados con el método de igual ocupación de estados (IOE). 102
5.9 Tasa de reconocimiento de texto manuscrito del modelo híbrido HMM/RBF. Los HMM han sido inicializados aleatoriamente. 102
5.10 Tasa de reconocimiento de texto manuscrito del modelo híbrido HMM/RBF. Los HMMs han sido inicializados con el método de igual ocupación de estados (IOE). . . 103
5.11 Tasa de reconocimiento de firmas del modelo híbrido HMM/RBF. Los HMM han sido inicializados aleatoriamente. 103
5.12 Tasa de reconocimiento de firmas del modelo híbrido HMM/RBF. Los HMMs han sido inicializados con el método de igual ocupación de estados (IOE). 103
5.13 Tasa de reconocimiento de voz del modelo híbrido HMM/Comité MLP aplicando distintos números de MLPs. Los HMMs están inicializados aleatoriamente. 107
5.14 Tasa de reconocimiento de voz del modelo híbrido HMM/Comité MLP aplicando distintos números de MLPs. Los HMMs están inicializados con el método de igual ocupación de estados. 108
5.15 Tasa de reconocimiento de voz del modelo híbrido HMM/Comité MLP aplicando los distintos métodos. 108
5.16 Tasa de reconocimiento voz del modelo híbrido HMM/MLP OCON. Los HMMs inicializados aleatoriamente. 110
5.17 Tasa de reconocimiento de voz del modelo híbrido HMM/MLP OCON. Los HMMs inicializados con el método IOE. 110
5.18 Tasa de reconocimiento de texto manuscrito del modelo híbrido HMM/MLP OCON. Los HMMs inicializados aleatoriamente. 111
5.19 Tasa de reconocimiento de texto manuscrito del modelo híbrido HMM/MLP OCON. Los HMMs inicializados con el método IOE. 111
5.20 Tasa de reconocimiento de firmas del modelo híbrido HMM/MLP OCON. Los HMMs inicializados aleatoriamente. 112
5.21 Tasa de reconocimiento de firmas del modelo híbrido HMM/MLP OCON. Los HMMs inicializados con el método IOE. 112
5.22 Tasa de reconocimiento de voz del modelo híbrido HMM/GMDH. Los HMMs están inicilalizados aleatoriamente. 119
5.23 Tasa de reconocimiento de voz del modelo híbrido HMM/GMDH. Los HMMs han sido inicializados con el método IOE 119
5.24 Tasa de reconocimiento de texto manuscrito del modelo híbrido HMM/GMDH. Los HMMs inicializados aleatoriamente. 119
5.25 Tasa de reconocimiento de texto manuscrito del modelo híbrido HMM/GMDH. Los HMMs inicializados con el método IOE. 120
5.26 Tasa de reconocimiento de firmas del modelo híbrido HMM/GMDH. Los HMMs inicializados aleatoriamente. 120
5.27 Tasa de reconocimiento de firmas del modelo híbrido HMM/GMDH. Los HMMs inicializados con el método IOE. 120

5.28 Tasa de reconocimiento de voz del modelo híbrido HMM/Red GMDH. Los HMMs están inicilalizados aleatoriamente. 123
5.29 Tasa de reconocimiento del modelo híbrido HMM/Red GMDH. Los HMMs han sido inicializados con el método IOE 124
5.30 Tasa de reconocimiento de texto manuscrito del modelo híbrido HMM/Red GMDH. Los HMMs inicializados aleatoriamente. 124
5.31 Tasa de reconocimiento de texto manuscrito del modelo híbrido HMM/Red GMDH. Los HMMs inicializados con el método IOE. 124
5.32 Tasa de reconocimiento de firmas del modelo híbrido HMM/Red GMDH. Los HMMs inicializados aleatoriamente. 125
5.33 Tasa de reconocimiento de firmas del modelo híbrido HMM/Red GMDH. Los HMMs inicializados con el método IOE. 125
5.34 Tasa de reconocimiento del comité de modelos HMMs inicilalizados aleatoriamente. . 127
5.35 Tasa de reconocimiento de voz del comité de modelos HMMs inicilalizados con el método IOE. 127
5.36 Estudio comparativo de los distintos modelos híbridos. Los modelos HMMs están inicilalizados aleatoriamente. 128
5.37 Estudio comparativo de los distintos modelos híbridos. Los modelos HMMs están inicilalizados con el método IOE. 128

Capítulo 1

Introducción

1.1 Antecedentes

Algunos fenómenos del mundo real originan señales o bien éstos se pueden cuantificar en señales que describen su comportamiento. Estas señales pueden ser de naturaleza discreta o continua, pueden clasificarse como señales estacionarias (señales cuyas propiedades estadísticas no varían con el tiempo), o no estacionarias (señales cuyas propiedades estadísticas varían con el tiempo); también pueden ser vistas como una secuencia de muestras aleatorias, donde el orden de las muestras pueden o no influir en el resultado.

El conocimiento sobre un fenómeno, la extracción de información a partir de las señales, el estudio de su comportamiento, de su evolución, de la posibilidad de predicción, de crear modelos que simulen un determinado fenómeno, ha sido el motivo de la creación de diversas disciplinas y áreas de investigación.

El tema de investigación de esta tesis, está centrado en la **clasificación de secuencias temporales**. El término de **clasificación** significa: asignar una localización, una posición, una ordenación, en definitiva una *clase* a un conjunto de cosas en virtud de una serie de características comunes, de tamaño, color, tipo, tema, etc. El término de **secuencia temporal** hace referencia a señales donde la variable tiempo es importante, o la secuencia con la que se van originando los datos ha de tenerse en cuenta.

Para el ser humano, el hecho de clasificar, por ejemplo 20 lápices de colores en distintas cajas por color, o por tamaño, no es tan complicado. Si se desea automatizar este proceso, quizás el problema se complique un poco más. Pero si lo que se pretende es automatizar un proceso de clasificación de una señal compleja, como lo es la señal de voz, de texto manuscrito o firmas, el problema es aún más complicado.

Para el reconocimiento de voz, el modelo a diseñar ha de tener en cuenta muchísimos factores, tales como el estado anímico, ruido ambiental, velocidad, variaciones a la hora de hablar, etc. Una misma palabra pronunciada por el mismo locutor, en distintos momentos puede tener distinta tonalidad, distinta duración, distintos ruidos ambientales, etc. Para las aplicaciones de texto manuscrito o firmas, hay que tener en cuenta la presión ejercida a la hora de escribir, la trayectoria de los trazos, la duración, el tamaño, la escritura de los escritores puede sufrir variaciones en determinadas situaciones: tiempo, estado de ánimo, la intencionalidad, etc.

Encontrar un buen modelo de clasificación, supone obtener suficiente información del fenómeno que se desea analizar. Estos modelos se estudian a través de las simulaciones. La experiencia demuestra que funcionan bastante bien en la práctica, como ejemplo citamos sistemas de predicción, sistemas de reconocimiento, sistemas de identificación, etc. Para que un modelo de clasificación trabaje adecuadamente ha de ser enseñado previamente, se le ha de proporcionar información. La fase donde el modelo es enseñado para que aprenda un determinado problema se denomina **entrenamiento**. La fase de comprobación del modelo se denomina **test**. El funcionamiento del modelo es sensible a la cantidad de información dada durante el entrenamiento.

Los sistemas de clasificación se dividen en dos tipos dependiendo del tipo de señal a trabajar: clasificadores estáticos y dinámicos. Los primeros se utilizan para clasificaciones de señales que no varían con el tiempo, o lo que es lo mismo la información temporal no es necesaria. Los segundos si tienen en cuenta el carácter temporal de la señal, siendo éstos el centro de nuestro estudio. Aunque también se aprovecharán las capacidades de los primeros.

Para comenzar, se ha de elegir una señal cuyas características sean dependientes de la variable tiempo. La señal de **voz** es un buen ejemplo y además se trata de una señal de la cual se dispone de bastante información tanto a nivel temporal como espectral. Los primeros trabajos se realizaron con ella. Posteriormente gracias a la elaboración de dos proyectos fin de carrera [Ciro99-pfc] y [Camino99-pfc], de la Escuela Técnica Superior de Ingeniera de Telecomunicación de la Universidad de Las Palmas de Gran Canaria, se pudieron realizar los mismos experimentos con otras señales: reconocimiento de texto manuscrito y de firmas.

Hay muchas otras señales del mundo real, tales como imagen, radar, señales médicas, señales de control, etc. que dependen del tiempo. Por lo tanto, los experimentos llevados a cabo en esta tesis pueden ser aplicables a cualquier señal no estacionaria imponiendo las condiciones de contorno para cada una de ellas. La señal por excelencia de los experimentos llevados a cabo utilizan la señal de voz. Aunque también se añaden pruebas con otro tipo de señales, lo que nos permitirá estudiar la generalidad de las distintas técnicas y modelos que aquí se proponen.

Uno de los trabajos de esta tesis ha sido elegir dentro de los modelos de clasificación existentes, aquél que trabaja con secuencias temporales, de manera eficiente, como lo son

los modelos ocultos de Markov y optimizarlos. Es decir, analizar aspectos que todavía no han sido estudiados, tales como: criterios de parada, métodos de inicialización, dependencia de los datos de entrenamiento, estructura de los modelos, etc. aspectos que afectan a la capacidad de generalización del modelo. Por otro lado, se proponen combinaciones de sistemas tales como los modelos ocultos de Markov, (HMM), y redes neuronales, (NN), formando nuevas estructuras híbridas con el fin de aprovechar las características de ambos mejorando a los sistemas clásicos cuando éstos se encuentran trabando de manera independiente.

En el siguiente apartado, planteamos de manera más concreta los objetivos de esta tesis. Seguidamente a los objetivos, se realizará una breve revisión histórica sobre los métodos aplicados al reconocimiento de voz. No debemos olvidar que el reconocimiento de voz, es una tarea de clasificación de secuencias temporales y es una de las áreas de investigación donde se ha invertido en los últimos 40 años, un gran esfuerzo.

1.2 Objetivos

La experiencia demuestra que el grado de una correcta clasificación no es siempre satisfactoria, debido a muchas causas, tales como: falta de generalización, modelos iniciales, criterios de parada, técnicas de entrenamiento, los patrones de entrada pueden sufrir algún tipo de distorsión, etc.

Uno de los métodos más eficientes hoy en día utilizados en reconocimiento de voz son los modelos ocultos de Markov. Este método es capaz de modelar bastante bien el comportamiento temporal de la señal de voz. Uno de los objetivos planteados fue estudiar algunos aspectos de los modelos ocultos de Markov, susceptibles de mejora y sobre todo analizar como influyen en su capacidad de generalización medido a partir de la tasa de reconocimiento. Aspectos tales son: estructura del modelo, criterios de parada, inicialización, influencia de los datos suministrados en el entrenamiento, etc. Así como en las redes neuronales podemos encontrar una gran variedad de bibliografía acerca de la optimización en cuanto a su estructura (técnicas de crecimiento, de poda etc.), métodos de inicialización, algoritmos de entrenamientos etc. [Maren90], [Haykin94]. No podemos decir lo mismo sobre los modelos ocultos de Markov. Es verdad, que en [Rabi92], se comenta ciertas ideas orientativas sobre estructura, inicialización de los modelos, entrenamiento, etc. pero en ningún momento se realiza un estudio exhaustivo acerca del tema. Después de una amplia consulta bibliográfica, no se encontró ningún trabajo acerca de la óptimización de los modelos HMM en cuanto a parámetros como número de estados, símbolos de observación, métodos de inicialización, criterios de parada, dependencia con los datos de entrenamiento, etc. y cómo podían afectar a la capacidad de generalización de los modelos HMMs. Ésto dio pie al **primer objetivo** planteado, *optimización de los modelos HMMs*, buscar las condiciones óptimas para el reconocimiento de voz.

A pesar de que los modelos ocultos de Markov modelen el carácter temporal de la señal, sin embargo, suposiciones tales como *independencia de las observaciones con los estados* e *independencia de las probabilidades de transición* hacen que estén limitados a la hora de clasificar. Así como las redes neuronales, en lo sucesivo NN, presentan una gran habilidad de discriminación o clasificación, sin embargo no tienen la cualidad de ser un método efectivo para el modelado temporal de la señal. Una posible solución al problema en cuestión es la aplicación de transformaciones de esos patrones temporales a estáticos, para así aprovechar las capacidades que presentan los diferentes métodos: modelado temporal y clasificación. Ésto puede ser probablemente una estrategia aceptable para resolver el problema que planteamos. Éste es el principio que se ha aplicado en el planteamiento de estructuras híbridas.

De aquí surge el **segundo objetivo** de la tesis, con la combinación de las capacidades del modelado temporal de los HMMs y de discriminación de las NNs, se llega a lo que actualmente se conoce como sistemas híbridos, HMM/NN. En los últimos años se han propuesto diferentes sistemas; por ejemplo en [Renals94], una NN es entrenada independientemente como estimadora de probabilidades; en [Robison94] se utiliza un procedimiento similar al anterior pero con redes recurrentes; otros autores proponen arquitecturas donde todos los parámetros son estimados simultáneamente; otros aplican transformaciones adaptativas de los vectores de observación, etc. En esta tesis se propone una estructura híbrida sencilla donde los HMMs tienen la finalidad de tratar el carácter temporal de la señal y la red neuronal la de discriminar.

Por último, el comienzo de esta tesis se planteó con el objeto de trabajar con la señal de voz, ya que se disponía de bastante información tanto a nivel temporal como espectral. El disponer de otras señales o fuentes de datos cuya evolución dependiese de la variable tiempo, o la secuencia de las muestras fuese importante, consolidaría los experimentos llevados a cabo. A medida que se avanzó en el tema de clasificación y reconocimiento, se vio la posibilidad de trabajar con otras señales diferentes a la de voz. De aquí surge el **tercer y último objetivo**: comprobar las técnicas de optimización aplicadas a los HMMs, además de estudiar el comportamiento de las estructuras híbridas propuestas a otras aplicaciones de clasificación temporal como lo son el reconocimiento de texto manuscrito o el reconocimiento de firmas.

Resumiendo, la tesis que aquí se presenta pretende, optimizar los modelos ocultos de Markov muy utilizados en reconocimiento de voz y combinarlos con otros sistemas de clasificación como las redes neuronales con el fin de aumentar la capacidad de generalización de los modelos. Estos modelos de clasificación han sido obtenidos a partir de conjuntos de entrenamiento de tamaño reducido.

Para concluir, los objetivos planteados en esta tesis son:

1. Estudiar algunos aspectos de los modelos ocultos de Markov susceptibles de mejora y sobre todo analizar como influyen en su capacidad de generalización medido a partir de la tasa de reconocimiento. Aspectos tales son: estructura del modelo, criterios

de parada, inicialización, influencia de los datos suministrados en el entrenamiento, etc.

2. Buscar nuevas estructuras denominadas híbridas, fruto de la combinación de modelos de clasificación, que mejoraran la tasa de clasificación. La estructura híbrida que se propone es el resultado de combinar los modelos ocultos de Markov y las redes neuronales con el fin de aprovechar las capacidades de ambos modelos.

3. El último de los objetivos planteados es el de trabajar con otro tipo de señales diferentes a la señal de voz, como texto manuscrito y firmas. Estas señales se codificaron como una secuencia de números que definen la trayectoria de los trazos. El orden de las muestras es importantísimo y ésto es lo que las hace apropiadas para trabajar con los modelos propuestos, con las mejoras introducidas a los HMMs y las distintas estructuras híbridas. Gracias a la elaboración de los proyectos fin de carrera [Ciro99-pfc] y [Camino99-pfc] se dispuso de otras fuentes de datos procedentes de texto manuscrito y firmas respectivamente. Realizar los mismos experimentos y comprobar que las técnicas de optimización propuestas y la aplicación de las estructuras híbridas corroboraban lo obtenido para la señal de voz suponía conseguir la validez de los métodos propuestos.

Para llevar a cabo los objetivos anteriormente expuestos fue necesario crear todo un *banco de trabajo*: base de datos, (grabaciones de los dígitos del 0 al 9 de diferentes locutores en castellano, recogida de texto manuscrito, recogida de firmas en distintos espacios reservados para ello), algoritmos y programas (se desarrollaron programas para el etiquetado de la señal de voz, conversión de las imágenes escaneadas en secuencias de observación, preprocesado, etc.), además se implementaron y se verificaron los algoritmos HMMs expuestos en [Rabi92]). Todos los programas fueron desarrollados para trabajar dentro del entorno de MATLAB.

1.3 Evolución de los clasificadores aplicados a voz

Atendiendo a que el reconocimiento de voz es un método de clasificación temporal, dedicamos este apartado a comentar brevemente los acontecimientos más importantes cronológicamente sobre las distintas técnicas, modelos, etc. que se han aplicado al reconocimiento de voz. Se observará dos etapas diferenciadas. La primera de ellas se basó en la aplicación de técnicas espectrales. Y la segunda etapa cuando se comenzó a aplicar el modelado temporal con la utilización de técnicas de alineamiento temporal y a tener en cuenta información estadística de la señal, como así lo hacen los modelos ocultos de Markov. Se hará una breve revisión de los acontecimientos y avances obtenidos en este área: aplicación de técnicas espectrales, medidas de predicción lineal, técnicas de alineamiento temporal, introducción de información estadística en los modelos, etc. Estos hechos se tratan de manera extensiva en [Rabi93].

El reconocimiento de voz es un tema de investigación que comenzó a crecer en los laboratorios de los años 50. El primer reconocedor fue realizado en los laboratorios Bell con los conocimientos de fonética, acústica y lo que la tecnología del momento les permitía desarrollar. En el año 1956, el laboratorio RCA, con independencia de los laboratorios Bell, realizaba pruebas de un reconocedor de 10 sílabas, sistema que utilizaba medidas espectrales realizadas con un banco de filtros analógicos. Fue en este momento cuando se comenzó a trabajar con medidas espectrales de la señal de voz.

Tres años más tarde en la Universidad de *College* en Inglaterra, se realizaba un reconocedor de fonemas, (cuatro vocales y nueve consonantes), el cual utilizaba un analizador de espectros y trataban de establecer una correspondencia entre los distintos patrones para delimitar las regiones de decisión. Una importante introducción en este sistema, fue la utilización de técnicas estadísticas. Aparece aquí, el concepto de correspondencia, de clasificación, sumándose además la componente estadística al modelo.

En el mismo año, en los laboratorios MIT *Lincoln* se trabajaba en el reconocimiento de locutores. Nuevamente se utilizaba un banco de filtros para obtener información espectral y estimaciones sobre variaciones en el tiempo del tracto vocal. Aparece aquí por primera vez, el estudio de variaciones en el tiempo aplicado a los reconocedores.

Pero fue en los años 60, cuando aparecen las primeras publicaciones y artículos sobre los trabajos llevados a cabo en reconocimiento de voz. Fue en este momento, cuando varios laboratorios japoneses comenzaron a trabajar y construir hardware específico sobre reconocimiento de voz. Uno de los primeros reconocedores de vocales fue diseñado por Suzuki y Nakata del laboratorio Radio Research en Tokio. El reconocedor diseñado utilizaba un analizador de espectros de bancos de filtros donde la salida de cada canal del analizador estaba conectado a un circuito de decisión de vocales. Otro diseño hardware llevado a cabo en Japón, fue el realizado por Sakai y Doshita en la Universidad de Kyoto en 1962, el cual está basado en un reconocedor de fonemas. Un tercer sistema fue el reconocedor de dígitos de Nagata diseñado junto con los laboratorios NEC de Japón en el año de 1963.

Paralelamente a los hechos anteriores, durante los años 60, se iniciaron tres proyectos basados en la investigación y desarrollo de los reconocedores de voz. El primero de ellos de llevó a cabo en los laboratorios RCA, su objetivo era el de intentar dar soluciones reales a los problemas ocasionados por la no uniformidad en la escala de tiempo de los eventos de voz. Para ello se desarrollaron métodos de normalización del tiempo basados en la capacidad de detectar el comienzo y fin.

El segundo de ellos se llevaba a cabo en la Unión Soviética, el cual trabajaba en el uso de la *programación dinámica* para el alineamiento de la voz. Aunque el concepto de programación dinámica no sale a la luz para el resto del mundo hasta los años 80.

El tercer proyecto es el trabajo pionero de Reddy sobre el reconocimiento continuo de voz.

Durante los años 70 se consiguieron grandes avances, Velichko y Zagoruyko en Rusia, Sakoe y Chiba en Japón e Itakura en EE.UU. Los rusos ayudaron en las ideas de reconocimiento de patrones. Los japoneses demostraron cómo aplicar las técnicas de programación dinámica. Y en EE.UU con las investigaciones de Itakura sobre la predicción lineal, LPC aplicado al reconocimiento. Todo ésto, supuso un avance muy importante en el campo del reconocimiento de voz.

Finalmente, los laboratorios de AT&T introdujeron en sus líneas de investigación el reconocimiento independiente de locutor aplicando técnicas de *clustering*. Siendo en los años 80, cuando la tecnología permitió la aplicación de modelos estadísticos especialmente los modelos ocultos de Markov, HMM, cuya repercusión fue de gran importancia en este campo. De hecho, se trata del modelo más utilizado como clasificador de secuencias temporales. Y es aquí cuando se inicia el modelado temporal de la señal de voz.

Por esta misma fecha, se trabajó sobre la aplicación de redes neuronales (NN) a problemas de reconocimiento, aunque ya se había intentado su utilización en los años 50, aunque sin éxito debido a problemas prácticos.

La introducción de los HMMs permitió la aplicación del reconocimiento de voz a vocabularios grandes en sistemas de habla continuo por la Agencia de Proyectos de Investigación Avanzados de Defensa (DARPA).

A partir de aquí, hay muchas y variadas aportaciones para intentar optimizar los sistemas así como de búsqueda de nuevas estructuras que mejoren los sistemas de reconocimiento de voz, los cuales se pueden aplicar a la clasificación de secuencias temporales.

Se puede concluir que los modelos de clasificación de secuencias temporales aplicables a la señal de voz, ha supuesto un conjunto de estudios centrados en:

1. Características espectrales de la señal de voz, con técnicas de predicción lineal.
2. Modelos de correspondencia o patrones.
3. Alineamiento temporal de patrones.
4. Modelado estadístico del carácter temporal de la señal de voz.

El reconocimiento de voz es fundamentalmente un problema de clasificación. Si los patrones de entrada o señal de voz fuesen invariantes, el problema sería muy sencillo: simplemente compararíamos los patrones de entrada con los almacenados y eligiríamos los que presentasen una menor distorsión. La verdadera dificultad del reconocimiento de voz es que la señal de voz es muy variable.

Todo ésto nos conduce a que los sistemas de clasificación de secuencias temporales conllevan: una fase de preprocesado de la señal y la aplicación de técnicas de alineamiento temporal con la utilización de modelos estadísticos.

En [Roe93] se realiza una breve descripción de los modelos de clasificación temporal que se han utilizado y se continuan utilizando aplicados al reconocimiento.

Los clasificadores temporales más utilizados son: clasificadores de patrones, sistemas basados en reglas de decisión, redes neuronales, NN (Neural Network) y modelos ocultos de Markov, HMM (Hidden Markov Model), etc.:

1. **Clasificadores de patrones.** Para los clasificadores de patrones, la secuencia de patrones se alinea temporalmente con una secuencia óptima de patrones. La idea central es comparar cada patrón de entrada con los almacenados en una librería y encontrar aquel patrón que presente menor distorsión. Como la velocidad del habla varía de una grabación a otra, se utiliza la técnica de DTW, (Dynamic Time Warping), la cual estira o encoge la señal de entrada, adaptando el eje del tiempo para minimizar la distorsión de la secuencia de patrones de entrada.

2. **Sistemas basados en reglas de decisición.** Se basan en la aplicación de una serie de criterios para construir un árbol de decisión, el cual determina qué unidades del lenguaje están presentes en la señal de voz. Cuando el sistema de reconocimiento es muy complejo, estos sistemas han demostrado ser ineficaces ya que es muy díficil crear o establecer el conjunto de reglas que generalicen adecuadamente las variaciones de la señal de voz.

3. **Redes Neuronales.** Son arquitecturas que tanto por su alto grado de interconectividad como por su modo de entrenamiento supervisado tienen una alta capacidad de discriminación entre clases. El problema es, su dificultad para modelar el alineamiento temporal de la señal de voz. Ésto es por lo que las NNs se usan con mucha frecuencia como discriminadores estáticos. Para este tipo de aplicaciones habría que recurrir a redes neuronales recurrentes o redes estáticas combinadas con otros modelos o aplicando técnicas que trabajen el carácter temporal de la señal.

4. **Modelos Ocultos de Markov, HMM.** Los HMMs, son actualmente el sistema más eficaz de reconocimiento. La principal ventaja es que son capaces de modelar el alineamiento temporal. Sin embargo presentan una escasa capacidad de discriminación debido en parte a su modo de entrenamiento no discriminativo.

5. **Input-Output HMM.** Los IOHMM, se trata de una estructura recurrente. Este modelo [Bengio96] presenta similitudes con los modelos de ocultos de Markov con la diferencia de que mapea la secuencia de entrada en una secuencia de salida utilizando redes neuronales recurrentes.

1.4 Estructura de la memoria

La memoria de la tesis se estructura básicamente en 5 partes o capítulos. A continuación se realiza una breve introducción de lo que se tratará en cada uno de ellos.

Capítulo 2

El capítulo 2, realiza una revisión del estado del arte sobre clasificación de secuencias temporales. En él se recogen los principales sistemas de clasificación: clasificadores de patrones, redes neuronales, modelos ocultos de Markov (HMM), etc. Se analizan además las ventajas e incovenientes de cada modelo a la hora de aplicarlos a secuencias temporales.

Capítulo 3

Para comenzar a trabajar en clasificación de secuencias temporales fue necesario crear *el entorno de trabajo*, es decir, base de datos de señales de voz, texto manuscrito, firmas, algoritmos, el estudio del preprocesado de las distintas señales, etc. Este capítulo se dedica con amplitud a tratar estos detalles.

Capítulo 4

En relación a los modelos clásicos de clasificación comentados en el capítulo 2, se eligieronn los modelos ocultos de Markov por ser los modelos más eficientes a la hora de trabajar con secuencias temporales. En este capítulo, se estudia la influencia de parámetros tales como: número de estados, símbolos de observación, conjunto de entrenamiento, inicialización, criterios de parada, etc., en la capacidad de generalización de los modelos. En definitiva, no es más que una optimización de los HMM.

Capítulo 5

Atendiendo al principio de mejorar la capacidad de generalización y aceptando que el entrenamiento de los modelos ocultos de Markov es un entrenamiento no discrimativo, se estudia la combinación de estos modelos con redes neuronales estáticas, con el fin de corregir aquellos errores que los modelos de Markov no son capaces de percibir. La combinación de estos dos modelos nos lleva a los modelos híbridos. En este capítulo se analiza distintas estructuras híbridas con distintas redes neuronales y topologías, etc.

Por último, finalizamos esta tesis con las conclusiones y líneas futuras de invesitigación y aplicación.

Capítulo 2

Estado del Arte. Clasificadores de Patrones

2.1 Introducción a los clasificadores

Uno de los objetivos básicos del estudio de la clasificación puede ser el crear un clasificador eficiente. Otro criterio importante, además de conseguir un buen clasificador, es tener alguna idea sobre la estructura de los datos que se van a manejar. Es obvio que para conocer si un clasificador es bueno o malo es necesario comprobar su funcionamiento y definir alguna variable que proporcione dicha información, como puede ser la **tasa de errores** o **tasa de reconocimiento**. Para llevar a cabo la aplicación de estos dos criterios se requiere de una base de datos a partir de la cual se formen dos conjuntos: uno de entrenamiento y otro de test. El primero le proporcionará información al sistema para que aprenda a clasificar y el segundo conjunto permitirá extraer información del sistema y conocer lo bueno o malo que éste es.

Una primera definición muy general de clasificación puede ser: *" Dado un conjunto de medidas sobre una magnitud física u objeto, se ha de encontrar una manera de predecir a qué clase pertenece"*

Otra definición matemática puede ser, ver [Breiman93]: *"Sea el espacio X y C el conjunto de clases, ésto puede ser visto también como el conjunto de todos los pares (x, j), donde $x \in X$ y $j \in C$. El espacio X se puede dividir en j subconjuntos disjuntos A_j de tal forma que $\bigcup_j A_j = X$, por lo que si $x \in A_j$, j será la clase a la que pertenezca"*

Ambas definiciones son muy generales y orientadas a sistemas de clasificación estáticos.

De aquí surgieron los primeros clasificadores utilizados en reconocimiento. Sin embargo, el problema que se encuentra en estos sistemas es que no tienen en cuenta el carácter temporal de la señal a clasificar, por lo que se han tenido que utilizar sistemas más complejos.

Hay clasificadores que requieren información de la salida que deberían dar en el entrenamiento. Este tipo de entrenamiento se conoce como entrenamiento supervisado. De igual forma, para poder estimar la calidad de un clasificador, se debe conocer la salida que debería dar, para comprobar los errores que comete ante un conjunto de patrones que no ha visto durante el entrenamiento. Se puede encontrar también clasificadores que no requieren de la salida ni en la fase de entrenamiento ni en la test, así como clasificadores mixtos, donde el aprendizaje es no supervisado y la fase de test requiera de la salida o viceversa.

La tarea de los clasificadores con entrenamiento supervisado se caracterizan por lo siguiente:

- Se dispone de un conjunto de objetos, los cuales están caracterizados por un vector de características de una determinada longitud y un conjunto de clases, tal que, a cada objeto se le asigna exactamente una clase.
- Se ha de tener un subconjunto de objetos, del que se sabe la clase que le corresponde a cada uno. A este subconjunto se le conoce como subconjunto de entrenamiento.
- El clasificador es entrenado para estimar las clases a la que pertenecen los objetos que no están dentro del subconjunto de entrenamiento.
- Se ha de definir una medida que indique lo óptimo o deficiente que es el clasificador. Esta medida puede ser la tasa de reconocimiento medido a partir de los objetos que han sido bien clasificados.

El vector de características que describe al objeto es frecuentemente nombrado como un patrón y a la tarea de clasificar, como reconocimiento de patrones. Estos patrones representan características o medidas obtenidas a partir del objeto. Nos podemos encontrar con aplicaciones donde la secuencia o el orden con la que se van generando esos patrones es importante porque es ahí donde se encuentra la información. Cuando esto ocurre, los patrones son denominados patrones o secuencias temporales.

A la hora de trabajar con patrones temporales se ha de encontrar un modelo o método capaz de modelar el carácter temporal de la señal. La solución sería aplicar métodos que modelen esa variación temporal. Igualmente, se podría plantear la solución del problema aplicando transformaciones de esos patrones, de temporales a estáticos, para así aprovechar las capacidades que presentan los clasificadores estáticos. Ésto puede ser, probablemente, una estrategia aceptable para resolver el problema que planteamos. Y siendo éste además, el principio que se ha aplicado a la hora de trabajar con estructuras híbridas.

En este capítulo se comenta brevemente los clasificadores ms utilizados en el reconocimiento de voz. Se incluyen tanto a los clasificadores estáticos como a los dinámicos, a pesar de que los primeros presenta limitaciones a la hora de trabajar con secuencias temporales. Sin embargo muchos de estos clasificadores estáticos sern utilizados en las estructuras híbridas que se proponen en esta tesis y éste es el motivo por el que se han incluído en este capítulo.

2.2 Clasificación de secuencias

Para comenzar el estudio se partió de los sistemas utilizados en reconocimiento de voz, ya que el reconocimiento es un problema de clasificación de secuencias temporales. El problema del reconocimiento de voz trata de buscar modelos que clasifiquen palabras o unidades fonéticas. Los modelos de clasificación utilizados son:

1. Clasificadores de patrones.
2. Técnicas de alineamiento temporal.
3. Sistemas basados en reglas de decisión.
4. Redes neuronales, NN (Neural Network).
5. Modelos ocultos de Markov, HMM (Hidden Markov Model).
6. IOHMM, Input-Output HMM.

A continuación se hará una breve descripción de ellos, en qué consisten, pero en ningún momento se ha pretendido hacer una exposición exhaustiva ya que para ello hay abundante bibliografía, sobre todo tutoriales que abordan cada uno de los sistemas de manera estricta.

2.3 Clasificadores de patrones

El sistema más sencillo y quizás uno de los primeros utilizados en reconocimiento es el método conocido como clasificadores de patrones. Este sistema se basa en comparar un patrón de entrada con otros de referencia que tiene almacenados en una biblioteca y elige aquél que presente la distorsión más pequeña. Este vector de la biblioteca puede hacer referencia a una clase determinada, con lo que el patrón de entrada queda perfectamente clasificado.

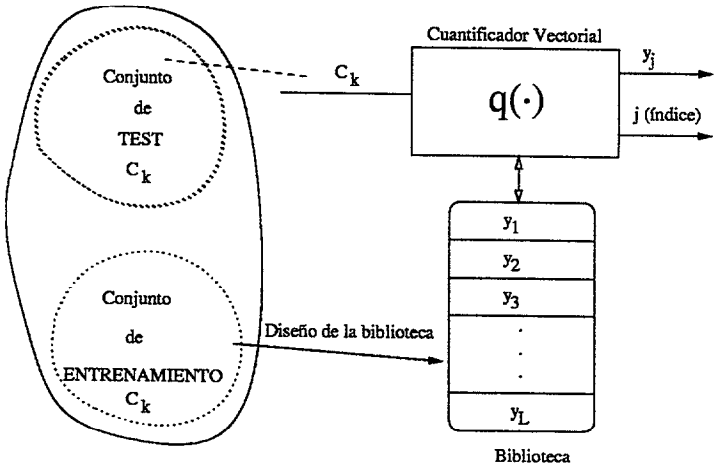

Figura 2.1: Diagrama de bloques de un cuantificador vectorial.

Un ejemplo de clasificador de patrones, muy utilizado en procesado de señal, es el cuantificador vectorial [Makhoul85], el cual se trata en el apartado siguiente.

2.3.1 Cuantificador vectorial

El cuantificador vectorial tiene dos estadios de trabajo:

1. Fase de entrenamiento. Esta fase se basa en encontrar el conjunto de patrones de referencia que constituya la biblioteca o *codebook* en terminología inglesa. Para calcular estos patrones de referencia se necesita de un conjunto de patrones de entrenamiento y aplicar algún algoritmo que seleccione o calcule la biblioteca con los patrones de entrenamiento.

2. Fase de test. Durante esta fase el vector de entrada se compará con todos los vectores de la biblioteca y se elegirá aquel vector de la biblioteca que presente menor distorsión.

En la figura 2.1 aparece el diagrama de bloques de un cuantificador vectorial. Del conjunto total de patrones disponibles se ha seleccionado un conjunto de patrones que conformarán el conjunto de entrenamiento. Con este conjunto más algún algoritmo, se diseña la biblioteca de patrones. Con la biblioteca diseñada, el resto de los patrones de entrada podrán ser cuantificados.

A continuación se analiza con más detalle el problema de la cuantificación vectorial.

Sea $X = [x_1, x_2, ..., x_N]^T$ el conjunto de vectores de dimensión N, a su vez cada componente $x_k \in X$, donde $1 \leq k \leq N$ puede ser un vector o un valor real. En el proceso

de la cuantificación vectorial, cada vector x_k es mapeado a otro vector $y_k \in Y$. Donde $Y = [y_1, y_2, ..., y_L]$ es el conjunto de vectores de referencia que forman la biblioteca de tamaño L.

Matemáticamente el funcionamiento del cuantificador vectorial vendría dado por la ecuación 2.1

$$y = q(x) \tag{2.1}$$

Donde $q(\cdot)$ es el operador cuantificador vectorial.

2.3.1.1 Fase de entrenamiento del VQ

Para diseñar una biblioteca, se divide el espacio de dimensión d en L regiones denominadas celdas o *centroides* y cada celda se le asocia un vector y_k. Las figuras 2.2 y 2.3, se puede ver este proceso.

En la figura 2.2 el espacio es de dimensión 2 y se ha elegido un tamaño de biblioteca de 8.

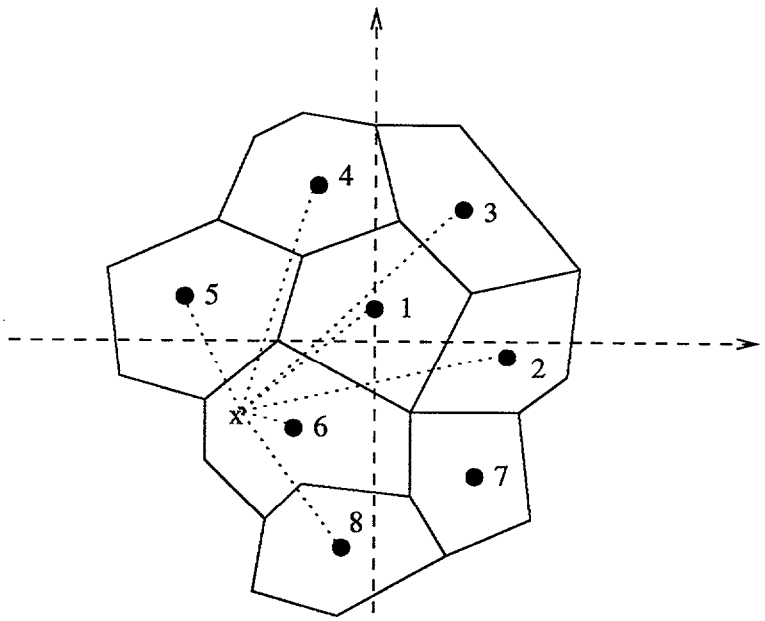

Figura 2.2: División del espacio de dimensión 2 en 8 celdas.

En la figura 2.2 el vector **x** se encuentra en la celda 6. Pero para saber que está en esa celda se mide la distancia que hay entre el vector **x** y todos los centroides. El vector **x** se asignará a la celda que presente la distancia menor, **celda número 6.**

Para la figura 2.3, el espacio es de dimensión 1 y el tamaño de la biblioteca es de 8.

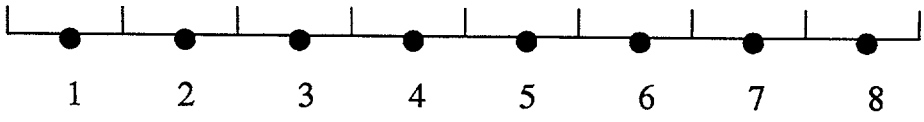

Figura 2.3: División del espacio de dimensión 1 en 8 regiones

Es evidente que el proceso de cuantificación conlleva a introducir un error o distorsión, que se le suele denominar medida de distancia, $d(x, y)$.

Las medidas de distancia más utilizadas en el procesado de señal son:

1. *Error cuadrático medio, MSE (Mean Square Error).* Se trata de la medida más común y más utilizada debido a la posibilidad de manejo y computo.

$$d(x,y) = \frac{1}{N}(x-y)^T \cdot (x-y) = \frac{1}{N}\sum_{k=1}^{N}(x_k - y_k)^2 \tag{2.2}$$

   Con esta medida se cuantifica por igual a todos los patrones de entrada.

2. *Error cuadrático medio ponderado, W-MSE (Weighted Mean Square Error).* En determinadas aplicaciones puede ser interesante que la medida de distancia no sea por igual para todos los patrones. Ésto se consigue añadiendo a la medida de distancia un factor de ponderanción.

$$d_W(x,y) = (x-y)^T \cdot W(x-y) \tag{2.3}$$

   Donde W es una matriz positiva y simétrica. En muchas aplicaciones de clasificación de patrones se elige W como la matriz de covarianza de un vector x aleatorio. En este caso la medida de distancia es conocida como *distancia Mahalanobis*

2.3.1.2 Diseño de la biblioteca

El primer paso en el diseño del cuantificador vectorial es calcular los vectores y_k asociados a cada celda o región C_k para formar la biblioteca. Un método para diseñar la biblioteca es el famoso *algoritmo de las K-medias*, donde K hace referencia al número de centroides. En el cuadro 2.1, se puede seguir el proceso que sigue dicho algoritmo.

En el cuadro 2.1, m es el índice de iteración y $C_k(m)$ es el centroide calculado en la iteración m. Cada vector y_k es obtenido a partir de los patrones de entrenamiento asociados al centroide C_k. Al vector y_k se le conoce con el nombre de *centroide*

$$y_k = cent(C_k) \tag{2.4}$$

El cálculo del centroide de cada región depende de la medida de distorsión utilizada.

Paso 1	*Inicialización. Se eligen los vectores* y_k
Paso 2	*Clasificación.* *Se clasifica el conjunto de entrenamiento* x *en las celdas* C_k *utilizando la regla del vecino más próximo* $x \in C_k(m)$ *si* $d[x, y_k(m)] \leq d[x, y_j(m)]$ $\forall j \neq k$
Paso 3	*Adaptación de los vectores de la biblioteca:* $m \leftarrow m + 1$ *Se adaptan los vectores de la biblioteca* *El centroide se calcula a partir* *de los vectores de entrenamiento asociados a cada celda*
Paso 4	*Si se decrementa la medida de distorsión total* $D(m) < D(m-1)$ *y está por debajo de un umbral se finaliza, sino se pasa al paso 2*

Tabla 2.1: Algoritmo de entrenamiento del VQ

El proceso de trabajo del cuantificador vectorial es el siguiente: el cuantificador asigna un vector de la biblioteca y_k, si x, patrón de entrada, está dentro de la región C_k. Para conocer en qué celda se debe colocar el vector x en el espacio de dimensión d, se ha de medir la distancia del vector x respecto a todas las celdas, y elegir aquella que presenta la distancia más pequeña.

Un gran número de algoritmos han sido propuestos en reconocimiento de patrones, los cuales han sido diseñados para disminuir el cálculo computacional basados en nociones del espacio Euclídeo. Por ejemplo, en el cálculo de una biblioteca de L celdas, lo primero que se hace es aplicar el algoritmo de las K-medias con $K = 2$, el siguiente paso es dividir ambas regiones en 2 subregiones, y así sucesivamente hasta alcanzar las L celdas. Se deriva que es condición necesaria que L sea potencia de 2.

El cuantificador vectorial es una función que permite reducir los datos de entrada a un conjunto de datos almacenados en una biblioteca, permitiéndose así trabajar con espacios de datos reducidos y de tamaño igual a la longitud de la biblioteca. Cuando un vector de entrada x se cuantifica, obtendremos el vector y_k, donde es tan valiosa la información del vector y_k como la del índice k, que muestra la posición del vector y_k dentro de la biblioteca. En muchas aplicaciones interesa trabajar directamente con los índices y no con los vectores cuantificados.

2.3.1.3 Conclusiones del VQ

Analizado el funcionamiento del cuantificador vectorial como sistema de clasificación se deduce que se trata meramente de un clasificador de patrones estáticos. Por supuesto la señal de voz, señal con la que se está trabajando tiene un carácter temporal por lo que resulta ser un modelo poco eficiente. El cuantificador vectorial sólo clasifica patrones de entrada, sin tener en cuenta la información temporal de los datos de entrada anteriores y posteriores. Es además un sistema carente de memoria. Es por lo que se ha desechado del estudio como clasificador de secuencias temporales trabajando de manera independiente. Para poder utilizar este clasificador de secuencias temporales se ha de combinar con técnicas de alineamiento temporal.

Un gran problema encontrado en el reconocimiento de voz, es que una palabra pronunciada varias veces no tiene porque tener la misma duración en el tiempo, con lo que surge un problema. Es por ello que una parte importante dentro de la historia del reconocimiento fue el estudio de técnicas de alineamiento temporal entre patrones. Esta técnica es conocida como DTW, *Dynamic Time Warping*, la cual se explica brevemente su concepto en el siguiente apartado.

Sin embargo el cuantificador vectorial trabajando independientemente, es muy útil porque permite hacer una reducción del espacio de patrones de entrada a otro espacio con un conjunto finito de vectores. Y dentro del sistema de reconocimiento discreto de voz se utiliza el cuantificador vectorial para hacer una primera clasificación estática de los datos de entradas, y reducir así los datos a procesar a un conjunto de valores conocidos. La información importante y que en definitiva se desea clasificar está en la secuencia de los vectores o índices. Por lo que necesitamos de un segundo clasificador que pueda procesar esa información temporal de los datos.

2.4 Técnicas de alineamiento temporal de patrones, (DTW)

Las técnicas de alineamiento temporal de patrones no son un método de clasificación sino una mejora introducida a la tarea de clasificación de secuencias temporales. Cuando un locutor habla y pronuncia varias veces la misma palabra se produce el efecto de que la duración de las palabras varían en el tiempo, lo que produce fluctuaciones no lineales en el tiempo. La eliminación de estas fluctuaciones o normalización ha sido uno de los problemas centrales en las investigaciones llevadas a cabo sobre el reconocimiento de voz. Los primeros métodos requerían de algún sistema que ampliara o encogiera el eje del tiempo para así homogeneizar los patrones de entrada, aplicándose algunas técnicas de transformación lineal del eje del tiempo. Los resultados de este tipo de transformaciones

demostraron ser insuficientes para eliminar estas fluctuaciones. Las técnicas más efectivas son aquellas que aplican transformaciones no lineales en el eje del tiempo.

Las técnicas de programación dinámica DTW, fue el primer método donde se intentó solventar el problema de que dos patrones con duración diferente en el tiempo, se correspondan con una misma clase. Por lo tanto la comparación directa de ambos patrones no es correcta, es necesaria una alineación de los patrones en el tiempo.

Este sistema se basa en comparar el espectro de la señal de voz a intervalos pequeños de tiempo en torno a los 10 ó 15 mseg. Estos intervalos de tiempo de voz son representados por un conjunto de medidas de características acústicas espectrales.

El principio de las técnicas DTW, trata de solventar las diferencias temporales entre dos patrones ensachando/encogiendo el eje del tiempo de uno de los patrones haciendo que los máximos coincidan. A continuación la distancia temporal normalizada es calculada como la mínima entre ellos. La idea básica de estas técnicas han sido publicadas en [Sakoe78], aplicadas al reconocimiento de dígitos en japonés.

Sean dos vectores $A = a_1, a_2, ..., a_i, ..., a_I$ y $B = b_1, b_2, ..., b_j, ..., b_J$ cuyas componentes son los parámetros espectrales de un intervalo de tiempo, donde $I \neq J$.

Con el fin de clarificar las diferencias temporales entre ambos patrones se considera el plano i - j, como se muestra en la figura 2.4, donde los patrones A y B son representados a lo largo del *eje i* y del *eje j* respectivamente. Ambos patrones son de la misma categoria, cuyas diferencias temporales pueden ser recogidas como una secuencia de puntos, $c = (i, j)$

$$F = c(1), c(2), ..., c(k) \tag{2.5}$$

donde

$$c(k) = (i(k), j(k)) \tag{2.6}$$

Esta secuencia de puntos, puede representar un función, la cual puede ser considerada como una forma aproximada de reflejar el eje del tiempo del patrón A sobre el eje del patrón B. A esta función se le denomina *warping*. Cuando no hay diferencias temporales entre estos patrones, la función **F** *warping* coincide con la diagonal formada por la línea $i = j$, ver figura 2.4. Las deviaciones respecto a la diagonal son el reflejo de las variaciones temporales.

Una posible medida de la distancia entre dos vectores de características a_i y b_j puede ser la expresada en la ecuación 2.7

$$d(c) = c(i, j) = \|a_i - b_j\| \tag{2.7}$$

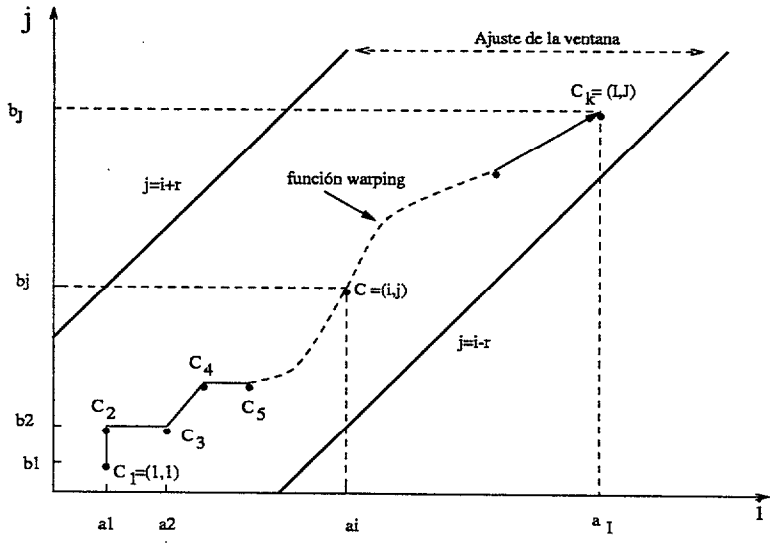

Figura 2.4: Función *warping* y ajuste de la ventana.

La suma ponderada de las distancias sobre la función **F** definida en 2.5 es, ver ecuación 2.8

$$E(F,k) = \sum_{k=1}^{K} d(c(k)) \cdot w(k) \tag{2.8}$$

Donde $w(k)$ es una ponderación no negativa, introducida intencionadamente con el fin de obtener una medida de **E** que sea flexible y además una medida razonable de la bondad de la función *warping* **F**. Esta distancia residual puede ser considerada como la distancia entre los dos patrones. Cuando es mínima **E(F)**, indica que se ha realizado un ajuste temporal mínimo. Una vez que se han eliminado las diferencias temporales entre los dos patrones se espera que el sistema se comporte de manera estable a las fluctuaciones o variaciones temporales. Bajo estas consideraciones la distancia temporal normalizada entre estos dos patrones puede ser definida por la ecuación 2.9

$$D(A,B) = \min_{F} \left[\frac{\sum_{k=1}^{K} d(c(k)) \cdot w(k)}{\sum_{k=1}^{K} w(k)} \right] \tag{2.9}$$

donde el denominador $\sum w(k)$ se utiliza para compensar el número de puntos **K** de la función **F**. La función 2.9 no es más que la definición fundamental de distancia temporal normalizada. La efectividad de esta medida depende de la función de *warping* F y de la definición de la función de ponderación $w(k)$.

El objetivo fundamental es encontrar el mejor alineamiento entre pares de patrones, lo que equivale a encontrar el mejor camino a través del *grid* de la figura 2.4, comparando las características acústicas de un patrón con las del otro. Encontrar el mejor camino

supone resolver un problema de minimización que evalue la distancia entre dos patrones. Este problema se resuelve mediante la aplicación de técnicas de programación dinámica.

2.4.1 Principios de la programación dinámica

El principio del DTW se basa en buscar el camino óptimo o mínima distancia. Consideremos un conjunto de puntos denominados del 1 a N. Asociados con cada par de puntos $c = (i, j)$ se define una función de coste $\zeta(i, j)$ la cual representa el coste de moverse del punto i al punto j. El problema es encontrar el camino de mínimo coste así como la secuencia de puntos y en cuántos movimientos. Puede ocurrir que el camino óptimo no sea el camino directo sino a través de otros puntos.

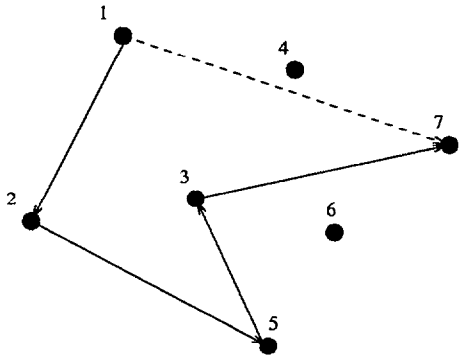

Figura 2.5: Red de 7 puntos. Hay que encontrar el camino óptimo entre el nodo 1 y el 7.

Para el caso de la figura 2.5, la función de coste será: $\varphi(7) = \zeta(1,2)+\zeta(2,5)+\zeta(5,3)+\zeta(3,7)$.

Generalizando esta ecuación nos queda

$$\varphi(i,j) = \min_{l}[\varphi(i,l) + \varphi(l,j)] \tag{2.10}$$

Donde $\varphi(i, j)$ es el coste mínimo asociado al camino entre i y j, existiendo otros l nodos entre los nodos de inicio y fin.

Otro problema que se puede resolver con esta técnica sería encontrar el camino óptimo en M pasos, es decir, fijar el número de iteraciones. Viendo la figura 2.5, se representa una matriz de $N \times M$. Desde cada nodo es posible realizar N movimientos hacia la derecha. Es posible que el camino finalice en l, en la iteración m, siendo $m < M$. En el movimiento $m+1$ la búsqueda del camino puede progresar hasta el nodo n. La función de coste puede ser definida de la misma manera que la dada en la ecuación 2.11.

$$\varphi_{m+1}(i,n) = \min_{l}[\varphi(i,l) + \zeta(l,n)] \tag{2.11}$$

La ecuación 2.11 describe una recursión que permite buscar el camino óptimo de una manera progresiva.

Otro de los aspectos a estudiar es la función *warping*. Hay que definir la función **F**, de la ecuación 2.5, a la cual se le ha de imponer una serie de condiciones con el fin de obtener un alineamiento temporal lo más óptimo posible. Algunas de estas consideraciones son:

1. Que sea monótona:
$$i(k-1) \leq i(k) \quad j(k-1) \leq j(k) \tag{2.12}$$
2. Que sea continua:
$$i(k) - (k-1) \leq 1 \quad j(k) - j(k-1) \leq 1 \tag{2.13}$$
3. Que esté acotada:
$$i(1) = 1 \quad j(1) = 1 i(K) = I \quad j(K) = J \tag{2.14}$$
4. Se ha de definir el tamaño de la ventana, ver figura 2.4:
$$|i(k) - j(k)| \leq r \tag{2.15}$$
Donde r es un valor entero y positivo, denominado tamaño de la ventana.
5. Pendiente. No se permiten ciertos saltos en la función F, porque tales desviaciones pueden ocasionar el efecto contrario al que se pretende. Estas consideraciones toman lugar a la hora de evaluar los posibles saltos entre puntos consecutivos, es decir si el punto c(k) se mueve en la dirección del eje i o j

De todo esto, se trata con más detalle en [Sakoe78] y en [Rabi94].

2.5 Árboles de decisión

Otro de los métodos de clasificación son los sistemas basados en reglas de decisión. A este tipo de sistemas se le conoce con el nombre de clasificadores estructurados de árboles o clasificadores estructurados de árbores binarios. Estos sistemas se crean por divisiones sucesivas del conjunto de entrenamiento T, de manera descendente. Se comienza con el conjunto total, T, y éste se divide en dos subconjuntos disjuntos t_2 y t_3. Este proceso se

repite sucesivamente para los subconjuntos, con lo que el árbol va creciendo de manera descendente. En la figura 2.6 aparece un ejemplo de árbol definido para clasificar 6 clases: C_1, C_2, C_3, ..., y C_6. Cada subconjunto se divide en dos hasta que se alcanze una condición de parada. En la misma figura se puede observar dos tipos de nodos: no-terminales y terminales. Los nodos no-terminales, representados con un círculo, son los que se dividen en dos subconjuntos. Los nodos terminales, representados con nodos rectángulares, son los que no se pueden dividir y representan a una clase. Este tipo de estructuras permite que una clase venga definida por varios nodos terminales. Es decir, observando la gráfica 2.6, la clase C_1 viene determinada por el subconjunto t_{15}, la clase C_2 viene definida por la unión del subconjunto $t_{11} \cup t_{14}$, la clase C_3 por la unión de $t_{10} \cup t_{16}$ y así sucesivamente.

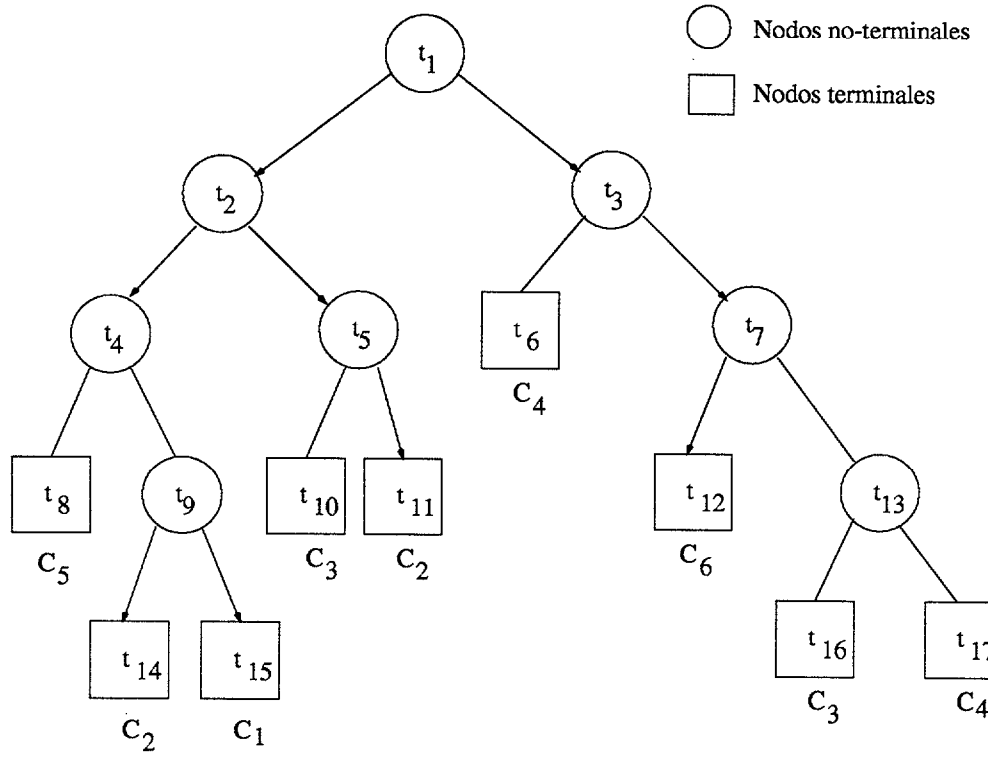

Figura 2.6: Red de 7 puntos. Hay que encontrar el camino óptimo entre el nodo 1 y el 7.

Las divisiones que un nodo puede sufrir vienen definidas por una serie de reglas definidas antes del entrenamiento. Estas medidas se realizan sobre el valor de los vectores de entrada. El patrón de entrada se va moviendo por el árbol de manera descendente hasta llegar a un nodo terminal. Al llegar a éste, se predice a la que clase pertenece.

La construcción de un árbol depende enormemente de tres factores:

1. La definición de las reglas que determinan las divisiones de los nodos.
2. La definición del criterio de parada, cuando un nodo es terminal o no.
3. La asignación de un nodo terminal a una clase.

Lo complejo de estos sistemas está en cómo tratar el conjunto de entrenamiento sobre el que se va a construir el árbol de decisión, cómo se van a definir los nodos terminales y las clases.

La idea fundamental es cómo realizar la división de un nodo en otros dos nodos (nodo izquierdo y nodo derecho) de tal forma que de la división se obtenga o definan subconjuntos más puros que los datos llegados al nodo actual. En realidad se trata de todo un proceso de *selección de datos.* Los datos llegados a un nodo, con las reglas definidas en él, se decidirá si seguir por el nodo izquierdo o derecho.

Los cuatro elementos importantes a la hora de definir un árbol son:

1. Definir un conjunto de reglas de decisión.
2. Definir un criterio de división de un nodo.
3. Definir un criterio de parada.
4. Definir una regla que asigne un nodo terminal a una clase.

En [Breiman93] se analiza con detalle estos cuatro puntos. Las reglas de decisión se basan principalmente en estudiar la probabilidad de que una clase determinada alcance o llegue a un nodo dentro del árbol.

2.6 Aplicación de las redes neuronales

2.6.1 Introducción

Una red neuronal *Neural Network, NN* es un sistema que permite establecer una relación no lineal entre la salida y la entrada, cuyas características es una simulación abstracta del sistema nervioso. Haciendo un pequeño paréntesis sobre el funcionamiento del sistema nervioso, una neurona envía sus salidas a otras neuronas vía el axón. El axón transporta la información a través de una serie de potenciales, en realidad se trata de todo un proceso químico de iones donde la información viene dada o se transporta como una diferencia de potencial. Las neuronas excitadas ante un estímulo, responden de manera paralela con una respuesta determinada. Se estima que el cerebro humano tenga entorno a 100 billones de neuronas y 10000 conexiones entre las distintas neuronas. Muchas de estas conexiones se establecen durante la fase de aprendizaje del individuo.

Pues bien, las redes neuronales son modelos cuyo funcionamiento recuerda al sistema nervioso del hombre en una serie de términos, aunque aún están muy lejos de lograr la perfección de éste. Estos términos podrían ser:

1. Aprendizaje. Las redes neuronales requieren de una fase de aprendizaje para poder dar una respuesta correcta ante una entrada. Durante esta fase, las conexiones entre nodos de distintas capas son ponderados por un determinado valor. Este valor contiene la memoria o contribución de un nodo con otro nodo de la red neuronal. Resumiendo, las redes neuronales están dotadas de capacidad de *aprendizaje*. Son sistemas que a base de enfrentarse con un problema dado, se adaptan en virtud de los resultados.

2. Estructura. Las redes neuronales tienen una estructura formada por elementos de proceso denominados neuronas o nodos, distribuidos por capas, donde la salida de un nodo de una capa contribuye en otros nodos de capas superiores. El estímulo de una entrada recorre toda la red hasta dar una respuesta.

Por supuesto, aunque se establezcan términos de similitud con el sistema nervioso, las redes neuronales están muy lejos de la perfección de éste y no se las puede sacar del entorno de que son sistemas no lineales y que presenta grandes ventajas en determinadas aplicaciones frente a los sistemas clásicos.

2.6.2 Descripción general de las redes neuronales

Las redes neuronales están formadas por un conjunto de elementos de proceso denominados nodos o neuronas, ver figura 2.7. Estos nodos dan una salida ante un patrón de entrada de manera paralela. En el diseño de redes neuronales hay una fase importante y es el aprendizaje, fase donde la red calcula como ha de contribuir cada elemento o nodo sobre cualquier otro nodo de la red. A esta contribución se le denomina pesos y viene reflejada en la conexión entre los nodos.

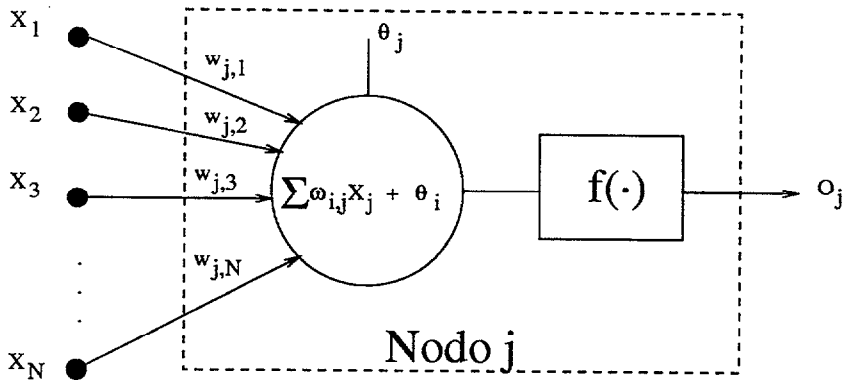

Figura 2.7: Nodo de una Red Neuronal.

En la figura 2.7, $\{x_1, x_2, ...x_n\}$ puede ser el patrón/vector de entrada o bien ser la salida de otras neuronas dependiendo si se trata de un nodo de la primera capa o de capas superiores respectivamente. Los $w_{l,j,i}$ representan los pesos (contribución) de las

conexiones de los nodos i hacia el nodo j. El valor θ_j es un valor umbral u *offset*. La función $f(\cdot)$ se trata de una función limitadora de la salida de la red. La señal o_j es la salida final de la neurona o nodo.

De aquí se deduce otra relación entre un nodo o elemento de proceso con una neurona biológica. El vector x serían las señales excitadoras y la salida o_j, la respuesta del elemento de proceso. Por supuesto la salida de este nodo puede contribuir a otros nodos o bien ser un nodo de salida.

En la figura 2.8 muestra una arquitectura típica de red neuronal, donde las unidades de proceso se distribuyen por capas. La capa etiquetada por L_0 es la capa de entrada y L_L es la capa de salida. Entre la capa L_0 y L_2 hay 1 capa denominada capa oculta. Se dice que la red tiene 2 capas porque la capa de entrada no produce ninguna transformación.

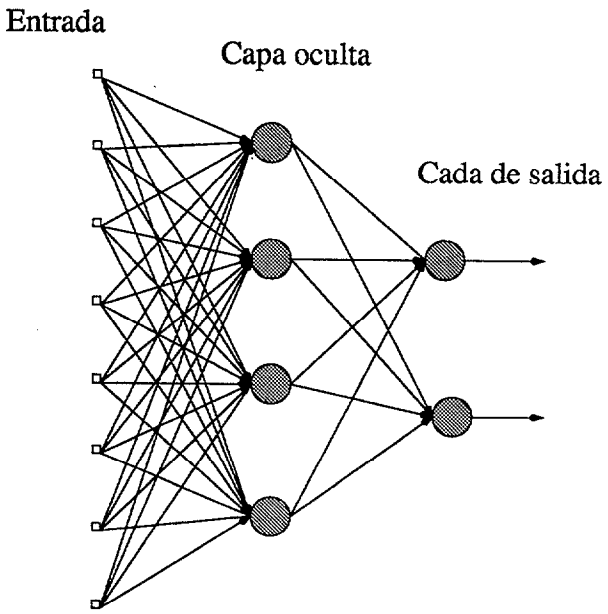

Figura 2.8: Estructura típica de una Red neuronal.

Los nodos de una capa se conectan unidireccionalmente con los nodos de la capa siguiente, en general todas con todas, aunque pueden encontrarse redes neuronales donde sus nodos no reciben la contribución de todos los nodos de la capa anterior. La salida de cada nodo se ve modificada por la multiplicación del peso, que se representa por w_{ji}^l, siendo l la capa de llegada y j e i los índices de posición de las neuronas de llegada y de salida respectivamente. El aprendizaje de la red se obtiene, mediante la modificación de los parámetros de la red: pesos y los propios de las unidades de proceso.

La mayoría de las redes son entrenadas usando un aprendizaje supervisado, esto significa que a la red se le proporciona el conjunto de pares de entrada-salida. Por el contrario, están las redes con entrenamiento no supervisado en las que sólo se le presenta a la red los patrones de entrada.

Las redes neuronales tienen un gran campo de aplicación: establecer correspondencias de carácter general, establecer asociaciones, de toma de decisión, de diagnóstico, reconocimiento y además llevar a cabo procesos de optimización.

Los modelos neuronales están divididos en dos categorías: redes estáticas y redes dinámicas. Las primeras se caracterizan porque el elemento de proceso realiza una función con ausencia de memoria. La salida es función sólo de la entrada actual no de las muestras pasadas o futuras. Dentro de este tipo de redes se encuentra las conocidas redes *MultiLayer Perceptron* en lo sucesivo Perceptrón Multicapa o MLP, las redes *Radial Basis Function*, en lo sucesivo Funciones de Base Radial o RBF. Las segundas, las redes dinámicas son sistemas con memoria. La función de transferencia viene determinada por una ecuación diferencial. Dentro de este tipo de redes encontramos: redes con realimentación hacia adelante, redes sin realimentación y redes con realimentación hacia atrás.

2.6.3 Redes estáticas

Las redes estáticas implementan transformaciones del tipo $u = G(x)$ donde $x \in \Re^n$ y $u \in [0,1]^m$, donde n y m son enteros los cuales representan la dimensión de x y u respectivamente. Como ejemplos de redes estáticas muy conocidas entre las que citamos: la red MLP, la red RBF y las redes polinómicas. A continuación se hace una breve descripción de ellas.

2.6.3.1 Perceptron multicapa, MLP

Esta red tiene su origen en la perceptron, concebida por Rosenblatt en 1958, ver [Hush93]. Sin embargo las capacidades de una simple perceptron están limitadas a una división lineal del espacio de decisión. La unión de varias perceptrones y su distribución en distintas capas pueden dar origen a regiones de decisión más compleja. Otra diferencia respecto a la perceptron de Rosenblatt, es que en este caso utiliza una función continua *sigmoid* en vez de una función escalón no derivable. En la MLP, el vector de entrada excitará a todas los nodos de la capa de entrada. La salida de estos nodos excitará a su vez las siguientes neuronas de la capa siguiente y así sucesivamente, ver figura 2.9.

La estructura clásica de una MLP es que todos los nodos del nivel l estén conectados con todos los nodos del nivel $l+1$. En la figura 2.9 representa una MLP de 3 niveles. Se distingue dentro de la red lo que es la capa de entrada y de salida, las capas intermedias son denominadas como capas ocultas.

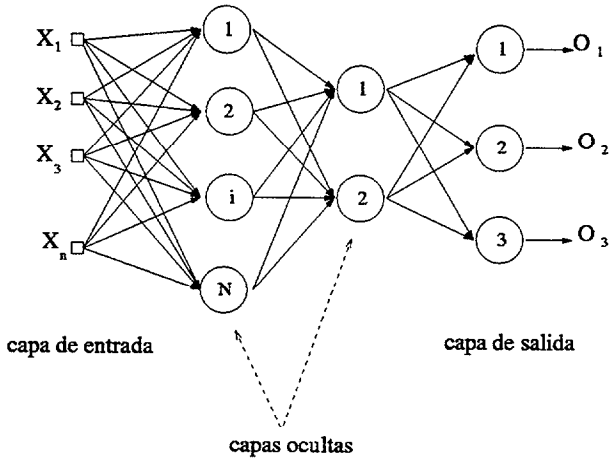

Figura 2.9: Estructura típica de una MLP.

Las MLPs tienen tres áreas de aplicación:

1. Implementación de funciones lógicas.
2. Problemas de clasificación, de división del espacio.
3. Implementación de transformaciones no lineales.

Dependiendo del problema se necesitarán de 2 a 3 capas con una distribución de nodos por capa dependiendo de la aplicación. No hay ninguna teoría acerca del número de nodos por capa, lo que si existen son ciertas reglas orientativas de cuántos nodos se pueden comenzar el entrenamiento de la red. Lo que si hay son técnicas de crecimiento o de poda que buscan la estructura óptima de la red.

Otro de los apartados fundamentales de las MLPs es el algoritmo de entrenamiento. Una de las limitaciones que presenta el perceptron de Rosenblatt fue justamente el algoritmo de entrenamiento. El procedimiento más utilizado para el entrenamiento de las MLPs es la aplicación del algoritmo de gradiente descendente. A continuación se comenta brevemente en qué consiste.

Para poder seguir el desarrollo del algoritmo, se muestra en la tabla 2.2 la notación de las variables que se van a utilizar.

La salida de un nodo viene determinado por

$$u_{l,j} = f(\sum_{i=0}^{N_{l-1}} w_{l,j,i} \cdot u_{l-1,i}) \tag{2.16}$$

Donde $f(\cdot)$ es la función de sigmoid, ver ecuación 2.17. Esta función se caracteriza por ser continua, no lineal y varía entre 0 y 1 monotonicamente desde $-\infty$ y ∞. Otra función

Notación	Significado
$u_{l,j}$	salida del nodo j de la capa l
$w_{l,j,i}$	pesos de la capa l que conecta el nodo i de la capa $l-1$, con el nodo j
x_p	muestra p de entrenamiento
$u_{0,i}$	componente i del vector de entrada
$d_j(x_p)$	respuesta deseada del nodo j
N_l	número de nodos de la capa l
L	número de capas
P	número de patrones de entrada

Tabla 2.2: Variables del algoritmo de aprendizaje.

con iguales características es la **tangente hiperbólica**, con la diferencia que varía entre -1 y 1. El término g de la ecuación 2.17 determina la pendiente de la zona de transición entre 0 y 1. Una de las ventajas de esta función es que es diferenciable.

$$f(x) = \frac{1}{1 + e^{-g \cdot x}} \tag{2.17}$$

La derivada de la función de sigmoid definida en la función 2.17 es:

$$f'(x) = \frac{df(x)}{dx} = f(x) \cdot (1 - f(x)) \tag{2.18}$$

Este algoritmo trata de encontrar los pesos de la red, los cuales minimizen una función de coste. Esta función de coste puede ser la suma del error cuadrático medio cometido por todos los nodos para todos los patrones de entrenamiento.

$$J(w) = \sum_{p=1}^{P} J_p(w) \tag{2.19}$$

Donde P es el número de patrones de entrenamiento, y $J_p(w)$ es el total del error cuadrático para los P patrones de entrenamiento, ver ecuación 2.20

$$J_p(w) = \frac{1}{2} \sum_{q=1}^{N_L} (u_{L,q}(x_p) - d_q(x_p))^2 \tag{2.20}$$

N_L, de la ecuación 2.20 es el número de nodos de la capa de salida. Los pesos de la red se modifican iterativamente de acuerdo a la ecuación 2.21:

$$w_{l,j,i}(k+1) = w_{l,j,i}(k) - \mu \frac{\partial J(w)}{\partial w_{l,j,i}} = w_{l,j,i}(k) - \mu \sum_{p=1}^{P} \frac{\partial J_p(w)}{\partial w_{l,j,i}} \tag{2.21}$$

Siendo μ una constante denominada constante de aprendizaje. Este algoritmo se desarrolla aplicando derivadas parciales a la función $J_p(w)$ respecto de $w_{l,j,i}$. Para ello es preciso aplicar la regla de la cadena, ver 2.22

$$\frac{\partial J_p(w)}{\partial w_{l,j,i}} = \frac{\partial J_p(w)}{\partial u_{l,j}} \cdot \frac{\partial u_{l,j}}{\partial w_{l,j,i}} \tag{2.22}$$

donde

$$\frac{\partial u_{l,j}}{\partial w_{l,j,i}} = \frac{\partial}{\partial w_{l,j,i}} \left[f(\sum_{m=0}^{N_{l-1}} w_{l,j,i} \cdot u_{l-1,m}) \right] \tag{2.23}$$

Resultando

$$\frac{\partial u_{l,j}}{\partial w_{l,j,i}} = u_{l,j} \cdot (1 - u_{l,j}) \cdot u_{l-1,i} \tag{2.24}$$

El término de la ecuación 2.22, $\frac{\partial J_p(w)}{\partial u_{l,j}}$, representa la sensibilidad de la función de coste $J_p(w)$ respecto a la salida del nodo $u_{l,j}$, y ésta influye en las salidas de los nodos de las capas inferiores a l de la red neuronal y así sucesivamente. Por lo tanto, este término depende de la capa donde se esté evaluando dicha derivada.

Para las capas ocultas

$$\frac{\partial J_p(w)}{\partial u_{l,j}} = \sum_{m=1}^{N_{l+1}} \frac{\partial J_p(w)}{\partial u_{l+1,j}} \cdot \frac{\partial u_{l+1,m}}{\partial u_{l,j}} \tag{2.25}$$

Llegando a la expresión 2.26

$$\frac{\partial J_p(w)}{\partial u_{l,j}} = \sum_{m=1}^{N_{l+1}} \frac{\partial J_p(w)}{\partial u_{l+1,m}} \cdot u_{l+1,m} \cdot (1 - u_{l+1,m}) \cdot w_{l+1,m,j} \tag{2.26}$$

Para la capa de salida obtenemos:

$$\frac{\partial J_p(w)}{\partial u_{L,j}} = u_{L,j}(x_p) - d_j(x_p) \tag{2.27}$$

El cálculo de 2.22 es un proceso *backwards*, (hacía atrás). Analizando la expresión 2.26 es necesario calcular el error cometido por los nodos de la última capa para poder evaluar 2.26, para la capas interiores, es por lo que se denomina a este proceso, algoritmo de retropropagación, o *algoritmo de backpropagation*. Es decir, primero se calcula el error de la capa de salida dada por la expresión 2.27 y seguidamente se aplica la ecuación 2.26 para las capas oculatas. Este algoritmo se repite hasta que un mínimo es alcanzado. En la práctica, no es tan fácil conocer a priori la superficie del error, por lo tanto para finalizar

se puede establecer distintos criterios de parada: cuando J cae por debajo de un cierto umbral o bien por el número de iteraciones. Otra posibilidad es parar el entrenamiento cuando la red desarrolle correctamente su función, aunque ésto puede implicar perder la generalización.

En la tabla 2.3 se expone el algoritmo de entrenamiento *Backpropagation*

2.6.3.2 Redes de función de base radial

Una red de función de base radial, (Radial Basis Function, RBF), es una red de dos capas, ver figura 2.10. La primera capa divide el espacio en tantos centroides como nodos tenga esta capa. Los nodos de salida capa de salida, forman una combinación lineal de funciones de base calculados por la primera capa. La función más común de base es una función Gaussiana de la forma, dada en la ecuación 2.28

$$u_{1,j} = \exp\left[-\frac{(x - w_{1,j}^T)(x - w_{1,j})}{2\sigma_j^2}\right] \qquad j = 1, 2, ..., N_1 \tag{2.28}$$

donde $u_{i,j}$ es la salida del nodo j de la primera capa, x es el patrón de entrada, $w_{1,j}$ son los pesos de la primera capa, centro de la función Gaussiana por nodo j. El parámetro de normalización viene determinado por σ_j^2 para el nodo j. Y N_1 es el número de nodos de la primera capa.

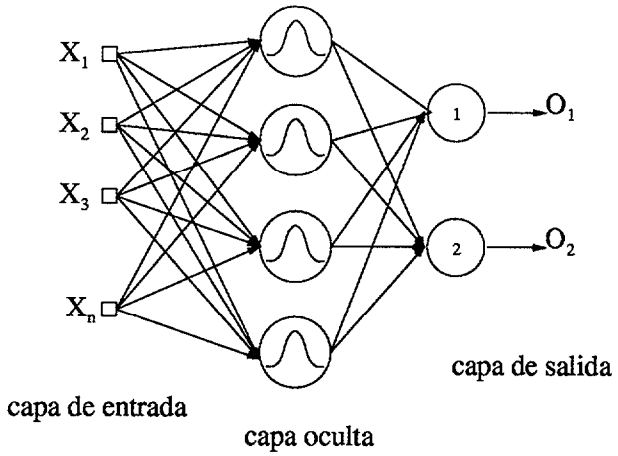

Figura 2.10: Estructura típica de una RBF.

El rango de salida de cada nodo está entre 0 y 1, así que cuando una entrada está próxima al centro de la Gaussiana, dará una salida alta a la salida. El nombre de Radial Basis Function viene del hecho de que las funciones Gaussianas son radialmente simétricas,

Procedimiento BACK-PROPAGATION
Inicializar los pesos
repetir
se elige la pareja de entrada-salida $(x, d)_p$, $u_0 = x$
FEED-FORWARD;
COMPUTE-GRADIENT;
UPDATE-WEIGHTS;
hasta que las condiciones de finalizar se alcancen

subrutina FEED-FORWARD
para layer=1 hasta L hacer
para nodo=1 hasta N_{layer} hacer
$u_{layer,nodo} = f(\sum_{i=0}^{N_{layer-1}} w_{layer,nodo,i} \times u_{layer-1,i})$
fin para
fin para
fin subrutina FEED-FORWARD

subrutuna COMPUTE-GRADIENT
para layer=L hasta 1 hacer
para nodo=1 hasta N_{layer} hacer
si layer=L entonces $e_{L,nodo} = u_{L,nodo} - d_{nodo}$
sino
$e_{layer,nodo} = \sum_{m=1}^{N_{layer-1}} e_{layer+1,m} \times u_{layer+1,m} \times (1 - u_{layer+1,m}) \times w_{layer+1,m,nodo}$
fin para
para todos los pesos de cada capa hacer
$g_{layer,j,i} = e_{layer,j} \times u_{layer,j} \times (1 - u_{layer,j}) \times u_{layer-1,i}$
fin para
fin subrutina COMPUTE-GRADIENT

subrutina UPDATE-WEIGHTS
para todos los $w_{layer,j,i}$ hacer
$w_{layer,j,i}(k+1) = w_{layer,j,i}(k) - \mu \times g_{layer,j,i}$
fin subrutina UPDATE-WEIGHTS

Tabla 2.3: Algoritmo de Backpropagation.

es decir cada nodo produce una salida idéntica para entradas que presenta la misma distancia al centro del kernel $w_{1,j}$.

La capa de salida viene determinada por la ecuación 2.29, donde y_j es la salida del nodo j, $w_{2,j}$ es el peso del mismo nodo, y u_1 es el vector de salida de la primera capa. N_2 es el número de nodos de la capa 2.

$$y_j = w_{2,j}^T \times u_1 \qquad j = 1, 2, ..., N_2 \tag{2.29}$$

Las RBFs se pueden utilizar para aplicaciones de clasificación y de aproximación de funciones.

Hay una gran variedad de procedimientos para tratar el aprendizaje de la RBF. Aquí sólo se comentará el procedimiento de entrenamiento más utilizado y general. Para comenzar el aprendizaje se divide en dos fases: entrenamiento de la capa oculta y entrenamiento de la capa de salida. El aprendizaje de la capa oculta se desarrolla mediante un método no supervisado, como puede ser cualquier algoritmo de clustering, como el de las *Kmedias*. El aprendizaje de la capa de salida es supervisado. En la tabla 2.4 se expone el algoritmo para la capa de salida.

2.6.3.3 Redes polinómicas

En [Hecht91] se hace una introducción de la red GMDH, *Group Method of Data Handling*. Esta red es polinomial; aproxima funciones de tipo $f : A \subset R^n \longrightarrow R$, que mapean un subconjunto del espacio Euclídeo de dimensión n a un número real. Su estructura es similar a la de cualquier red neuronal, estando compuesta de nodos, distribuidos por capas. La diferencia con el resto de las redes está:

1. Cada nodo tiene dos entradas Z_j^{k-1} y Z_i^{k-1}, donde el subíndice hace referencia al nodo y el superíndice a la capa, y una única salida Z_l^k. Ver figura 2.11.

2. La función no lineal que realiza el nodo es polinomial y viene dada en la ecuación 2.30.

$$Z_l^k = a_l^k \times \left(Z_j^{k-1}\right)^2 + b_l^k \times \left(Z_i^{k-1}\right) \times \left(Z_j^{k-1}\right) + c_l^k \times \left(Z_i^{k-1}\right)^2 + d_l^k \times Z_j^{k-1} + e_l^k \times Z_i^{k-1} + f_l^k \tag{2.30}$$

Siendo a_l^k, b_l^k, c_l^k, d_l^k, e_l^k, y f_l^k, los coeficientes del polinomio del nodo l de la capa k. Esta ecuación, puede expresarse de forma matricial, como el producto de la matriz de combinación de entradas, a la que denominaremos X_l y la matriz A_l^k de coeficientes.

Procedimiento LMS
Inicializar los pesos $w_{2,j}$
repetir
se elige la pareja de entrada-salida $(u_1, d)_p$

*/*Se calcula las salidas*/*
para todos los nodos
$y_{nodo} = w_{2,nodo}^T \times u_1$
fin para

/ Se calcula el error*/*
para todos los nodos hacer
$e_{nodo} = y_{nodo} - d_{nodo}$
fin para

/ Adaptación de pesos */*
para todos los nodos
$w_j(k+1) = w_j(k) - \mu e_{nodo} \times u_1$
fin para
hasta que las condiciones de finalización se alcancen

Tabla 2.4: Algoritmo LMS para la capa de salida de la RBF.

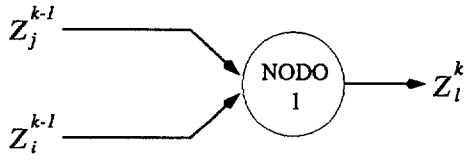

Figura 2.11: Nodo o elemento de proceso.

$$Z_l^k = \left[\left(Z_j^{k-1}\right)^2 \quad \left(Z_i^{k-1}\right) \times \left(Z_j^{k-1}\right) \quad \left(Z_i^{k-1}\right)^2 \quad Z_j^{k-1} \quad Z_i^{k-1} \quad 1\right] \times \begin{bmatrix} a_l^k \\ b_l^k \\ c_l^k \\ d_l^k \\ e_l^k \\ f_l^k \end{bmatrix} = X_l \cdot A_l^k \tag{2.31}$$

El objetivo es encontrar el conjunto de coeficientes que satisfacen la ecuación 2.31, para cada uno de los nodos. Éstos son calculados durante la fase de entrenamiento.

Método de entrenamiento

En [Hecht91] se expone el método de entrenamiento del GMDH, que a continuación se pasa comentar brevemente. La red GMDH tiene un entrenamiento supervisado, es decir hay que proporcionarle la salida deseada, ésta puede ser un valor real o bien una señal binaria 0 ó 1, que denominaremos Y. Cada capa se entrena de forma independiente y todos los nodos de una misma capa son entrenados para dar la misma salida, lo más cercana al valor de Y. Para el cálculo de la matriz A_l^k de coeficientes, lo único que hay que hacer es sustituir el valor, Z_l^k por la salida deseada Y en la ecuación 2.31. El conjunto de ecuaciones no tendrán una solución exacta sino aproximada con el fin de obtener un conjunto de coeficientes que minimicen el MSE (Mean Square Error). Se ha probado que cuando la matriz A_l^k se calcula a partir de la pseudoinversa de X_l, el MSE es mínimo.

Cada nodo construye un polinomio, ya que cada nodo realiza una función cuadrática de sus entradas. Por lo tanto cada capa de la red incrementa el grado de este polinomio por 2 y la salida final puede ser expresada como un polinomio de grado $2k$, siendo k, el número de capas. A este polinomio se le conoce como *polinomio de Ivakhrenko.*

El primer paso, es configurar el nivel k, para ello será necesario crear tantos elementos de proceso o nodos, como combinaciones de entradas tomadas de 2 en 2 sean posibles. Esto quiere decir que si el nivel $k-1$ tiene M nodos, el nivel k se creará con tantos nodos como combinaciones de todos los pares de entradas sean posibles, formando así las entradas de los elementos de proceso del nivel k. El número de nodos o elementos de proceso del nivel k viene expresado en la ecuación 2.32

$$C(M_{k-1}, 2) = M_{k-1} \cdot (M_{k-1} - 1) \div 2 \tag{2.32}$$

En la figura 2.12 aparece una red GMDH con 4 entradas, Z_1, Z_2, Z_3 y Z_4. La combinación de 4 entradas tomadas de 2 en 2 da 6, por lo que la primera capa se creará con 6 nodos.

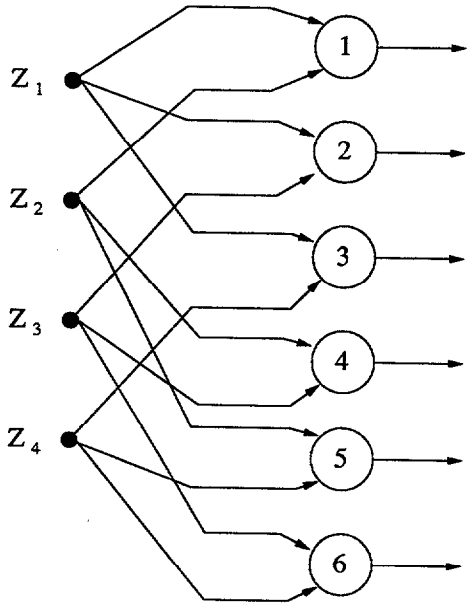

Figura 2.12: Formación de la primera capa de la Red GMDH.

Cuando un nivel es añadido a la red, se calculan las matrices de coeficientes A_l^k de cada nodo para un conjunto de entrenamiento (pares de entrada-salida) determinado. Todos los nodos son entrenados para dar la misma salida. Cada nivel se entrena con un conjunto diferente de entrenamiento, por lo tanto será necesario disponer de un gran conjunto de entrenamiento. Una vez que los coeficientes hayan sido calculados para el nivel k, el siguiente paso es evaluar el funcionamiento global de cada elemento nodo. Una forma de hacerlo es midiendo el error cuadrático medio, (**MSE**) que cada elemento de proceso produce sobre un conjunto de muestras nuevas. Para hacer ésto utilizaremos la fórmula :

$$\frac{1}{R}\left\|X_l \times A_l^k - Y\right\|^2 \tag{2.33}$$

Siendo R el número de muestras del conjunto de validación. El **MSE** de un nodo puede ser diferente al de otro, por lo tanto el siguiente paso será aplicar un proceso de poda a la red. Durante este proceso, se eliminarán aquellos nodos cuyo **MSE** sea grande. El proceso de construir la red capa a capa, continúa hasta que se cumple un criterio de parada. Este criterio se basa en ir evaluando el MSE de cada capa, cuando el MSE aumenta con el número de capas entonces el procedimiento de crecimiento se para. De

la última capa se selecciona aquel nodo que mejor salida presente y a continuación se procede a realizar un segundo proceso de poda, eliminando aquellos nodos que no están conectados con el nodo de salida reduciendo enormemente la estructura de la red.

En la figura 2.13 aparece la estructura final de la red GMDH de la figura 2.12. Durante el procedimiento de crecimiento de la red se ha supuesto que la información está relacionada con la entrada Z_2 por lo que al aplicar el método de poda se han eliminado todos aquellos nodos no relacionados con dicha entrada.

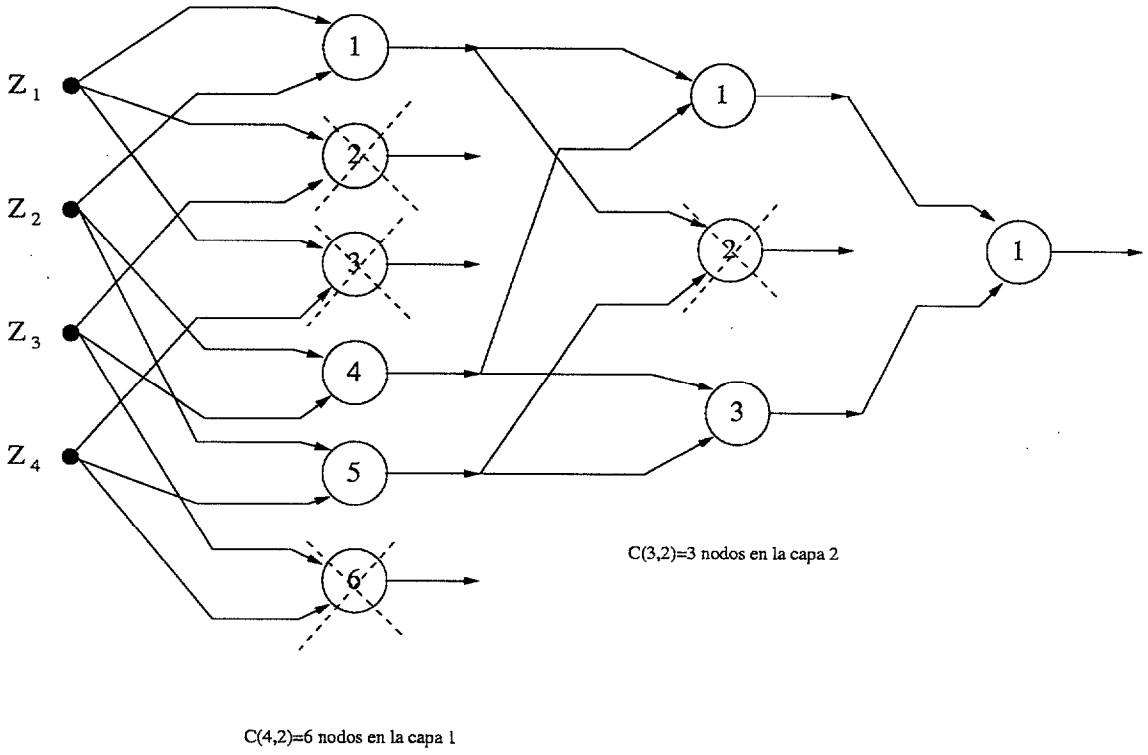

Figura 2.13: Red GMDH para 4 entradas.

Sea I_R, el conjunto de entrenamiento de entrada, de tamaño R e Y_R la salida deseada de la red para ese conjunto. Cada par de entrada-salida lo denominaremos i_r e y_r. Para crear el primer nivel de la red, se aplica lo expuesto anteriormente. Se combinan las entradas de 2 en 2. El siguiente paso, será calcular los coeficientes a_l^1, b_l^1, c_l^1, d_l^1, e_l^1, y f_l^1 para todo el conjunto de entrenamiento $i = 1, 2, 3, ..., r$. Este conjunto de coeficientes se almacena en la matriz A_l^1 de la ecuación 2.31. La rutina de cálculo de los coeficientes se muestra en tabla 2.5.

Una vez calculado la matriz W, (matriz con los coeficientes de cada nodo), procederemos a la poda de nodos, para ello aplicaremos el procedimiento de poda, que aparece en la tabla 2.6.

Una vez que se hayan eliminado, los nodos del nivel k, crearemos una capa nueva $k+1$, salvo que ahora, los nodos de entrada de esta capa son los nodos ganadores de la poda.

desde i=1:nodos(nivel)
  desde r=1:R
    Se calcula la matriz X_i^{nivel} *(ver ecuación 2.31)*
  fin desde
  Se calcula la matriz de coeficientes $A_i^k = Y_r \times X_i^{nivel}$
  Se guarda los coeficientes de cada nodo en la matriz W
fin desde

Tabla 2.5: Algoritmo de cálculo de los coeficientes de la red GMDH.

desde r=1:R
  desde i=1:nodos(nivel)
    Se calcula la matriz X_i^{nivel}
    Se calcula la salida de cada nodo $d = W \times X_i^{nivel}$
    Se calcula el MSE *de cada nodo*
  fin desde
fin desde
desde i=1:nodos(nivel)
  Si $MSE >$ *umbral*
    eliminar nodo
  fin si
fin desde

Tabla 2.6: Procedimiento de poda de la red GMDH.

2.6.3.4 Conclusiones sobre las redes estáticas

Las redes estáticas son redes que no tienen memoria por lo que parecen ser inapropiadas para abordar problemas espacio-temporales. Sin embargo, tanto las MLP como las RBF han demostrado ser buenas herramientas para aplicaciones de clasificación con una buena capacidad de discriminación. Sin embargo no presenta tener la misma cualidad para la clasificación de secuencias temporales.

Se han tratado en este apartado, porque serán utilizadas posteriormente en las distintas estructuras híbridas propuestas de esta tesis.

2.6.4 Redes dinámicas

El algoritmo *back-propagation* utilizado en las redes MLPs es uno de los métodos más conocidos para el entrenamiento de redes neuronales. Sin embargo presenta una gran limitación y es que sólo es aplicable a patrones estáticos. Se sabe que la variable tiempo puede ser un factor muy importante en muchas aplicaciones, voz, señales de control, visión, procesado de señales, etc. La cuestión es cómo integrar la variable tiempo dentro de las redes, cómo proporcionarles propiedades dinámicas, darles memoria. Los métodos utilizados para dotarlas de estas cualidades son con unidades de retardo o bien usando conexiones recurrentes.

Las redes que se obtendría serían de dos tipos: no recurrentes y recurrentes. Las primeras son las denominadas TDNN y las segundas RNN. Fue [Waibel89] quien propuso en el año 1989 la utilización de redes neuronales con retardo en el tiempo TDNN, para el reconocimiento de fonemas. Quizás ésta sea la estructura más sencilla utilizada en la clasificación de patrones dinámicos. Sin embargo, esta red tiene unos 544 pesos definidos. La figura 2.15 muestra la estructura de esta red.

Las TDNN es una MLP cuyos nodos de la capa oculta y los nodos de salida se duplican en el tiempo. En esta red, la unión entre dos nodos está representada por un filtro de respuesta finita (FIR). A esta red se la conoce como perceptron multicapa FIR. La complejidad del entrenamiento de este tipo de red está en función de las unidades de retardo. Se entrenan con el algoritmo *backpropagation*, el cual sufre algunas modificaciones por ejemplo considerando que la derivada de la función de coste se realiza a partir de la suma de los errores instantáneos para cada nodo, obtenidos en cada intervalo de tiempo. En [Day93] se generaliza el algoritmo de entrenamiento incorporando unidades de retardo adaptativas.

En aplicaciones reales esta red presenta grandes desventajas. La complejidad de su estructura requiere de un calculo computacional muy elevado además de invertir mucho tiempo en su entrenamiento. Además se ha demostrado [Kung93] que son redes muy sensitivas a degradaciones que pueda sufrir la señal.

Otro tipo de redes son las redes recurrentes RNN. Su estructura es similar a una red MLP pero con unidades de retardo conectadas en bucle entre las distintas capas. El algoritmo de entrenamiento es mucho más complicado que el famoso algoritmo *backpropagation* ya que se ha de entrenar en el espacio y en el tiempo.

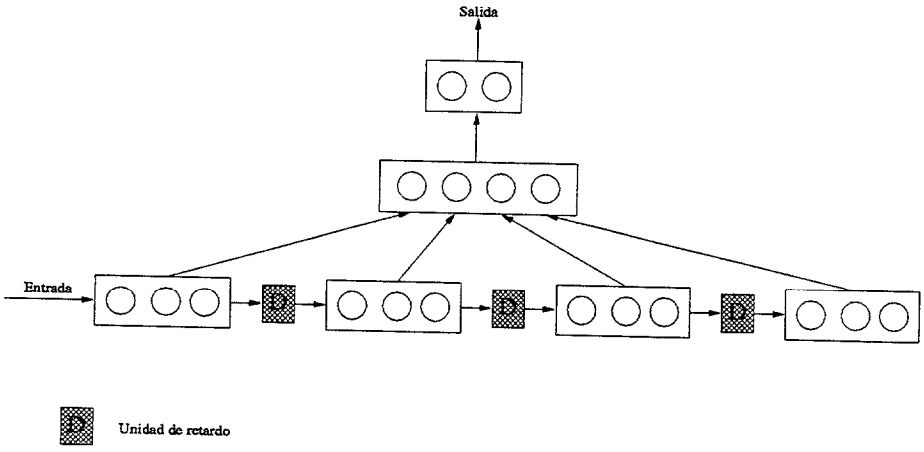

Figura 2.14: Ejemplo de una Red básica TDNN.

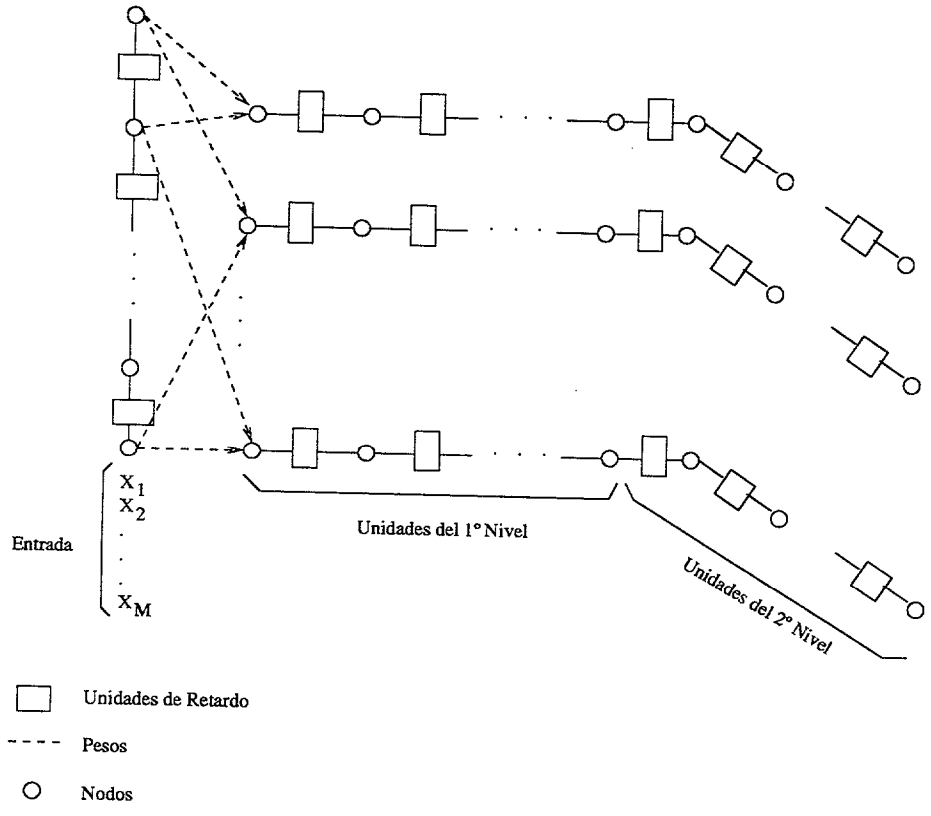

Figura 2.15: Red TDNN para un reconocedor de fonemas.

Otro de los incovenientes que presentan las redes dinámicas es el escalado temporal, cómo determinar la longitud temporal del patrón de entrada para que la red neuronal dé una salida correcta.

Concluímos este apartado añadiendo, que las redes neuronales dinámicas presentan una serie de incovenientes a la hora de trabajar con ellas. Requieren de estructuras complejas, además de un entrenamiento tanto en el espacio como en el tiempo, manifestando un comportamiento inestable ante cualquier variación. Este es el motivo principal por lo que no se ha trabajado con esta estructura.

2.7 Modelos ocultos de Markov, HMM

2.7.1 Introducción

Los Modelos ocultos de Markov son modelos estadísticos los cuales utilizan únicamente las propiedades estadísticas de la señal. Desde este punto de vista la señal puede ser vista como un proceso aleatorio. Son un importante modelo de clasificación de patrones temporales. La razón de ésto está en lo siguiente:

1. Los HMM son capaces de modelar la variabilidad temporal de las secuencias de observación.
2. Durante la fase de entrenamiento, cada modelo es entrenado para ser modelos generadores (reconocedores) de secuencias de observación. En la fase de test, se le proporciona una secuencia de observación y cada modelo estima la probabilidad de que esa secuencia haya sido generada por él o no. La dinámica de la secuencia temporal está gobernada por la matriz de transición de probabilidades de estados, ver [Rabi92].

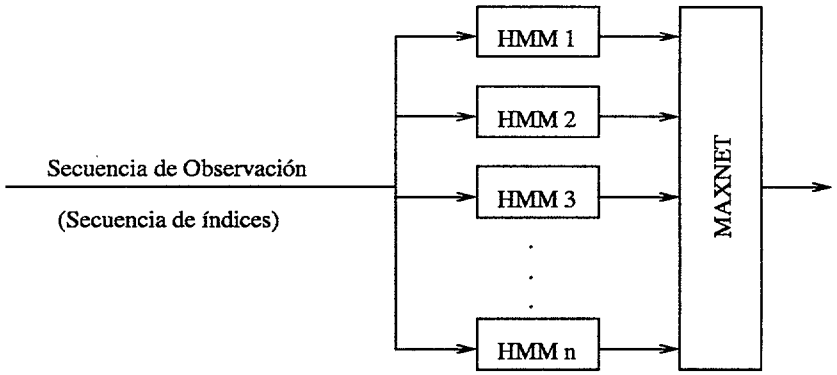

Figura 2.16: Modelos de Ocultos de Markov.

2.7.2 Proceso discreto de Markov

Consideremos un sistema, el cual puede ser descrito como un conjunto de estados $S_1, S_2, ...S_N$, que se puede encontrar en uno de ellos en cualquier momento, ver figura 2.17

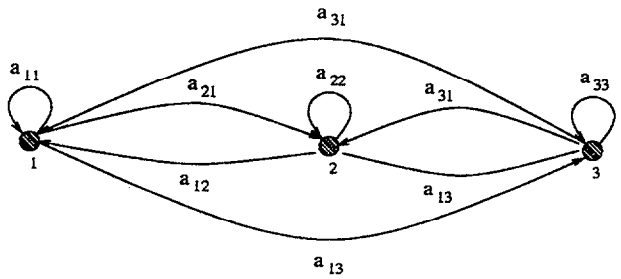

Figura 2.17: Gráfico de estados de los HMM

A cada intervalo de tiempo el sistema experimenta un cambio de estado (hacia adelante, hacia atrás, o permanecer en el mismo estado). Denominaremos a cada intervalo de tiempo $t = 1, 2, ..., T$ y al estado actual q_t.

Para una completa descripción probabilística del sistema, se requiere de especificaciones del estado actual así como de los estados anteriores. Sin embargo, para el caso especial de un sistema discreto, sólo es necesario la información probabilística del estado actual y del predecesor.

$$P[q_t = S_j \mid q_{t-1} = S_i] \tag{2.34}$$

Otra forma de expresar la ecuación anterior es mediante la matriz de probabilidades de transición de estados, dada por a_{ij}

$$a_{ij} = P[q_t = S_j \mid q_{t-1} = S_i] \qquad 1 \leq i, j \leq N \tag{2.35}$$

Cumpliéndose las propiedades de $a_{ij} \geq 0$ y $\sum_{i=1}^{N} a_{ij} = 1$.

La salida de este proceso estocástico es el conjunto de estados en cada instante de tiempo donde cada estado se corresponde con un evento físico (observable). En realidad los modelos ocultos de Markov van más allá, pues incluye la observación como una función probabilística del estado, es decir, hay dos procesos estocásticos ejecutándose, donde uno de ellos es oculto o simplemente no observable, porque sólo puede ser observado a partir del otro proceso estocástico que produce la secuencia de observación. En [Rabi92] se exponen algunos ejemplos tales como el lanzamiento de una moneda, la extracción de las bolas de colores de diferentes urnas, etc. Con este último ejemplo es muy sencillo entender el doble proceso, consideremos tres urnas y bolas de colores en cada una de

ellas. El proceso físico es observar la secuencia de colores de las bolas que un niño va extrayendo de las urnas. La secuencia de colores se ve afectada por dos procesos; primero la elección de la urna y segundo la elección de la bola. Es por ello que se denominan modelos ocultos de Markov, porque uno de los procesos no es observable.

Haciendo una comparación con este ejemplo, la elección de las urnas viene determinado por la matriz de transición de estados, que determina la probabilidad de que el modelo vaya de un estado, a otro o permanezca en el mismo. Además será necesario definir una matriz de probabilidades de la secuencia de observación, (colores de las bolas), que modele el proceso estadístico de lo que ocurre en cada estado (urna).

2.7.3 Elementos de un HMM

Un HMM se caracteriza por los siguientes parámetros, utilizaremos la misma nomenglatura que en [Rabi92]:

1. **N** es el número de estados del modelo (o de urnas). Aunque los estados son ocultos en muchas aplicaciones existe una dependencia con esta variable. Generalmente los estados están interconectados, ver figura 2.17, de tal forma que un estado pueda ser alcanzado por otro estado. Puede haber aplicaciones donde se requiera otro tipo de conexiones, donde el sistema avance de izquierda a derecha, también conocido como *modelo de Bakis*, permitiendo sólo pequeños saltos hacia la derecha de distancia no superior a dos estados.

2. **M** es el número de símbolos de observación por estado (Número de colores posibles de las bolas). Denominaremos los símbolos individuales como $V = v_1, v_2, ...v_N$.

3. La matriz de transición de probabilidades $A = \{a_{ij}\}$, ver ecuación 2.35. Para el caso general donde cualquier estado puede alcanzar a otro, $a_{ij} > 0 \quad \forall i, j$, sin embargo para otro tipo de conexiones de HMM, $a_{ij} = 0$ para uno o más pares de i, j, por ejemplo el modelo de izquierda a derecha o modelo de Bakis.

4. La matriz de probabilidades de los símbolos de observación, $B = \{b_j(k)\}$. Determina la probabilidad que estando en el estado j ocurra o suceda la observación v_k.

$$b_j(k) = P[O_t = v_k \mid q_t = S_j] \qquad 1 \leq j \leq N \qquad 1 \leq k \leq M \tag{2.36}$$

5. La matriz de distribución de estados iniciales $\Pi = \{\pi_i\}$.

$$\Pi_i = P[q_1 = S_i] \qquad 1 \leq i \leq N \tag{2.37}$$

Como se ha podido observar, para que un modelo HMM quede perfectamente definido son necesarios los parámetros **N** y **M**. Además de definir las matrices de transición **A**, de

probabilidad de secuencias de observación **B**, y matriz inicial de estados Π. La notación utilizada a la hora de referirnos a un modelo HMM vendrá especificado por $\lambda = (A, B, \Pi)$.

Para implementar los modelos ocultos de Markov hay que resolver además los tres problemas fundamentales:

1. Cálculo de la probabilidad de una secuencia de observación, dado el HMM.
2. Determinar la mejor secuencia de estados.
3. Ajustar los parámetros del modelo para una secuencia de observación.

En [Rabi92] se analiza por separado estos tres problemas para el diseño de los HMM.

2.7.4 Los 3 problemas básicos de un HMM

Hay 3 problemas básicos de interés que han de resolverse para que los HMM tengan aplicación en el mundo real. Estos problemas son:

1. Dado una secuencia de observación $O = O_1, O_2, ..., O_T$ y un modelo $\lambda = (A, B, \Pi)$, ¿cómo se calcula la probabilidad $P(O \mid \lambda)$ de que la secuencia de observación O haya sido generada o no por el modelo λ? , o que el modelo la reconozca como suya?
2. Dado la secuencia de observación $O = O_1, O_2, ..., O_T$ y un modelo λ, ¿cuál es la secuencia óptima de estados $Q = q_1, q_2, ..., q_T$?
3. ¿Cómo se ajusta el modelo λ para maximizar la $P(O \mid \lambda)$?, es decir ¿cómo ajustar el modelo para ser generadores de una determinada secuencia de observación? La secuencia de observación utilizada para ajustar los parámetros del modelo se denomina secuencia de entrenamiento. Cada modelo deberá ser entrenado para que secuencias de observación de una determinada clase la reconozca como suya.

Los modelos ocultos de Markov definen un modelo por clase a clasificar. Para la construcción de los modelos individuales será necesario resolver el problema 3, ver [Rabi92], [Rabi93], ésto sería la fase de entrenamiento. Una vez que el conjunto de modelos han sido diseñados, se pasaría a la fase de reconocimiento de un patrón desconocido para ello es necesario dar solución al problema 1. Aquel modelo que reconozca la secuencia de observación como suya dará un valor alto, próximo a la unidad y aquellos modelos que dan una probabilidad baja a la salida implica que la secuencia de observación no fue generada por ellos.

En [Rabi92] y [Rabi93] se expone los procedimientos y algoritmos necesarios para resolver estos problemas, necesarios en la implementación de los HMMs.

2.7.5 Tipos de HMMs

El caso general de modelos HMMs son los ergódicos, es decir es aquel modelo que estando en un estado se puede saltar a otro estado, el modelo se encuentra completamente conectado. Para el caso de un modelo de 3 estados, la matriz de transición de estados vendría dada por la matriz A, ver ecuación 2.38

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \tag{2.38}$$

Sin embargo para algunas aplicaciones, interesa utilizar otro tipo de estructuras de HMM, ya que funcionan mejor que los modelos ergódicos, tales como los modelos denominados de izquierda a derecha o modelo de Bakis, en este caso el modelo progresa de izquierda a derecha o permanece en el mismo estado pero nunca puede volver hacia atrás ya que no existen tales conexiones. Este tipo de modelos tiene la propiedad que fácilmente pueden modelar señales cuyas propiedades cambian con el tiempo como es el caso de la señal de voz. Todo ésto vendría especificado en la matriz de transición de estados **A**. La condición sería: $a_{ij} = 0$ donde $j < i$ y por otro lado la matriz de estados iniciales cumple la siguiente propiedad

$$\pi_i = \left\{ \begin{array}{ll} 0, & i \neq 1 \\ 1, & i = 1 \end{array} \right\} \tag{2.39}$$

Por lo que la secuencia de estados debe comenzar por el estado 1 y finalizar en el estado N. En este tipo de modelos a veces se utiliza alguna restricción en el sentido de que no se permiten grandes cambios en los saltos. Ésto quiere decir que saltos de 2 o más estados no son permitidos.

2.8 Conclusión

El método de clasificación de patrones es una técnica de clasificación para secuencias estáticas, carentes de memoria y no se tiene en cuenta para nada el carácter temporal de la señal. Sin embargo es un método muy eficiente para reducir el espacio de vectores de entrada a un conjunto reducido de vectores representativos de los demás.

Las técnicas de programación dinámica tienen su importancia en la medida que trata el problema del alineamiento temporal de las secuencias, que se utilizan de manera combinada con otros clasificadores como el cuantificador vectorial.

Las redes neuronales estáticas, son las redes más conocidas y utilizadas en una gran variedad de aplicaciones. Presentan una buena capacidad de discriminación en tareas de clasificación de secuencias estáticas, sin embargo, son inadecuadas en la clasificación de secuencias temporales debido a la ausencia de memoria ya que no puede almacenar información pasada. Es más aunque presenten unidades de retardo en sus conexiones hay que disponer de un apropiado número de unidades de retardo, ya que una secuencia de patrones puede ser diferente a otra. Por lo tanto, son las redes recurrentes las redes que permiten obtener un modelo dinámico, lo que les permite almacenar, recuperar la información de una manera flexible. En la actualidad, hay desarrollándose muchísimas investigaciones sobre este tipo de redes, sobre todo en la etapa de aprendizaje o entrenamiento. Sin embargo estas redes neuronales presentan muchas dificultades a la hora de su implementación ya que son sistemas que presentan mucha dificultad a la hora de entrenarse con técnicas de gradiente conjugado. Se ha demostrado que cualquier sistema dinámico con una recurrencia no lineal, tales como las redes neuronales recurrentes, es cada vez más difícil de entrenar con técnicas de gradiente cuando la duración de las secuencias se extienden en el tiempo. Hay trabajos anteriores sobre alternativas de algoritmos de entrenamiento basados en la propagación en el tiempo hacia atrás de las variables espacio-estado en vez del error.

Por otro lado, el gran problema de los modelos HMM está en que por ejemplo para el reconocimiento de dígitos aislados se entrena un modelo para cada dígito a clasificar. Su entrenamiento sólo se hace con secuencias de observación de la clase en cuestión. Este tipo de entrenamiento hace que los modelos HMM no sean tan discriminativos. Sin embargo presentan ser una buena y eficiente herramienta para el tratamiento de secuencias temporales.

Hasta el momento son los modelos ocultos de Markov la técnica más eficiente para el reconocimiento de voz, a pesar de ser una herramienta poco discriminativa. Como solución a este problema está la posibilidad de combinar distintos modelos de clasificación formando estructuras híbridas de tal forma que entrenados adecuadamente se pueda conseguir resultados buenos, con un pequeño coste computacional. La utilización de este tipo de estructuras híbridas es parte de nuestra tesis, por lo que se hablará en el capítulo 5.

Capítulo 3

Bases de datos

3.1 Introducción

El primer paso para llevar a cabo todos los experimentos y pruebas que se pretendían desarrollar en esta tesis, era la necesidad de una base de datos de dígitos en castellano. Nos encontramos en el dilema si bien, de crear nuestra propia base de datos o sondear en alguna de las Universidades con las que se tenía contacto, y que dispusiese de alguna. En la época en la que se comenzó esta tesis, año 1995, sólo se tuvo constancia de la UPC, Universidad Politécnica de Cataluña, la cual comenzaba a trabajar en el diseño de una base de datos en castellano mucho más ambiciosa, que incluía números aislados, cantidades numéricas, frases, palabras, etc. e integraba a un gran número de locutores. Por otro lado, nos llegaron algunas noticias de que quizás Telefónica I+D, contase ya con una base de datos. Igualmente, ese mismo año se nos pidió la colaboración al objeto de realizar grabaciones a los alumnos de nuestra escuela, ya que se encontraban en la fase de creación también. A la vista de que era casi imposible conseguir una base de datos de dígitos de datos a corto plazo, se decidió crear una base de datos de dígitos aislados en castellano ajustada a nuestras necesidades.

Simultáneamente a la creación de la base de datos se comenzó el estudio del preprocesado más utilizado en tareas de clasificación temporal, analizándolo desde el punto de vista del reconocimiento de voz.

Posteriormente, con la realización de los proyectos fin de carrera [Ciro99-pfc] y [Camino99-pfc] se hizo posible disponer de otras señales diferentes a la de voz: texto manuscrito y firmas.

En este capítulo se hablará de las bases de datos diseñadas y utilizadas en los experimentos.

3.2 La señal de voz

La voz es una señal primordial, ya que interviene en muchas de las aplicaciones del mundo actual, la cual ha originado múltiples áreas de investigación, tales como codificación, síntesis, reconocimiento, identificación de locutores, traducción de lenguajes, etc. [Rabi94].

En el proceso de producción de la voz intervienen como es sabido, consultar [Rabi93], muchos órganos: pulmones, cuerdas vocales, laringe, etc.; las distintas posiciones de la cavidad bucal: labios, lengua, velum, etc. Hacen que con la toma de aire en los pulmones, éste sea expelido hacia la traquea y actúe como excitación del tracto vocal, donde las cuerdas vocales se tensarán y vibrarán al paso del flujo del aire. Este flujo de aire será modulado en frecuencia al pasar por la faringe y cavidad bucal. Dependiendo de la posición de los labios, lengua, etc. se originan los distintos sonidos. La secuencia de estos sonidos produce la **señal de voz**. El proceso de producción de la señal de voz nos demuestra el carácter temporal de esta señal, ya que podemos verlo como un proceso de producción de sonidos diferentes que combinados de una manera u otra pueden tener significados diferentes. Ésto demuestra que la formación de una palabra, frase, etc. y que a su vez sea inteligible, dependa de la secuencia de sonidos en el tiempo.

En este sentido, es importante conocer el mecanismo que tiene el cuerpo humano de percibir los sonidos, ya que esto nos ayudará a desarrollar un modelo cuyo comportamiento a muy bajo nivel, pueda asemejarse al sistema auditivo del hombre.

El proceso de percepción de voz es más complicado que el de la producción. En principio la persona que escucha procesa la señal acústica a lo largo de la membrana basilar del oído interno, el cual genera un análisis espectral de la señal entrante. A continuación, se produce un proceso de conversión de la señal espectral a una señal que contiene características extraídas a partir de la señal espectral. Esta señal viaja a lo largo del nervio auditivo, y de una manera no descubierta aún, se produce el proceso de comprensión en el cerebro.

Una vez comentado brevemente el proceso de producción y percepción de la señal de voz, pasamos a comentar detalles en cuanto a la forma de onda. La señal de voz, [Rabi93] es una señal que evoluciona lentamente con el tiempo, siempre y cuando se examine en un periodo de tiempo corto, entre 5 y 10 mseg. Dentro de este período, se observa que las características de la señal permanecen cuasi estacionarias. Sin embargo, para intervalos de tiempo superiores, entre 1 y 5 segundos, las características de la señal de voz cambian, pudiéndose observar los distintos sonidos.

La señal de voz es una señal compleja en la cual la suposición de casi estacionariedad no es siempre cierta pues los mecanismos de producción pueden producir cambios rápidos de gran importancia. Otros ejemplos son los cambios vocal-consonante o las transiciones entre un silencio y un fonema oclusivo. Ésto nos sitúa en una visión temporal de las

cualidades de la voz. De la forma de onda, también se puede obtener información acerca de los cambios de silencio a sonoridad y viceversa. Se sabe además, que el contenido frecuencial de la voz es muy rico y puede cambiar con el tiempo, incluso en los casos en que la señal es estacionaria.

3.2.1 Diseño de la base de datos de voz

Aprovechando la proximidad con el colectivo estudiantil, se pensó que podrían ser ellos los principales locutores, consiguiendo así una amplia variedad de individuos. Crear una gran base de datos con muchos locutores y muchas grabaciones por locutor era muy pretencioso, y además complicado. No todo el mundo estaría de acuerdo con el sometimiento a una grabación, a sabiendas de que le esperaban muchas horas delante de un micrófono. Es por lo que se decidió construir dos bases de datos diferentes:

1. Base de datos 1. Está formada por 5 locutores con 132 realizaciones o grabaciones por locutor. Esta base tiene pocos locutores, ya que el número de grabaciones por locutor es considerable. De los 5 locutores, 2 son femeninos y 3 masculinos. De cada locutor se realizaron 132 grabaciones de los dígitos del 0 al 9. El total de ficheros grabados son: $132 \times 5 = 660$. Podemos añadir que aunque el objetivo principal fue 132 grabaciones por locutor, en la actualidad se disponen de algunos locutores más realizaciones, por lo tanto siempre haremos referencia al tamaño de los conjuntos de entrenamiento y test para cada prueba. El disponer, en el caso de algunos locutores, de mayor número de grabaciones, nos permitirá realizar algunas pruebas experimentales sobre la dependecia de los resultados cuando se aumenta o cambia el conjunto de entrenamiento o de test, etc.
2. Base de datos 2. Formada por 75 locutores de 11 realizaciones o grabaciones por locutor. Como se puede ver, esta segunda base de datos tiene un mayor número de locutores, pero el número de repeticiones por locutor es menor.

La creación de esta base de datos, conlleva una serie de fases tales como: grabación, etiquetado y preprocesado de la señal. La primera fase, grabación es la captura de la señal de voz, $s(t)$ y convertirla a formato de datos discretos con los que se pueda trabajar, $s[n]$. La segunda fase, a la que se ha denominado etiquetado, recoge el proceso de separación de la señal grabada en los distintos dígitos. La última fase trata de la conversión de las señales en parámetros. A continuación se trata cada fase de manera extensiva.

3.2.1.1 Grabación

Aunque el primer paso del reconocimiento de voz es la parametrización de la señal, existe un paso previo, que es la captura de la señal de voz a través de un micrófono y su

conversión a una señal digital. Se decidió, que la interfaz que convierta la señal de voz en una secuencia de muestras, fuese una interfaz sencilla y asequible, de tal forma que las pruebas que se realizarán, fueran aplicables al mundo real y puesto en marcha en cualquier equipo. Para tal propósito se eligió una tarjeta de sonido Sound Blaster 16 ASP, disponible en el mercado a un precio muy económico.

Por lo tanto, el sistema de grabación está formado por una tarjeta SOUND BLASTER 16 ASP, con el micrófono por defecto. Todas las grabaciones se han llevado a cabo en un ambiente de oficina, laboratorios etc. donde la presencia de personas era inevitable con el consecuente ruido ambiental y de murmullo. Toda tarjeta de sonido trae consigo software y aplicaciones que permiten grabar, reproducir, amplificar, añadir ruido, añadir eco, etc. Sin embargo, no permiten seleccionar tan libremente los parámetros de grabación. Sólo se permiten seleccionar un rango determinado de parámetros, véase 3.1.

Frecuencia de muestreo	Bits	Mono/Estéreo
8000	8	Mono
	8	Estéreo
	16	Mono
	16	Estéreo
11025	8	Mono
	8	Estéreo
	16	Mono
	16	Estéreo
22050	8	Mono
	8	Estéreo
	16	Mono
	16	Estéreo
44100	8	Mono
	8	Estéreo
	16	Mono
	16	Estéreo

Tabla 3.1: Parámetros de grabación de una tarjeta de sonido SOUND BLASTER 16 ASP

Analizando la tabla 3.1 se decidió el siguiente formato de grabación: frecuencia de muestro 8000 KHz, 16 bits por muestra, mono. Este formato parecía ser el más generalizado y utilizado en otros entornos y aplicaciones de procesado de la señal. Se grabó en formato WAVE y posteriormente cada uno de los ficheros se convirtieron a formato de MATLAB para poder trabajar con ellos.

El proceso de grabación más adecuado para no cansar a los locutores fue el de grabar los dígitos del 0 al 9 de forma continua, es decir en un mismo fichero. Además se tuvo en cuenta que durante la fase de grabación de la base de datos 1, que el número de grabaciones por día no superase a 25, para no provocar la fatiga del locutor. Sin embargo, para la base 2, se realizaron las grabaciones en una misma sesión.

Una vez obtenidas las señales grabadas se pasó a la siguiente fase: la separación de los dígitos o etiquetado.

3.2.1.2 Etiquetado

El hecho de que la grabación se realizara de forma continua, es decir todos los dígitos en un mismo fichero, introducía un problema, no se conocía la duración de cada dígito, su principio y final, por lo fue necesario etiquetar cada fichero. Para las grabaciones de ambas bases de datos no se disponía de herramientas adecuadas para etiquetar cada fichero, es decir, marcar donde comienza y finaliza cada dígito de manera automática. Se decidió mantener el fichero intacto en formato MATLAB añadiéndole una variable que indicara el comienzo de cada dígito. Por tanto, todos y cada uno de los ficheros fueron etiquetados con comprobación visual y auditiva, tarea que tengo que calificar como pesada y muy laboriosa. En la figura 3.1 aparece la forma de onda de uno de los ficheros grabados. Junto a esta señal se grabará la variable *labels* que indique el comienzo de cada dígito.

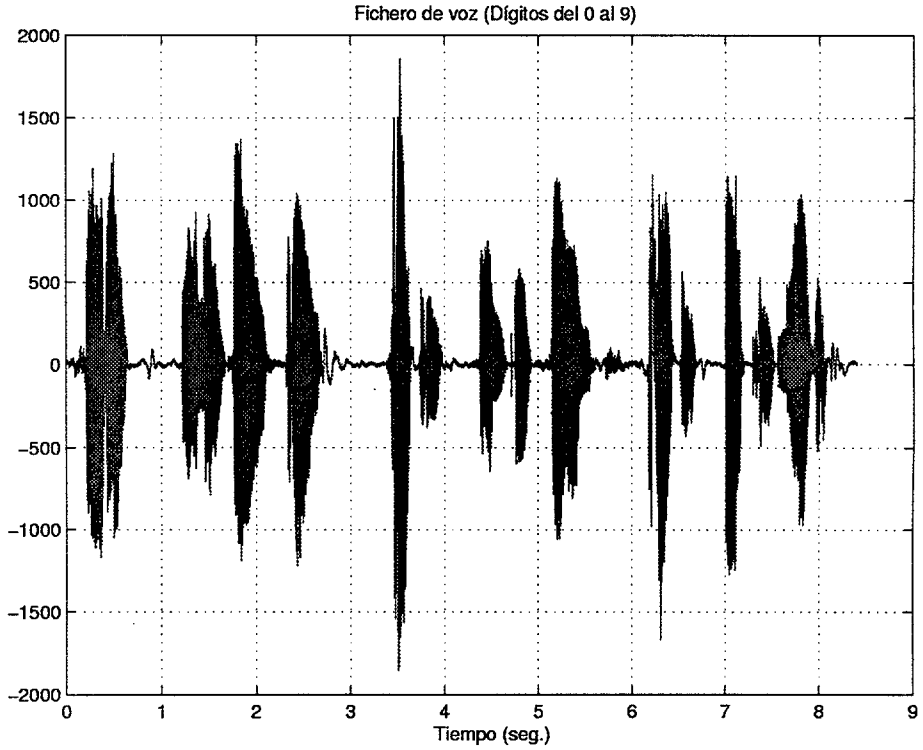

Figura 3.1: Forma de onda de un fichero de grabación.

Los ficheros grabados fueron denominados de la siguiente manera, *lcy-xxx.mat*. La variable *y* nombra al locutor numéricamente y la *xxx* denomina a la grabación, puede variar desde 1 hasta 132 o superior en algunos casos. Este fichero guarda en su interior la variable *voz* y *labels*. La primera guarda las muestras de la señal de voz en un vector *voz* y la segunda variable *labels* contiene el principio de cada dígito, se trata de un vector

columna de dimensión 11×1. El final de cada dígito es el comienzo del siguiente menos 1. Por ejemplo para trabajar con el dígito 0, $voz(labels(1) : labels(2) - 1)$.

Con la ayuda de un proyecto final de carrera, [J.Alonso98], se pudo automatizar el proceso de grabación. La aplicación desarrollada es un detector de actividad de voz, en tiempo real y ejecutable en cualquier PC i80486 en entorno de Windows. La función principal es la detectar o separar el silencio de lo que es voz aplicando técnicas espectrales. Todo lo referente a este proyecto se encuentra recogido en [J.Alonso98].

Esta aplicación permite grabar, etiquetar automáticamente con comprobación visual y auditiva y su posterior grabación en formato MATLAB.

Como ya se ha comentado en el párrafo anterior nuestras señales han sido obtenidas con la tarjeta SOUND BLASTER 16 ASP, con el micrófono por defecto y en un ambiente ruidoso.

Las mayoría de las pruebas y experimentos realizados se han hecho con la primera base de datos. Del conjunto de las 132 grabaciones que se tienen de cada locutor se ha divido en 2 conjuntos: entrenamiento y test. El tamaño de cada conjunto dependerá de la prueba o experimento que se haya llevado a cabo, aunque en la mayoría de las pruebas, el conjunto de test está formado por 100 realizaciones y el de entrenamiento por las 32 restantes.

Lo importante del proceso de digitalización es producir una representación discreta de la señal con una alta relación señal a ruido, en lo sucesivo SNR. Sin embargo, se detectó que el micrófono utilizado introducía ruido indeseado tal como el procedente de la línea de alimentación del sistema de grabación ("50 Hz"), por lo que hizo falta un filtro que eliminara dicha señal.

Una vez vistos los pormenores de la creación de la base de datos el siguiente paso a analizar es el pre-procesado de la señal.

3.2.2 Pre-procesado de la señal

El preprocesado de la señal implica: filtrado, segmentación, enventanado y parametrización. La primera fase consiste en enfatizar aquellas componentes importantes de la señal. La segunda fase, segmentación, es el proceso, en el que atendiendo al principio de que la señal de voz se comporta de manera cuasi estacionaria únicamente en intervalos inferiores a 10 mseg. es necesario dividir la señal. Ésta se segmenta en itervalos de tiempo inferiores a 10 mseg. con solape de información. Para suavizar los cambios bruscos entre los segmentos se suele aplicar una ventana del tipo Hamming, ver [Pico93], ésta sería la tercera fase dentro del preprocesado. La cuarta fase y última es la parametrización. Es la fase en la que se obtienen los parámetros representativos de cada segmento de la señal. éstos

parámetros han de contener información sobre los cambios con el tiempo del espectro de la señal. Dentro del preprocesado es la parte más importante y es el primer paso en el proceso de reconocimiento de voz.

Hay muchas técnicas que obtienen una representación paramétrica con un significado perceptual; parámetros que simulan algún comportamiento observado en el sistema auditivo humano. Hay tres aspectos importantes en el modelado de la señal.

1. La parametrización ha de buscar aspectos sobresalientes de la señal, osea parámetros que sean análogos a aquellos utilizados por el sistema auditivo humano.
2. La parametrización ha de ser robusta frente a variaciones del canal, locutor, transductor, etc.
3. Los parámetros han de captar los cambios con el tiempo del espectro de la señal.

Como se observa en la figura 3.2 hay dos partes diferenciadas; la primera es la captura de la señal de voz, parte hardware (comentada en los apartados anteriores) y la segunda se trata del preprocesado de la señal de voz, parte software, desarrollada en el entorno de MATLAB

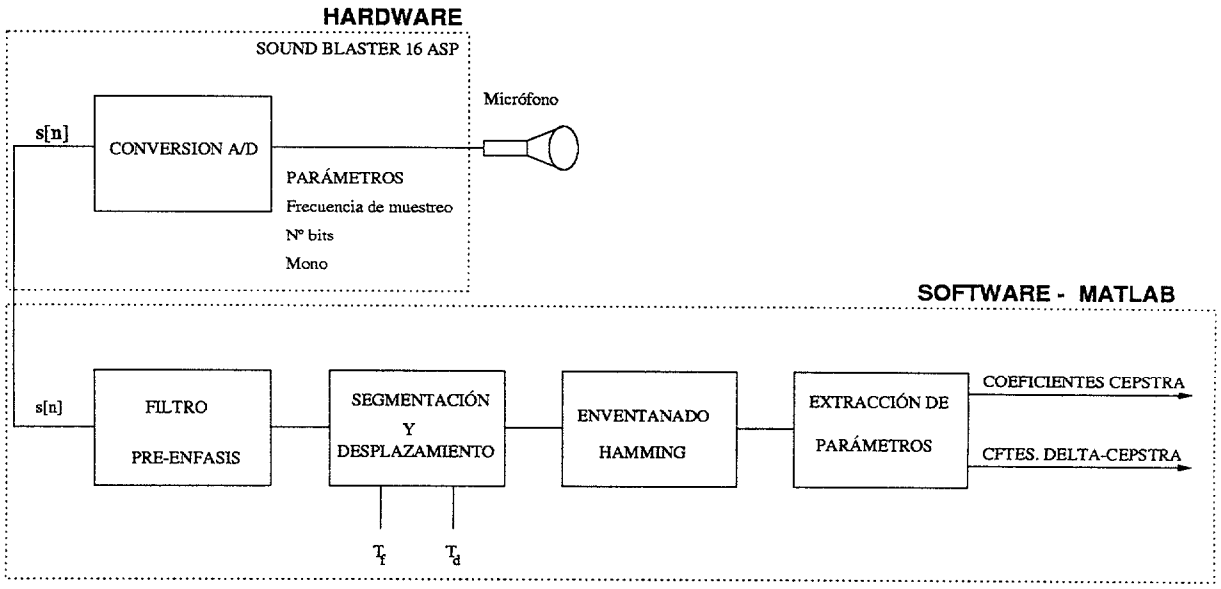

Figura 3.2: Diagrama de bloques del procesado de la señal.

3.2.2.1 Filtro de pre-enfasis

El primer paso será filtrar la señal de voz con un filtro de pre-énfasis con el fin de enfatizar aquellas componentes importantes de la señal. Se trata de un filtro FIR de primer orden, ver 3.1.

$$H_{pre}(z) = 1 + a_{pre} \cdot z^{-1} \tag{3.1}$$

Un valor típico del coeficiente a_{pre} es de 0.95. El objetivo de este filtro es la de levantar el espectro de la señal, aproximadamente 20 dB por década. Una explicación de por qué se utiliza puede ser que las partes de la señal de voz sonoras tienen una pendiente espectral negativa (de atenuación) de aproximadamente 20 dB por década debido a las características fisiológicas de producción de la voz. O bien es que el oído es más sensitivo a la región por encima de 1 KHz. El filtro de pre-énfasis amplifica este área del espectro.

En la figura 3.3 se puede ver la función de transferencia de este filtro.

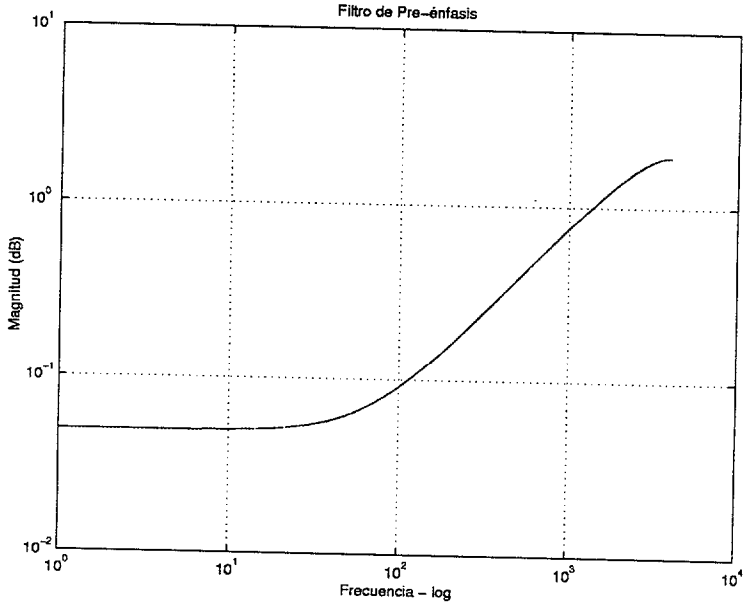

Figura 3.3: Función de transferencia del filtro de pre-énfasis.

3.2.2.2 Segmentación y desplazamiento

No se suele trabajar directamente con la señal $s[n]$ sino a partir de ella, se obtienen una serie de parámetros de los cuales se pueda extraer información de los cambios con el tiempo del espectro de la señal. Estos parámetros no pueden evaluarse directamente sobre el tiempo total de señal de la cual se disponga sino a intervalos cortos donde la señal se comporte de manera cuasi estacionaria.

La señal de voz, [Rabi93] es una señal que evoluciona lentamente con el tiempo siempre y cuando se examine en un periodo de tiempo corto, entre 5 y 10 mseg. Dentro de este período se observa que las características de la señal permanecen cuasi estacionarias. Sin

embargo, para intervalos de tiempo superiores, entre 1 y 5 segundos las características de la señal de voz cambian, pudiéndose observar los distintos sonidos.

Una vez que la señal ha sido filtrada, ésta es dividida en segmentos o tramas de T_f(mseg), y sobre esta trama se extraen los parámetros. Se define T_f como la longitud de tiempo sobre el cual un conjunto de parámetros es válido. Típicamente se suele utilizar un T_f entre 10 y 30 mseg. Igualmente importante es el desplazamiento, las tramas adyacentes están solapadas, es decir el final de la trama no es el comienzo de la siguiente sino que el comienzo de la siguiente es el mismo que la trama actual pero desplazada T_d(mseg). Por tanto, las tramas futuras comparten datos de la trama anterior y a su vez con la trama futura. El valor de T_d depende de T_f, por ejemplo para una trama de 30 mseg. el desplazamiento puede ser de 10 ó 15 mseg. Una vez que la señal de voz se ha dividido en distintos segmentos se extraen los parámetros característicos de cada uno de ellos. Al compartir la información entre tramas consecutivas, los parámetros de cada trama están correlados entre sí.

3.2.2.3 Enventanado

El siguiente paso será aplicar a la señal segmentada una función de ventana, entre las que se encuentran las ventanas: rectángular, Hanning, Hamming, Blackman, Bartlett y Kaiser. Entre éstas es la ventana Hamming la más utilizada en reconocimiento. El objetivo de la ventana es ponderar las muestras hacia el centro de la ventana, ésto permite un suavizado de los parámetros a extraer, junto con la segmentación y desplazamiento visto anteriormente.

3.2.2.4 Parametrización

Hay varios algoritmos utilizados entre los que se encuentran: banco de filtros digitales, coeficientes cepstra, coeficientes de predicción lineal, etc. Nos centraremos únicamente en los coeficientes cepstra debido a que son coeficientes muy robustos.

Los coeficientes cepstra proporcionan una buena representación de las propiedades espectrales de la señal de voz, para una trama dada. Una representación mejorada puede ser, si se incluye información acerca de la derivada temporal cepstra, ambas, la primera y segunda derivada. Se ha comprobado que mejoran el funcionamiento del sistema de reconocimiento cuando esta información es tenida en cuenta. En las simulaciones se ha trabajado con los coeficientes cepstra y los obtenidos de la primera derivada, obteniendo 13 coeficientes cepstra y 13 coeficientes delta-cepstra haciendo un total de 26 coeficientes.

3.2.3 Etiquetador

La siguiente etapa al pre-procesado de la señal es la cuantificación vectorial. En el capítulo anterior se hizo una introducción teórica del funcionamiento del cuantificador vectorial. Una vez que los coeficientes cepstra han sido calculados, se procede a la siguiente etapa. El sistema es un reconocedor discreto de dígitos, por lo que se necesita convertir el conjunto continuo de coeficientes en un conjunto discreto, más pequeño y representativo del anterior.

Como sistema etiquetador se comenzó a ensayar con un VQ entrenado con el algoritmo LBG, descrito en [Makhoul85] e introducido en el capítulo anterior. El problema de este clasificador, basado en el algoritmo de las K-medias es su escasa capacidad de generalización [Cerf94], característica que puede influir negativamente en el conjunto global del reconocedor. Por esta razón también se consideró ensayar con redes neuronales como etiquetador, las cuales tienen una mayor capacidad de generalización. Esta idea no es nueva y ya ha sido empleado en [Cerf94] con perceptrones multicapa MLP [Hush93].

En el estudio de las redes neuronales como sistema etiquetador, se tienen diversas alternativas. La primera ensayada fue un perceptron multicapa, MLP. Las pruebas realizadas mejoran en dos o tres puntos la tasa de reconocimiento pero sus altos requerimientos computacionales nos persuadieron para abandonarlas. Además se encontró el incoveniente de que las redes neuronales MLPs se entrenan con un entrenamiento supervisado, es decir al mismo tiempo que se le suministra la entrada, se le proporciona la salida que debería dar la red neuronal. Esta señal de salida fue la obtenida por el VQ. El resultado de esto es entrenar una red neuronal MLP para que aprenda a dar la misma respuesta que el VQ. Este es el motivo principal que se desechara del estudio como etiquetador. Lo mismo ocurre con las redes RBF, al tratarse de redes con entrenamiento supervisado, aunque se dejó de un cierto margen de error para evitar que la NN se ajustara al VQ. Todo esto llevó a elegir el VQ como etiquetador.

Los coeficientes cepstra y delta-cepstra serán cuantificados vectorialmente, ver [Makhoul85]. Para ello será necesario disponer previamente de una biblioteca de vectores representativos. El utilizar un cuantificador vectorial tiene sus ventajas y desventajas. Entre las ventajas citamos primero un representación discreta de la señal. Se asocia cada vector de coeficientes con los contenidos en la biblioteca. El proceso de elegir un vector de la biblioteca es equivalente a asignar un índice, el cual indica la posición que ocupa ese vector, dentro de ella. Esto nos lleva a tener una representanción de la señal mediante una secuencia temporal de índices.

En cuanto a las desventajas sería la presencia de una distorsión en los coeficientes, ya que éstos son sustituidos por otros coeficientes. Por otro lado, está la necesidad de almacenar la biblioteca. Cuando ésta es pequeña no resulta ser una desventaja, pero en el caso de ser muy grandes, está el problema de almacenarla y además el problema de manejarla, así como el tiempo de cálculo que lleva al comparar un vector de coeficientes

con todos los de la biblioteca.

3.3 Diseño de la base de datos de texto manuscrito

Esta base de datos está formada por 90 escritores. De cada escritor se dispone de 12 repeticiones de los 10 dígitos del 0 al 9. Los escritores fueron elegidos de acuerdo a su sexo, edad, etc. De cada dígito se recogerá la información geométrica de los dígitos.

La creación de esta base conlleva tres fases: preprocesado de la imagen y extracción de parámetros.

Se utilizará un reconocimiento *Off-line*, es decir, los dígitos escritos fueron capturados mediante un escáner.

3.3.1 Preprocesado de la señal

Con el fin de facilitar la extracción de parámetros se realizó una mejora de la imagen de los dígitos escritos. El primer paso fue aplicar un filtro para eliminar el ruido introducido por el propio escáner. El paso siguiente se corrige la inclinación que pueda tener el papel en el momento de escánear la imagen. Por último la binarización del gráfico que contiene al dígito [O'Gorman95].

3.3.2 Extracción de parámetros

Para la extracción de parámetros se han aplicado dos técnicas: detección de contorno y esqueletización.

- **Detección de contorno.** Está basado en la eliminación de todos los valores en negro del gráfico, excepto aquellos puntos que limitan con el contorno, dejando el trazado que bordea a todo el dígito para mantener la conectividad entre los pixeles y que contiene la información del recorrido de su escritura. El resultado que se obtiene es un contorno cerrado. Con esta curva es muy fácil pasar de una matriz de dimensión 2 a un vector mediante una codificación que capte las variaciones del trazado.

  Con respecto a esta técnica se han realizado pruebas con el tamaño original 160×160 pixeles y con una imagen reducida de 32×32 pixeles, introduciendo en ésta última un

escalado del dígito para mantener los vectores a un tamaño relativamente parecido y evitando la desproporcionalidad de tamaños en la creación de los vectores. La elección del sentido del recorrido y del punto de inicio del dígito son dos parámetros que se fijan para estándarizar el método. Sólo queda comentar que la reducción del dígito hace un efecto de filtro paso bajo, eliminando detalles muy concretos del escritor y generalizando el dígito para su reconocimiento.

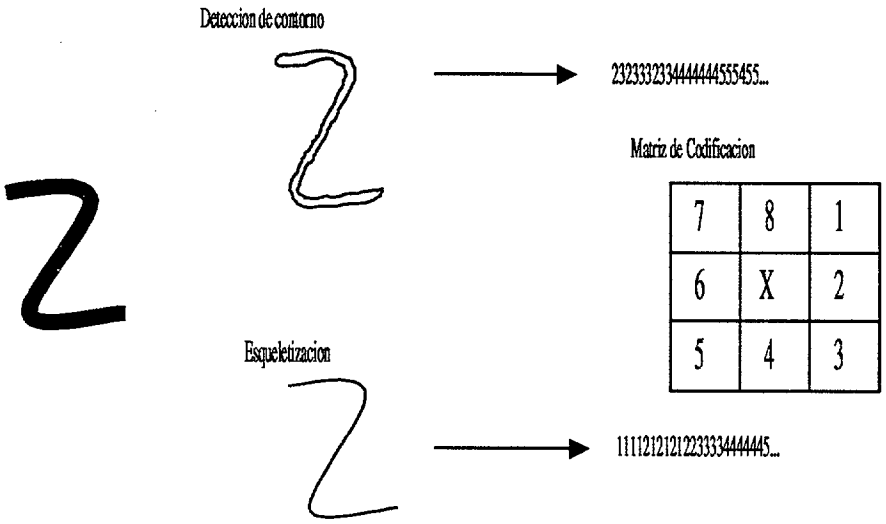

Figura 3.4: Etapa de preprocesado de imagen.

Para codificar el dígito en un vector se define una matriz de dimensión 3×3, tal como aparece en la figura 3.4. Para comenzar se busca unos pixels iniciales aplicando siempre las mismas condiciones. Esta matriz tiene en el centro, el pixel etiquetado con **X**, que es el pixel actual o pixel inicial. El siguiente paso es fijar la dirección para recorrer el bucle de pixels, moviéndola en la dirección donde se ha encontrado un pixel en negro. Esta matriz recorre todos los pixels con la finalidad de encontrar la dirección del pixel vecino y su posición en la matriz. De esta forma es como se codifica el dígito en un vector. En la figura 3.5 aparece un ejemplo de vectorización y como se genera el vector de codificación.

- **Esqueletización.** Este algoritmo consiste en la eliminación del grosor o capas exteriores del gráfico del carácter hasta que sólo queda un único trazado, con conectividad entre todos sus pixeles y que definen el interior del dígito. Igualmente, al solo tener un pixel conectado con otro, ésto facilita el paso de una matriz de dos dimensiones a un vector.

  El proceso que continúa es el mismo que para la detección de contorno. En este algoritmo se establece un pixel inicial y uno final. La matriz 3×3 barre todos los pixels conectados y retorna un vector. Los parámetros de pixel inicial y final son parámetros que se establecen al comienzo para estándarizar así el método. Todos los vectores se construyen de la misma manera.

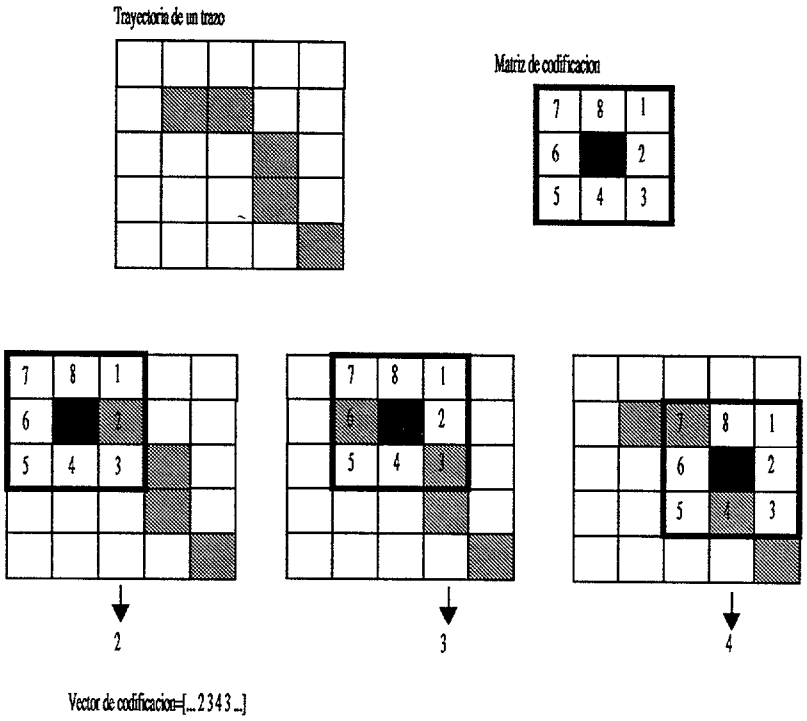

Figura 3.5: Trayectoria. Ejemplo de vectorización

Se aplicó también la reducción de la matriz original y se normalizó con la misma finalidad que para el proceso de extracción de contorno.

Para más información consutar [Ciro99-pfc].

Los resultados de las pruebas que presentamos fueron obtenidos para los datos vectorizados por el método de esqueletización.

3.4 Diseño de la base de datos de firmas

La base de datos de firmas está formada con las firmas de 60 escritores, con 24 repeticiones cada una de ellas. Cada firma fue recogida en un papel con 24 distribuciones o cajetines de distintos tamaños, ver la figura 3.7. Los escritores fueron elegidos de acuerdo a su sexo, edad, etc. Las firmas fueron capturadas mediante un escáner. Se aplicó el mismo pre-procesado que para el reconocimiento de dígitos manuscritos: filtrado de ruido, binarización, aplicación del algoritmo de esqueletización. A continuación se realizó un proceso de vectorización siguiendo el contorno de la imagen. Por último, el espacio bi-dimensional de la firma se convierte a un vector uni-dimensional. En la figura 3.6 aparece un ejemplo

de una firma, a la derecha la imagen escaneada y a la izquierda su vectorización.

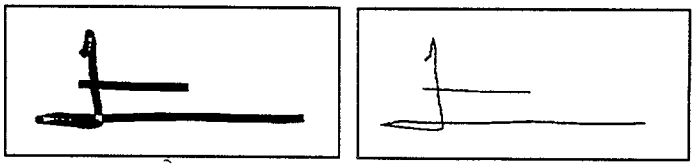

Figura 3.6: Esqueletización de una firma.

Para más información consultar [Camino99-pfc]

Los resultados de las pruebas que presentamos fueron obtenidos para los datos vectorizados por el método de esqueletización.

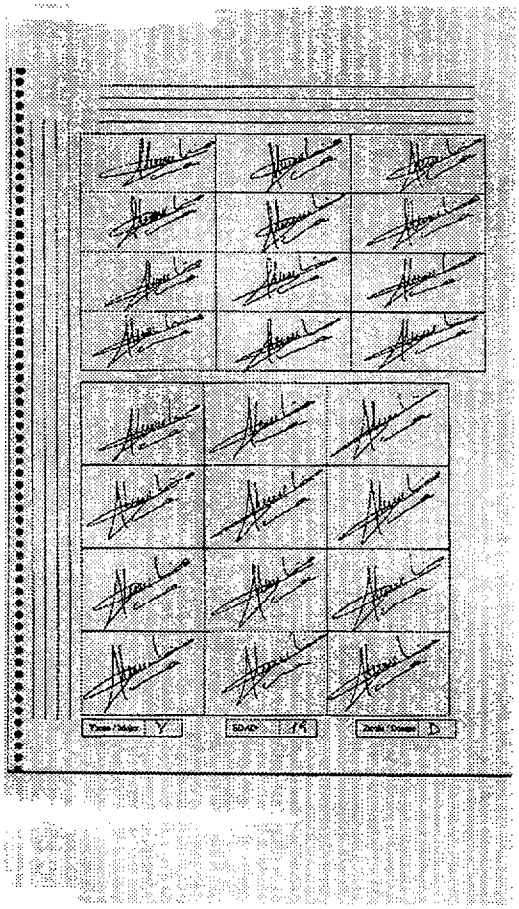

Figura 3.7: Ejemplo de hojas de firmas.

Capítulo 4

Mejora de la capacidad de la generalización de los HMMs

4.1 Introducción

Uno de los ámbitos de trabajo de esta tesis ha sido obtener una optimización de los modelos HMM. Como ya se ha comentado en capítulos anteriores, los modelos ocultos de Markov han demostrado ser una herramienta muy eficaz a la hora de trabajar con secuencias temporales, en aplicaciones tales como el reconocimiento de voz. Básicamente un HMM describe las características tanto estacionarias como dinámicas de la voz mediante un modelado espectral y un modelo de transición de estados [Rabi92], [Rabi93]. El entrenamiento de dicho modelo se realiza maximizando un criterio de máxima verosimilitud mediante el algoritmo Baum-Welch. A pesar de su uso bastante extendido, hay algunas características que no han sido bien estudiadas como son la velocidad de aprendizaje, convergencia, sensibilidad al modelo inicial, criterios de parada, etc. Todos estos aspectos afectan más a la capacidad de generalización del sistema final que al algoritmo de entrenamiento del HMM.

Una medida de la capacidad de generalización de un sistema de aprendizaje es cómo aproxima una salida a una entrada nunca vista. Un método habitual para medirla es:

1. Dividir la entrada en dos subconjuntos, uno llamado secuencia de entrenamiento y otro llamado secuencia de test,
2. Entrenar el sistema con la secuencia de entrenamiento y
3. Medir la capacidad de generalización de la secuencia de test [Sarkar96].

Como se observa es una técnica muy sencilla y habitual en el entrenamiento de cualquier modelo. Pero el estudio para que el modelo generalice bien va más allá de partir de un modelo, de un conjunto de entrenamiento y test, etc. El problema radica, en averiguar la sensibilidad del sistema en:

1. La dependencia con el modelo inicial.
2. Relación de la complejidad del modelo.
3. ¿Qué tamaño debe tener el conjunto de entrenamiento para que el sistema funcione correctamente?
4. ¿Cuál es la dependencia con el conjunto de entrenamiento?
5. ¿Cuándo se debe parar el entrenamiento, para que no se produzca un sobreentrenamiento u *overfitting*, usando la terminología inglesa?

Hasta el momento, una táctica razonable para mejorar la capacidad de generalización ha sido aumentar el número de grados de libertad aumentando el tamaño de la secuencia de entrenamiento. Este método no siempre es viable, ya que conseguir grandes secuencias de entrenamiento puede ser díficil, y en caso de conseguirlo, se requieren grandes cantidades de memoria para almacenarla y mucha carga computacional para procesarla. Cuando no se dispone de una fuente ilimitada de datos, se utilizan técnicas como la *validación cruzada* de la que se comentará en apartados siguientes.

Otro de los grandes problemas del entrenamiento es, cuándo hay que parar, para que dichos sistemas no se adapten al conjunto de entrenamiento produciéndose el efecto de sobreentrenamiento y consiguientemente, un mal funcionamiento, cuando al sistema se le presentan muestras que nunca ha visto.

Este capítulo trata de analizar todos estos aspectos: modelos iniciales, criterios de parada, tamaño del conjunto de entrenamiento y su dependencia, número de estados de los modelos, etc. y cómo afectan éstos a la capacidad de generalización.

Este capítulo se ha estructurado en 3 partes. En la primera se hará una intruducción al concepto de capacidad de generalización. Seguidamente, se aborda el problema de la optimización de los HMMs para cada aplicación de reconocimiento de voz, texto manuscrito y firmas. Y por último, se realiza un estudio del cálculo computacional invertido en cada proceso.

4.2 Capacidad de generalización

Un modelo se dice que generaliza bien cuando la relación entrada-salida es calculada correctamente para todos o la gran mayoría de los patrones de entrada-salida nunca vistos por el modelo, en la fase de entrenamiento. La generalización depende fundamentalmente de tres factores:

1. Del número de las muestras de entrenamiento,
2. De la complejidad del modelo,
3. Y por supuesto, del problema en sí.

El último factor es imposible controlarlo, ya que depende de la aplicación en sí. En su evolución, la investigación sobre la capacidad de generalización ha estado enfocada en los dos primeros factores encontrándose dos tendencias [Hush93]:

1. Modificación de la estructura del sistema antes, después o durante la fase de entrenamiento, así están los métodos de poda *pruning* y crecimiento, entrenamiento discriminativo, manteniendo fijo el conjunto de entrenamiento [Sarkar96] etc.
2. La arquitectura del modelo es fija (siempre en función de la complejidad del problema a tratar) y determinar la medida del conjunto de entrenamiento para que el modelo generalice bien.

La capacidad de generalización se puede obtener a partir de los errores cometidos por el modelo cuando se le proporcionan entradas las cuales nunca ha visto anteriormente. También se han utilizado otras medidas de predicción de la capacidad de generalización, haciendo uso de alguna técnica media y varianza de Lacouture [Lacou97].

Ambas tendencias han sido principalmente aplicadas en el entrenamiento de Redes Neuronales, dada su simplicidad e idoneidad para implementar sistemas de clasificación. En este capítulo se detallan los resultados obtenidos al aplicar algunas de estas técnicas de generalización a un clasificador de secuencias temporales, como son los modelos ocultos de Markov.

Se quiere resaltar que el objetivo planteado no fue obtener una tasa de reconocimiento del 100 % por el contrario explorar si el hecho de aplicar técnicas de generalización mejoran o no, la calidad total del sistema.

Como ya se ha tratado en la introducción, la generalización es una medida del óptimo funcionamiento de un clasificador, cuando se aplican entradas que nunca ha visto, una

vez que el entrenamiento ha finalizado. La capacidad de generalización puede ser medida a partir de la tasa de reconocimiento.

En cuanto al número de las muestras de entrenamiento, es sabido que cuantos más datos se le presente al clasificador mejor funcionará y además, debería aprender mejor a dar una respuesta correcta. Si ésto es así, nos encontramos con el problema de la disponibilidad de datos, es decir, no siempre se dispone de una fuente ilimitada de ellos.

En función de la disponibilidad de los datos, el problema de la generalización puede ser tratado desde las dos tendencias comentadas anteriormente: primero, cuántos datos son necesarios para que un modelo determinado funcione correctamente; y segundo, partiendo de un conjunto fijo de datos de entrenamiento, encontrar el modelo que generalice bien.

Tanto para las tareas de reconocimiento de voz, texto manuscrito y firmas, los datos de que se disponen son limitados, por lo que la mayoría de las pruebas están enfocadas desde el segundo punto de vista; partiendo de un conjunto de datos, encontrar aquel modelo que generalice bien. Por supuesto, se ha añadido algunos experimentos para estudiar la sensibilidad de los modelos en función de los datos de entrenamiento.

En un principio parece correcto entrenar el modelo hasta minimizar o alcanzar un determinado valor de la función de coste. Sin embargo, esta técnica no nos proporciona ninguna información acerca de la capacidad de generalización que tiene el modelo. Esta información debe de ser extraída a partir de los datos del conjunto de test y se debe definir algún parámetro que mida la capacidad de generalización e indique cuando el modelo comienza a funcionar peor.

En cuanto a la complejidad del modelo se han realizado una serie de problemas con el fin de obtener una estructura óptima de los modelos. El apartado siguiente trata extensivamente sobre las consideraciones, pruebas y conclusiones acerca de la estructura del modelo.

Sobre la dependencia de la generalización con el problema hay que añadir que es cierto que cada aplicación requiere de un estudio particular, ya que las condiciones iniciales no son las mismas. Sin embargo, a medida que se van estudiando las condiciones óptimas de trabajo de los modelos ocultos de Markov, se observan ciertas características comunes de optimización.

4.3 Estructura de los modelos HMMs

Inicialmente es necesario definir un modelo de estudio al cual se le aplican, de forma sucesiva las distintas modificaciones. Se parte de un modelo general y a medida que se obtienen los resultados, este modelo se va transformando hasta concluir en el modelo final.

Según [Rabi92] un modelo queda definido por el número de estados **N**, los símbolos de observación **M**, la matriz de transiciones **A**, la matriz de probabilidades de los símbolos de observación **B** y la matriz de estados iniciales Π. El modelo recomendado en [Rabi92] para un reconocedor de dígitos aislados es el modelo BAKIS, de izquierda a derecha, limitando el número de saltos.

Este apartado trata de encontrar los dos primeros parámetros: el número de símbolos de observación **M** y el número de estados óptimos **N**.

4.3.1 Símbolos de observación

4.3.1.1 Señal de voz

Un factor importante es definir los símbolos de observación **M**. Para el caso de la señal de voz, el número de símbolos de observación viene determinado por el tamaño de la biblioteca del cuantificador vectorial de tal forma, que los modelos ocultos de Markov den una salida razonablemente buena sin una excesiva carga computacional.

Para ello se probó el funcionamiento de los HMMs con distintas bibliotecas. Los modelos fueron entrenados con 32 realizaciones por palabra y locutor. El test se realizó con 100 realizaciones por palabra y locutor. Los modelos HMMs utilizan la estructura Bakis con salto 2, inicializados de manera aleatoria y 6 estados. Los resultados que aparecen en la tabla 4.1 son los valores promedios de entrenar 10 modelos por palabra para las distintas bibliotecas, de tamaño **M**.

Biblioteca	Loc 1	Loc 2	Loc 3	Loc 4	Loc 5	Media	Varianza
M=8	68.2 %	85.8 %	62.9 %	76.3 %	45.7 %	67.8 %	15.1
M=16	83.5 %	90.1 %	85.4 %	89.4 %	68.5 %	83.4 %	8.7
M=32	91.4 %	91.7 %	93.3 %	89.5 %	89.3 %	91.0 %	1.7
M=64	86.4 %	97.2 %	96.5 %	90.2 %	94.7 %	93 %	4.6
M=128	92.0 %	95.9 %	92.2 %	95.7 %	91.3 %	93.4 %	2.2

Tabla 4.1: Tasa de Reconocimiento para distintas Bibliotecas de tamaño M para la secuencia de test.

En la figura 4.1 se representa los resultados de la tabla 4.1, trazo señalado por **x** para la secuencia de test, junto con la tasa de reconocimiento de la secuencia de entrenamiento (trazo señalado por una **o**). A medida que aumenta el tamaño de la biblioteca se observa que la tasa de reconocimiento aumenta tanto para la secuencia de test como para la de entrenamiento. En la gráfica 4.2 se ha representado la varianza de la media de la tasa

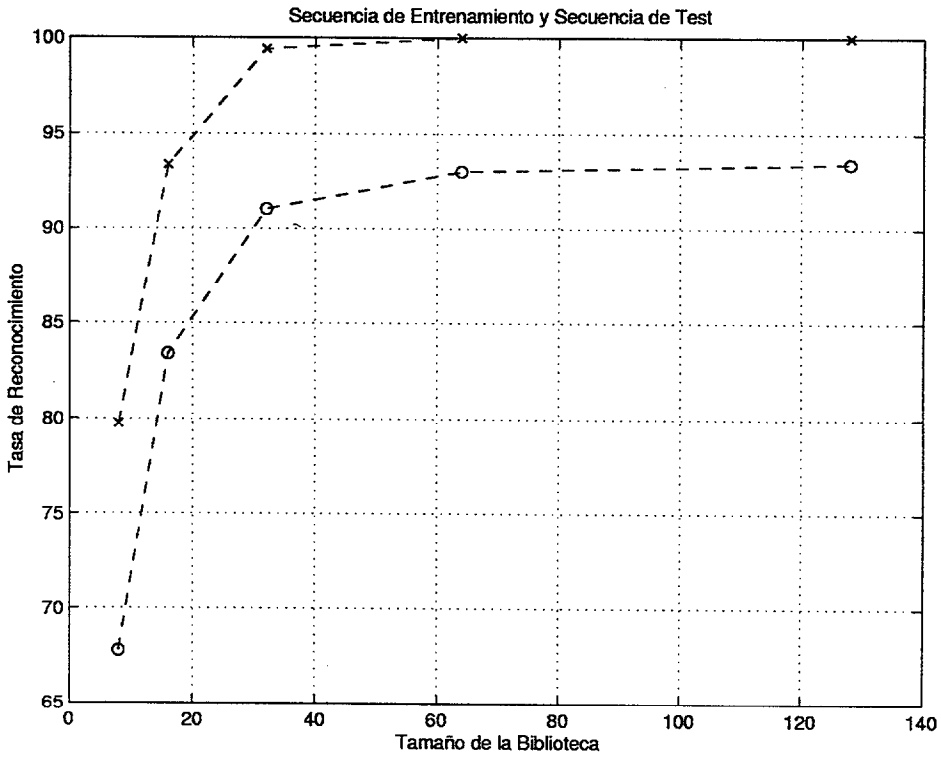

Figura 4.1: Tasa de reconocimiento de voz en función de la biblioteca del Cuantificador Vectorial.

de reconocimiento para todos los modelos entrenados para la secuencia de test (trazo marcado por una **x**) y para la secuencia de entrenamiento (trazo marcado por **o**). Se observa que es la biblioteca de tamaño 32 la que representa la varianza más baja, es decir la tasa de reconocimiento es más uniforme entre los distintos locutores. Para bibliotecas de tamaño 16 y 64 la varianza aumenta, hay más diferencias entre los locutores. Se observa que para la biblioteca de tamaño 64, la tasa de reconocimiento está 2 puntos por encima de la librería de 32. Sin embargo ésta última presenta una varianza más pequeña que la biblioteca anterior (de tamaño 64).

Se observa que la tasa de reconocimiento se estabiliza para bibliotecas de tamaño superior a la de 32. Sin embargo no se percibe el mismo efecto en la varianza, la cual experimenta un salto de tres puntos entre la biblioteca de tamaño 32 y 64. Debido a esto se ha elegido realizar los siguientes experimentos con ambas bibliotecas, hasta obtener claramente el tamaño de la biblioteca.

4.3.1.2 Texto manuscrito y firmas

Para el caso de las señales de texto manuscrito y firmas, es diferente, ya que este tipo de señales se codificaron como secuencias de números obtenidas a partir de la trayectoria

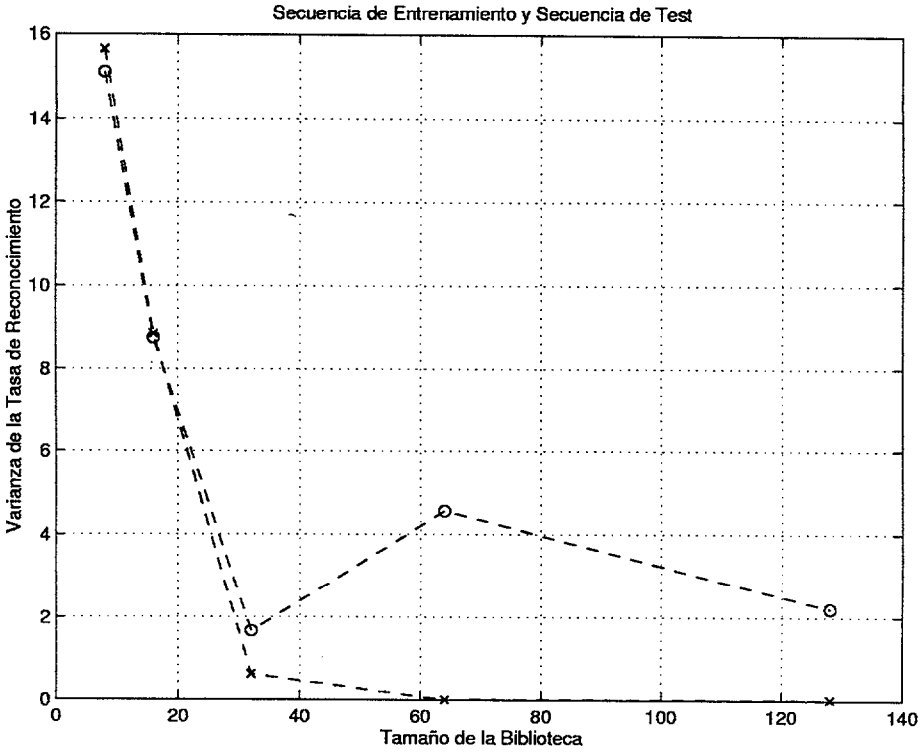

Figura 4.2: Varianza de la tasa de reconocimiento de voz en función de la biblioteca del Cuantificador Vectorial.

de los trazos. En el capítulo 3, se describió este proceso, pero para más información consutar [Ciro99-pfc] y [Camino99-pfc]. Para obtener la dirección de los trazos se aplicó una matriz 3×3, tal como vio en en el capítulo 3, la cual define 8 posibles direcciones que puede tomar el trazo. Cada dígito y firma se conviertió en una secuencia de números que definen la dirección de los trazos. Los vectores sólo tendrán 8 posibles valores. Por tanto el número de símbolos de observación para la aplicación de texto manuscrito y de firmas viene definido por el proceso de codificación y es igual a 8, ver 4.3.

4.3.2 Número de estados

El siguiente paso es la de encontrar el número de estados óptimos.

4.3.2.1 Señal de voz

Se partió de un modelo BAKIS (de izquierda a derecha) con salto 2, inicializados de manera aleatoria. Para ello se entrenó los modelos con el algoritmo clásico de Baum-Welch. Se

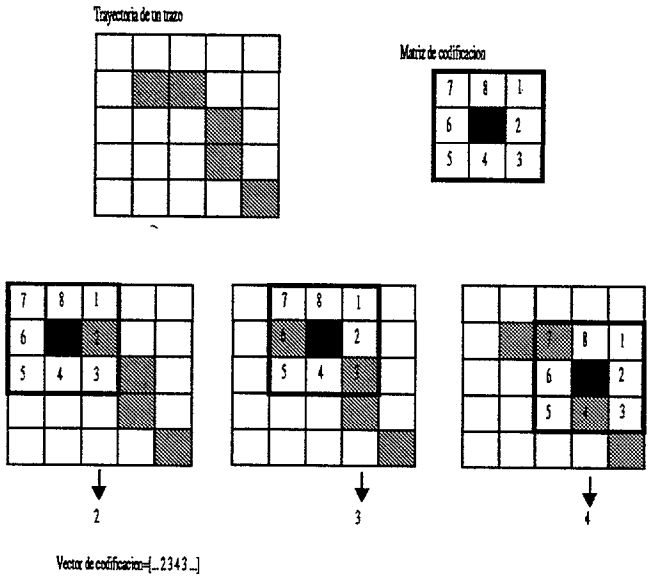

Figura 4.3: Ejemplo de vectorización de una trayectoria de un trazo.

utilizó 32 muestras por dígito para el entrenamiento de los modelos. Para ello se realizaron 10 pruebas con las bibliotecas del cuantificador vectorial de tamaño 32 y 64 vectores para los 5 locutores. Se realizó con las dos bibliotecas para comparar los resultados y elegir de manera óptima el número de estados. Los resultados promedios de las tasas de reconocimiento y varianza vienen dados en las figuras 4.4 y 4.5 respectivamente.

Como se puede observar en la figura 4.4 la tasa de reconocimiento para la biblioteca de 64, permanece casi constante a partir de $N > 3$. Sin embargo para la biblioteca de 32 es más inestable, la tasa de reconocimiento más alta es para $N = 8$.

El comportamiento de la varianza ver figura 4.5, es similar a la media de la tasa de reconocimiento. La biblioteca de 64 presenta a partir de $N > 3$ una misma sintonía. Mientras que la biblioteca de 32 presenta un rango y una varianza superior a la bibliteca de 64 siendo $N = 8$ el que presenta una varianza menor.

De los resultados de aplicar las distintas bibliotecas a modelos inicializados de manera aleatoria y con número de estados, $N = 6$, se extrajo la conclusión que el tamaño idóneo podría estar entre $M = 64$ y $M = 32$ siendo este último, el tamaño de biblioteca que presenta una varianza menor. En cuanto al tamaño del número de estados parece que para el tamaño de $M = 64$ los modelos son insensibles al número de estados a partir de $N > 3$, sin embargo para la biblioteca de $M = 32$ el modelo es más sensitivo al número de estados siendo $N = 8$ el valor de número de estados con varianza más pequeña y tasa de reconocimiento más alta.

Para concluir sobre el adecuado valor de **M** y **N** se realizó una tercera prueba. Se entrenó:

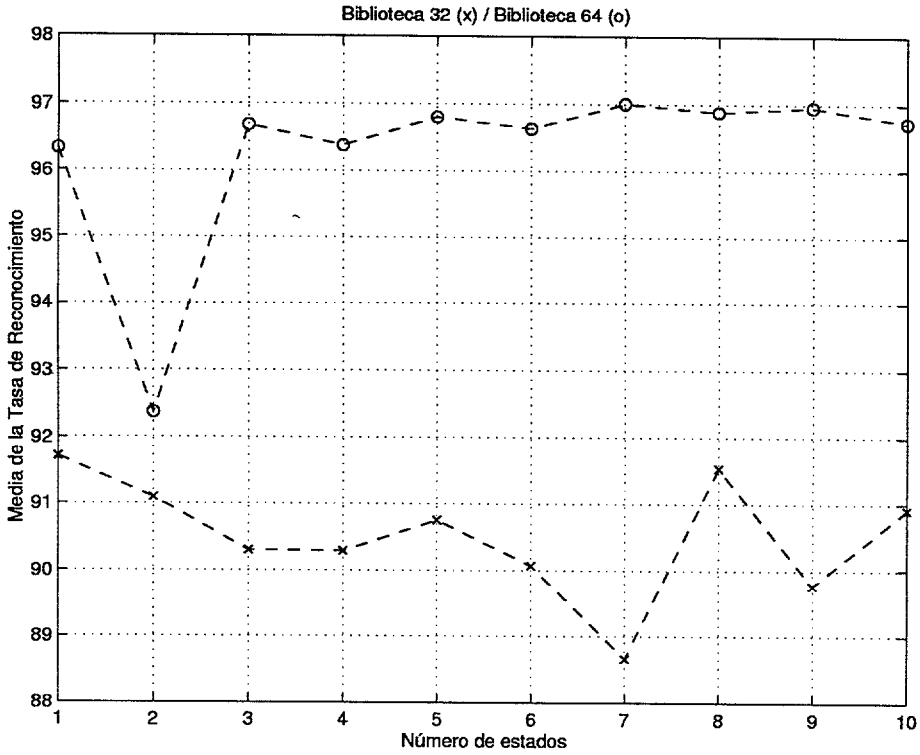

Figura 4.4: Media de la tasa de reconocimento de voz para la secuencia de test en función del número de estados.

- 10 modelos HMMs con $M = 32$ y $N = 6$.
- 10 modelos HMMs con $M = 64$ y $N = 6$.
- 10 modelos HMMs con $M = 32$ y $N = 8$.
- Y 10 modelos HMMs con $M = 64$ y $N = 8$.

Los resultados de esta tercera prueba se han recogido en la tabla 4.2

HMM	Loc 1	Loc 2	Loc 3	Loc 4	Loc 5	Media	Varianza
M=32 y N=6	91.4 %	91.7 %	93.3 %	89.5 %	89.3 %	91.0 %	1.7
M=64 y N=6	86.4 %	97.2 %	96.5 %	90.2 %	94.7 %	93.0 %	7.6
M=32 y N=8	89.2 %	95.9 %	93.1 %	92.5 %	91.7 %	92.5 %	2.4
M=64 y N=8	92.5 %	97.5 %	90.9 %	84.4 %	95.3 %	91.7 %	5.7

Tabla 4.2: Tasa de reconocimiento de voz para distintas bibliotecas de tamaño M y distintos estados N para la secuencia de test.

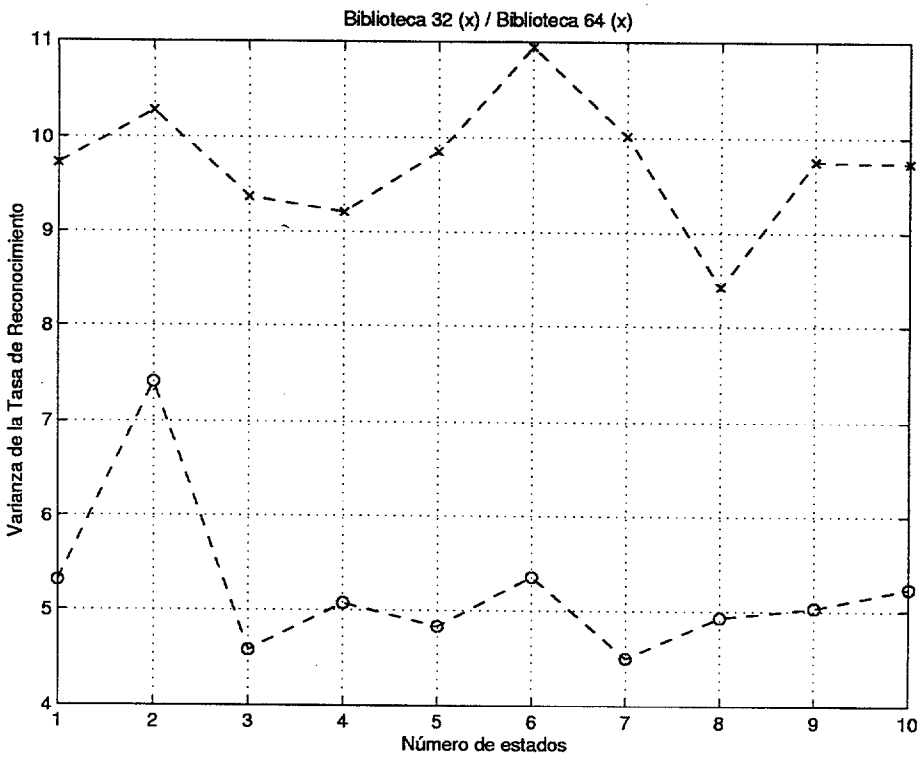

Figura 4.5: Varianza de la tasa de reconocimento para la secuencia de test en función del número de estados.

La tasa de reconocimiento más alta es para la biblioteca de $M = 64$ y número de estados $N = 6$, sin embargo la varianza que presenta estos modelos es la más alta. Es la biblioteca de $M = 32$ y $N = 6$ la que presenta la varianza más pequeña, pero la tasa de reconocimiento está 2 puntos por debajo.

Atendiendo al criterio de homogeneizar el funcionamiento de los modelos se eligió aquellos valores de **M** y **N** que tuvieran una varianza pequeña, aunque no se correspondieran con la tasa de reconocimiento más alta. La estructura final elegida para trabajar como modelos iniciales es:

1. Número de símbolos de observación $M = 32$.
2. Número de estados de cada modelo $N = 6$.

Se ha considerado que se ha de partir de modelos HMMs que sean lo más uniforme posible entre los distintos locutores, aunque la tasa de reconocimiento no sea la más alta.

4.3.2.2 Texto manuscrito

Para la aplicación de texto manuscrito se utilizó modelos HMMs Bakis con salto 2, inicializados de manera aleatoria y entrenados con el algoritmo Baum-Welch. De los 40 escritores que se disponen en la base de datos se utilizaron 20 escritores. De cada escritor se disponían de 12 realizaciones. La distribución de las realizaciones para el entrenamiento y test es como sigue, 8 para entrenamiento y 4 para test. Como resultado, cada modelo se entrenó con 160 muestras y se procedió a realizar el test con 80 realizaciones.

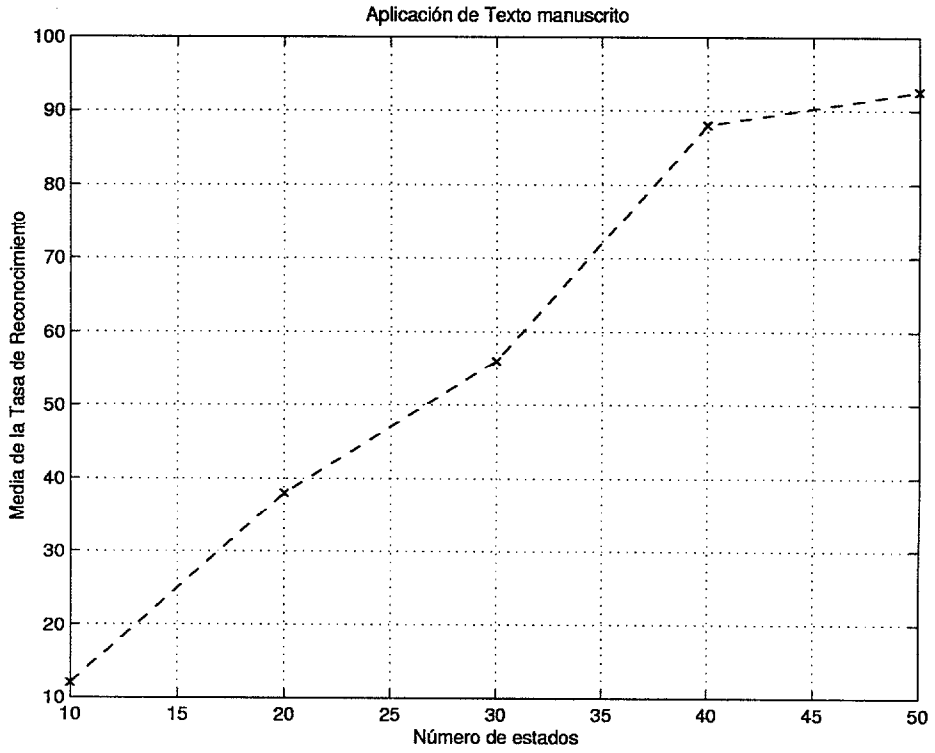

Figura 4.6: Media de la tasa de reconocimento de firmas para la secuencia de test en función del número de estados.

La figura 4.6 presenta la tasa de reconocimiento frente al número de estados. Las pruebas se realizaron para 10, 20, 30, 40 y 50 estados. Los modelos de 30 y 40 estados presenta una tasa de reconocimiento relativamente baja por lo que se ha elegido los modelos de 50 estados.

4.3.2.3 Firmas

Para la aplicación de firmas al igual que en las aplicaciones anteriores se utilizó modelos HMM Bakis con salto 2, inicializados de manera aleatoria y entrenados con el algoritmo

Baum-Welch. De los 60 escritores que se disponen en la base de datos se utilizaron 10 escritores. De cada escritor se disponían de 24 realizaciones. La distribución de las realizaciones para el entrenamiento y test es como sigue, 14 para entrenamiento y 10 para test.

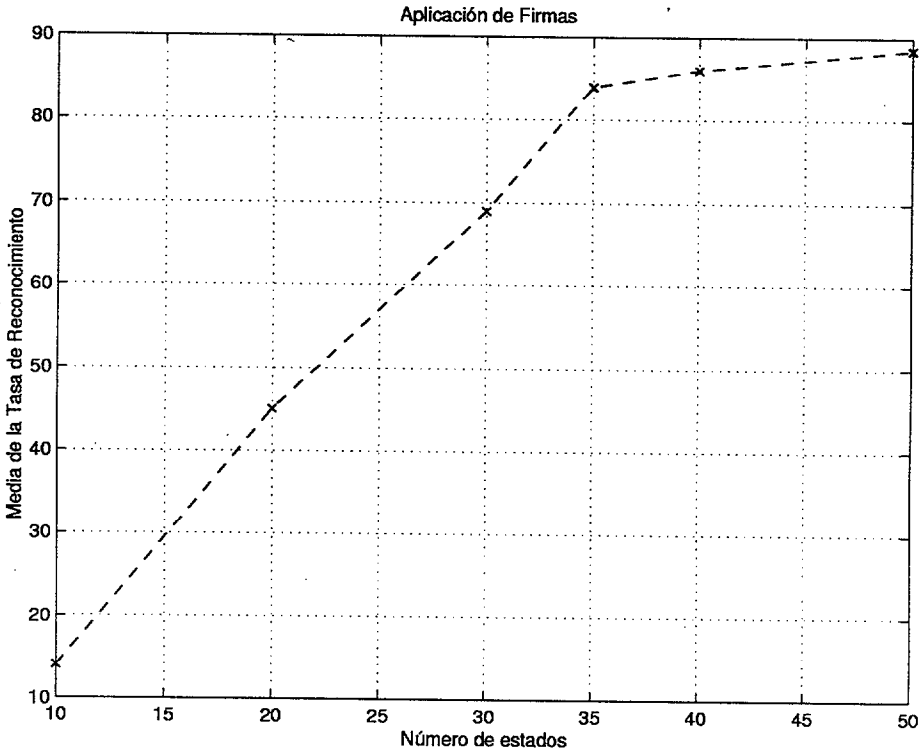

Figura 4.7: Media de la tasa de reconocimento de firmas para la secuencia de test en función del número de estados.

La figura 4.7 presenta la tasa de reconocimiento frente al número de estados. Las primeras pruebas se realizaron para 10, 20, 30, 40 y 50 estados. Se experimentó con modelos de 35 estados para observar el comportamiento, y se observó una mejoría frente al de 30 estados y no muy por debajo de los modelos con número de estados superiores. Con el fin de no trabajar con modelos con un número de estados elevado, se eligió 35 estados para la aplicación de firmas.

4.4 Inicialización de los modelos HMM

Con las variables seleccionadas del modelo, un aspecto importante es cómo inicializar los modelos y cómo afecta a la capacidad de generalización.

4.4.1 Métodos de inicialización

Fijada ya una estructura, el siguiente paso nos lleva al entrenamiento mediante un algoritmo de aprendizaje. Un factor clave para comenzar dicho entrenamiento es cómo se inicializan los parámetros del modelo. En este apartado se realizará una breve descripción de los métodos de inicialización conocidos, así como una pequeña valoración de cada uno de ellos y por último, introduciremos un nuevo método de inicialización.

El método clásico para iniciar el entrenamiento de un HMM es el descrito por Rabiner en [Rabi92], para reconocimiento de voz, que consiste en:

1. Generación de los modelos iniciales de forma aleatoria, las matrices **A** de transición de estados, **B** la matriz de probabilidad de emisión de símbolos y la matriz Π o matriz de estado inicial.
2. Aplicar segmentación Kmedias.
3. Iterar con el algoritmo de Baum-Welch tantas veces como sean necesarias hasta su convergencia. La experiencia nos indica que de 4 a 8 iteraciones son suficientes para la convergencia.

Este método es muy eficiente y rápido, pero su capacidad de generalización es muy dependiente del modelo inicial aleatorio, es decir de la semilla inicial del generador de números aleatorios.

En algunos casos el modelo inicializado con este método hace que el algoritmo Baum Welch no converja.

Tratando de evitar esta aleatoriedad en la capacidad de generalización, [Sarkar96] propone diversos métodos. Entre ellos citamos los siguientes:

- *Reentrenamiento* [Makhoul85]. Una práctica habitual es entrenar varios modelos HMMs con diferentes semillas y utilizar aquel que presente una mayor capacidad de generalización. El incoveniente de este sistema es el aumento de carga computacional en la fase de entrenamiento.
- *Voting-model.* Este método entrena varios HMMs con diferentes semillas y a diferencia del caso anterior, los utiliza todos en la fase de reconocimiento: el patrón reconocido es el más votado [Jain97]. Este método aumenta la carga computacional tanto en la fase de entrenamiento como en la fase de test.
- *Probabilidad de transición equiprobable.* Este método inicializa los modelos HMMs con la matriz de probabilidad de transición de estados matriz **A** y Π con todos sus

valores iguales, es decir existe la misma probabilidad de saltar a un estado que a otro. Opera de igual forma con las probabilidades de emisión de símbolos, matriz **B**. En este caso el modelo inicial es único.

- *Modelo inicial preentrenado* [Rabi93]. En este caso se inicia el algoritmo de entrenamiento con unos valores de las matrices **A**, **B** y Π apropiado al modelo y precalculados anteriormente. De este modo se facilita enormemente la labor del algoritmo de entrenamiento y éste converge rápidamente entre 2 ó 3 iteraciones. Su inconveniencia es la fuerte dependencia de los resultados obtenidos con el modelo inicial, sin considerar el hecho de que hay que disponer de un modelo inicial apropiado al modelo y a los datos.

Veamos a continuación los resultados de estos métodos de inicialización.

4.4.1.1 Señal de voz

Fijada la estructura de los modelos HMMs para la señal de voz, **N=6** y **M=32**, los resultados obtenidos, en términos de porcentajes de reconocimiento con las diferentes inicializaciones aparecen en la tabla 4.3. Los resultados para los métodos de *Reentrenamiento* y *Voting Model* fueron obtenidos para 10 modelos por palabra.

Método	Loc 1	Loc 2	Loc 3	Loc 4	Loc 5	Media	Varianza
Aleatorio	91.4 %	91.7 %	93.3 %	89.5 %	89.3 %	91.0 %	1.7
Reentrenamiento	92.0 %	95.8 %	94.4 %	94.5 %	94.3 %	94.2 %	1.4
Voting Model	95.1 %	97.4 %	95.6 %	95.9 %	95.7 %	95.4 %	0.86
Equiprobables	91.4 %	88.2 %	93.9 %	93.0 %	89.5 %	91.2 %	2.4

Tabla 4.3: Comparación de los resultados obtenidos con un reconocedor de voz de la inicialización con diferentes métodos.

No se han realizado pruebas con el método de inicialización de modelos preentrenados, pues el resultado es muy dependiente del modelo inicial.

A la vista de los resultados de la tabla 4.3 se observa que los mejores métodos son: *Voting Model* y *Reentrenamiento* con las tasas de reconocimiento más altas y varianzas más pequeñas. Son razonables los resultados obtenidos, ya que para el caso del método *Voting model* la clase elegida es la más votada por varios modelos, por lo que, errores de modelos individuales pueden corregirse por los aciertos de la mayoría. En cuanto al método de *Reentrenamiento* hay que entrenar n modelos para elegir el mejor de ellos.

El incoveniente que presentan estos dos métodos es la carga computacional, la cual es elevada al tener que entrenar varios modelos, es decir si se entrenan n modelos, la carga

computacional será n veces superior a cualquiera de los dos métodos restantes. Y en el caso de *Voting Model* se ve además incrementada durante la fase de test, ya que todos los modelos participan durante esta fase.

En el capítulo siguiente dedicaremos una especial atención al método de *Voting Model*, como mecanismo de combinación de varios modelos para dar una respuesta.

Una vez descritos los métodos más utilizados de inicialización de los modelos HMMs pasamos a proponer un método de inicialización de HMMs a la que se ha denominado método de *inicialización de igual ocupación de estados, (IOE).*

4.4.2 Inicialización de igual ocupación de estados

Dentro de la búsqueda del modelo inicial que permita una mayor capacidad de generalización tras el entrenamiento, se propone un nuevo método, [Itzi97], que consiste en una modificación del método clásico de Rabiner, ver la tabla 4.4. Simplemente se basa en introducir un punto de verificación que permita comprobar si el modelo cumple los requisitos de este método. En la tabla 4.5 aparecen los pasos de este método de inicialización.

Paso 1: Generación del modelo inicial aleatorio
Paso 2: Verificar que el modelo inicial es apropiado
Paso 3: Si el modelo inicial no es el apropiado volver al paso 1
Paso 4: Segmentación de las Kmedias
Paso 5: Algoritmo de Baum-Welch

Tabla 4.4: Algoritmo de inicialización.

La noción básica propuesta para verificar que el modelo generado aleatoriamente sea el apropiado o no, es que el número de veces que cada estado es visitado sea similar. Esto es, se da por válido el modelo inicial cuyo número de visitas a cada estado presenta una distribución cercana a la uniforme. A este método lo denominaremos de **igual ocupación de estados**.

El número de veces que cada estado es visitado puede ser facilmente estimado mediante el algoritmo de Viterbi o con el máximo de la matriz γ, ver [Rabi93].

Así pues, el algoritmo de entrenamiento con inicialización de igual ocupación de estados aparece en la tabla 4.5.

Como condición de *razonablemente uniforme* se ha utilizado lo siguiente: el número de

Paso 1: Generación aleatoria del modelo inicial
Paso 2: Estimación del número de veces que cada estado es ocupado
Paso 3: Si la distribución número visitas/estado es
razonablemente uniforme, paso 4, sino ir al primer punto
Paso 4: Segmentación de las Kmedias
Paso 5: Algoritmo de Baum-Welch

Tabla 4.5: Algoritmo de inicialización Igual Ocupación de Estados.

veces que cada estado es visitado debe de mantenerse dentro del rango $[\mu - 0.3\mu \quad \mu + 0.3\mu]$, donde μ es la medida del número de veces que cada estado es visitado. Evidentemente esta condición puede relajarse o endurecerse.

Un posible problema es que la condición del punto 3 nunca se satisfaga, entrando en un bucle infinito. En nuestra experiencia y con la condición antes mencionada nunca se nos ha dado el caso, cumpliéndose la condición de 3 a 5 iteraciones. Evidentemente si se exige como condición una uniformidad estricta, puede darse el caso de un bucle infinito.

4.4.2.1 Señal de voz

A continuación se presenta los resultados de este método de inicialización junto a una evaluación comparativa con los anteriores. Estos resultados puede encontrarse en la tabla 4.6.

Método	Loc 1	Loc 2	Loc 3	Loc 4	Loc 5	Media	Varianza
Viterbi	96.9 %	97.1 %	97.3 %	96.9 %	92.1 %	96.0 %	2.2
Gamma	93.4 %	96.2 %	94.5 %	94.8 %	88.4 %	93.5 %	3.0

Tabla 4.6: Resultados obtenidos con el método de inicialización Igual Ocupación de Estados (IOE).

En ella se tienen los resultados utilizando como método de estimación de secuencia de estados el algoritmo de Viterbi y el máximo de la matriz γ [Rabi93]. Evidentemente el algoritmo de Viterbi provee de un modelo inicial que da lugar a un reconocedor con una mejora de la capacidad de generalización, tanto en media como en varianza. De aquí en adelante, siempre que se mencione el método de igual ocupación, se supone con el algoritmo de Viterbi.

En cuanto a la comparación entre los métodos de inicialización se encuentra en la tabla 4.7. En ella se comparan los resultados entre la inicialización aleatoria o clásica, reentrenamiento, *voting model*, equiprobables, y de igual ocupación. Los resultados que aparecen en esta tabla son el promedio de entrenar 10 modelos HMMs para cada método de inicialización.

Método	Loc 1	Loc 2	Loc 3	Loc 4	Loc 5	Media	Varianza
Aleatorio	94.6 %	93.2 %	96.9 %	91.4 %	91.8 %	93.6 %	2.3
Reentrenamiento	95.6 %	97.0 %	98.4 %	97.0 %	95.8 %	96.7 %	1.1
Voting Model	97.3 %	97.9 %	98.5 %	97.5 %	95.9 %	97.4 %	0.96
Equiprobables	95.5 %	87.7 %	97.4 %	95.8 %	92.6 %	93.8 %	3.8
Igual Ocupación Estados	96.9 %	97.1 %	97.3 %	96.9 %	92.1 %	96.0 %	2.2

Tabla 4.7: Comparación de los métodos de inicialización en el reconocimiento de voz.

Analizando los resultados de la tabla 4.7 podemos ver que en media, el método de inicialización con igual ocupación de estados presenta una mejora de 2.5 puntos frente a los modelos inicializados de manera aleatoria. La varianza es similar a la de los modelos inicializados de manera aleatoria. Por supuesto los métodos de *Reentrenamiento* y *Voting Model* son los que presentan mejores resultados pero a costa de un coste computacional.

Comparando los resultados de la tabla 4.7, los cuales fueron obtenidos aplicando el algoritmo de *Viterbi* con los de la tabla 4.3 obtenidos a partir del máximo de la matriz γ hay una cierta mejoría en los datos presentados en la tabla 4.7. Por lo que a partir de ahora, todas las pruebas que se realicen, se utilizará el método de *Viterbi.*

Los resultados obtenidos en la tabla 4.7 son los valores promedios de entrenar 10 modelos por dígito, salvo en el método de *reentrenamiento* que de entre los 10 modelos, se elige el mejor.

4.4.2.2 Texto manuscrito

En esta sección se recoge los resultados de aplicar los diferentes métodos de inicialización aplicados al reconocimiento de texto manuscrito.

Observando la tabla 4.8, el mejor método al igual que en la aplicación de voz es el de reentrenamiento sin embargo el método de igual ocupación de estados presenta resultados buenos, tanto en media de la tasa de reconocimiento como en varianza.

Método	Media	Varianza
Aleatoria	92.5 %	7.9
Equiprobable	86.7 %	16.6
Reentrenamiento	98.1 %	1.6
Igual Ocupación Estados	96.6 %	2.83

Tabla 4.8: Comparación de los métodos de inicialización para la aplicación de texto manuscrito

4.4.2.3 Firmas

En esta sección se recoge los resultados de aplicar los diferentes métodos de inicialización aplicados al reconocimiento de firmas.

Método	Media	Varianza
Aleatoria	84 %	13.5
Equiprobable	79%	19.7
Igual Ocupación Estados	92 %	6.3

Tabla 4.9: Comparación de los métodos de inicialización para la aplicación de firmas.

Se comprueba nuevamente la mejora introducida en los modelos aplicando la inicialización de *igual ocupación de estados*

4.5 Criterios de parada

Cuando un HMM se entrena, las matrices **A**, **B** y Π son modificadas con el fin de incrementar la probabilidad de los patrones de entrenamiento. El error de la secuencia de test y de entrenamiento tiende a decrementar con el tiempo, pero en algún punto, el error originado por la secuencia de test alcanza un mínimo y comienza a incrementar. Esto es originado porque a medida, que se intenta minimizar el error durante la fase de entrenamiento, los modelos HMMs tienden a adaptarse al conjunto o secuencia de entrenamiento. Un procedimiento para evitar el sobreentrenamiento u *overfitting* es mediante la estimación de la capacidad de generalización y parar cuando comienza a decrementar. El problema es cómo estimar la capacidad de generalización para parar el entrenamiento. Varios métodos de parada han sido propuestos:

1. *Número fijo de iteraciones.* Un método muy frecuente utilizado para entrenar los modelos HMMs es fijar el número de iteraciones. Para la aplicación de reconoci-

miento de voz suele ser de 2 a 3 iteraciones para evitar el *overfitting*

2. *Fijar un umbral.* Otro método utilizado para parar el entrenamiento es utilizar una medida relativa, obtenida a partir de la probabilidad acumulada sobre el conjunto de entrenamiento. Esta medida es comparada con un umbral denominada δ y si está por debajo se para el entrenamiento si no, se continua. Normalmente este umbral suele tener un valor aproximado de 0.01 con el fin de evitar muchas iteraciones y como consecuencia el *overfitting.*

3. *Media-Varianza.* La táctica habitual en el entrenamiento es minimizar la suma de errores cuadráticos (SE) para todas las muestras de la secuencia de entrenamiento, o de validación en caso de trabajar con validación cruzada. Pero si sólo se utiliza este criterio no se consigue la generalización del modelo. Para aliviar este incoveniente, Lacouture en [Lacou97] sugiere que si se impone un comportamiento regular para todas las muestras se conseguirá una generalización mejor. Concretamente propone medir tal comportamiento como la varianza del error cuadrático a lo largo de las muestras. Así en el entrenamiento se obtiene mejor generalización si se minimiza la media del error cuadrático más λ veces su varianza. En [Kung93] [Utans91] propone algunos estimadores de la capacidad de generalización tales como

$$GA = \lambda \cdot Mean(SE) + (1 - \lambda) \cdot Var(SE) \tag{4.1}$$

Donde GA es la estimación de la capacidad de generalización, $Mean(SE)$ y $Var(SE)$ son la media y la varianza de las probabilidades del conjunto de entrenamiento y λ es una constante. Generalmente el valor de λ, se calcula a partir del número de parámetros y del tamaño de la secuencia de entrenamiento [Utans91]. Como aportación al cálculo de λ, en el estudio empírico realizado sobre el valor óptimo λ se ha comprobado que uno de los mejores valores es aquel que da tanta relevancia a una variación de la media del valor (SE) como a una variación de la varianza del valor de (SE). El procedimiento propuesto para conseguir esto ha sido, en la primera iteración, tras obtener la medida de la media y la varianza del valor de SE. Se calcula a partir de $\lambda \cdot Mean(SE) = (1 - \lambda) \cdot Var(SE)$ obteniendo el valor de λ por la ecuación 4.2

$$\lambda = \frac{Var(SE)_1}{[Mean(SE)_1 + Var(SE)_1)]} \tag{4.2}$$

donde $Mean(SE)_1$ y $Var(SE)_1$ son la media y la varianza de la probabilidad del conjunto de entrenamiento (validación) después de la primera iteración. En la práctica, el valor obtenido de λ es muy cercano a los obtenidos con el método clásico [Utans91].

Veamos a continuación los resultados de aplicar estos criterios de parada para las distintas tareas de reconocimiento.

4.5.1 Experimentos y resultados

4.5.1.1 Señal de voz

En la tabla 4.10 se recoge el estudio comparativo acerca de la capacidad de generalización medida como tasa de reconocimiento utilizando los distintos criterios de parada aplicadas al sistema de clasificación HMM trabajando como reconocedor de palabras aisladas orientado a locutor. La base de datos utilizada es la misma que la comentada en apartados anteriores, 32 realizaciones para el entrenamiento y 100 realizaciones para test por locutor. El método de inicialización de los modelos es aleatorio.

Los criterios comparados son: número fijo de iteraciones, umbral fijo, y media-varianza. En todos los casos se utilizó una inicialización aleatoria. Como se puede ver los mejores resultados se obtuvieron con el criterio de un umbral fijo.

Método	Loc 1	Loc 2	Loc 3	Loc 4	Loc 5	Media	Varianza
Umbral	95.7 %	95.5 %	96.0 %	95.5 %	91.7 %	94.9 %	1.8
Iteraciones	94.6 %	93.2 %	96.9 %	91.4 %	91.8 %	93.6 %	2.3
Media-Varianza	96.2 %	95.9 %	92.0 %	95.3 %	91.5 %	94.2 %	2.2

Tabla 4.10: Tasa de reconocimiento de voz aplicando diferentes criterios de parada (los HMM están inicializados aleatoriamente).

En la tabla 4.11 se comparan los resultados cuando se utilizan los modelos HMM inicializados con el método IOE. Se obtienen mejores resultados para el criterio de parada por umbral.

Método	Loc 1	Loc 2	Loc 3	Loc 4	Loc 5	Media	Varianza
Umbral	96.9 %	97.1 %	97.3 %	96.9 %	92.1 %	96.0 %	2.2
Iteraciones	93.6 %	96.4 %	96.3 %	96.7 %	90.8 %	94.7 %	2.5
Media-Varianza	96.9 %	96.6 %	96.8 %	96.7 %	91.9 %	95.8 %	2.2

Tabla 4.11: Tasa de reconocimiento de voz aplicando diferentes criterios de parada, (los HMMs están inicialización con el método igual ocupación de estados (IOE)).

4.5.2 Texto manuscrito

En la tabla 4.12 se recoge el estudio comparativo de los diferentes criterios de parada aplicados a los HMMs cuando se trabaja con secuencias de observación pertenecientes a texto manuscrito. La base de datos es la misma que la utilizada en pruebas anteriores, 20 escritores con 160 muestras para entrenamiento y 80 para test. El método de inicialización de los HMMs es aleatorio.

Método	Media	Varianza
Umbral	92.5 %	7.9
Iteraciones	90.7 %	9.1
Media-Varianza	93.3 %	6.1

Tabla 4.12: Tasa de reconocimiento de texto manuscrito aplicando diferentes criterios de parada, (HMMs están inicializados aleatoriamente).

En la tabla 4.13 aparece las tasas de reconocimiento, para los distintos criterios de parada, pero utilizando la inicialización *igual ocupación de estados.*

Método	Media	Varianza
Umbral	96.6 %	2.8
Iteraciones	92.1 %	7.7
Media-Varianza	94.4 %	2.2

Tabla 4.13: Tasa de reconocimiento de texto manuscrito aplicando diferentes criterios de parada, (HMMs están inicializados con el método IOE).

Se observa que, el método de la media-varianza se adapta perfectamente al problema, aunque la mejor tasa de reconocimiento es para el método de parada umbral. El criterio de iteraciones presenta resultados buenos respecto a los otros dos métodos. Esto se debe a que el número de iteraciones fijadas se adapta perfectamente al problema. Veremos que para el reconocimiento de firmas el método de iteraciones no es acertado porque no se eligió adecuadamente el número de iteraciones. El inconveniente quizás de este método está en la elección de las iteraciones de entrenamiento de los HMMs.

4.5.3 Firmas

En la tabla 4.14 se recoge el estudio comparativo de los diferentes criterios de parada aplicados a los HMMs cuando se trabaja con secuencias de observación pertenecientes a

firmas. La base de datos es la misma que la utilizada en pruebas anteriores, 10 escritores con 14 muestras para entrenamiento y 10 para test por escritor. El método de inicialización de los HMMs es aleatorio.

Método	Media	Varianza
Umbral	84 %	13.5
Iteraciones	57 %	27.5
Media-Varianza	86 %	12.6

Tabla 4.14: Tasa de reconocimiento de firmas aplicando diferentes criterios de parada, (HMMs están inicializados aleatoriamente).

En la tabla 4.15 aparece las tasas de reconocimiento, para los distintos criterios de parada, pero utilizando la inicialización *igual ocupación de estados*

Método	Media	Varianza
Umbral	92 %	6.3
Iteraciones	77 %	12.5
Media-Varianza	87 %	11.7

Tabla 4.15: Tasa de reconocimiento de texto manuscrito aplicando diferentes criterios de parada, (HMMs están inicializados con el método IOE.

Observamos que el mejor método para finalizar el entrenamiento de los HMMs es fijando un umbral. La diferencia de los resultados entre las tablas 4.14 y 4.15 está en la inicialización de los modelos, se comprueba que la tasa de reconocimiento de la tabla 4.15 presenta una mejoría respecto a la inicialización aleatoria.

4.6 Dependencia con los datos de entrenamiento

Cuando no se disponen de datos suficientes, sino de un conjunto limitado de datos de tamaño L, se utilizará un subconjunto $L_{train} \in L$ para construir el modelo y otro $L_{test} \in L$ para obtener información del funcionamiento del modelo. Una posible información de lo bien o mal que funciona el sistema es contabilizar los errores que comete el sistema.

Sea $d(x)$, un clasificador a construir, $X(\cdot)$ una función que dé a la salida un 1 cuando es verdadero y un 0 cuando es falso, j_n la clase a la que pertence la entrada x_n. Una medida de los errores que comete el clasificador estaría expresada en la ecuación 4.3

$$R(d) = \frac{1}{N}\sum_{n=1}^{N} X(d(x_n) \neq j_n) \tag{4.3}$$

Siendo N el tamaño del conjunto de datos sobre los que se está realizando la medida.

Hay tres formas de determinar $R(d)$ de la ecuación 4.3:

1. El conjunto de entrenamiento sea el mismo que el de test, $L = L_{train} = L_{test}$. Se trata de una técnica poco segura, poco fiable pero sin embargo muy utilizada en determinados casos.

2. En este caso el conjunto de datos L se divide en dos subconjuntos, L_{train} y L_{test}. El primero se utiliza para construir el clasificador y el segundo para estimar $R(d)$. El procedimiento más común para obtener L_{test} es elegiendo aleatoriamente los datos de L. Suele elegirse $\frac{2}{3}$ de L para construir el clasificador y $\frac{2}{3}L$ para test, aunque no hay una justificación teórica de este reparto. Cuando el conjunto L es grande la medida de $R(d)$ suele ser bastante efectiva y honesta. La ecuación 4.3 quedaría para este método de la siguiente manera:

$$R(d) = \frac{1}{N_{test}} \sum_{(x_n, j_n) \in L_{test}} X(d(x_n) \neq j_n) \tag{4.4}$$

3. Para conjuntos de muestras pequeños se utiliza otro método denominado de validación cruzada. Este método consiste en descomponer el conjunto L en v subconjuntos de igual tamaño cada uno de ellos L_1,L_2, ...,L_v. En este caso para construir el clasificador se utiliza $L - L_v$ subconjuntos y para medir $R(d)$ se utiliza el conjunto L_v.

$$R(d) = \frac{1}{N_v} \sum_{(x_n, j_n) \in L_v} X(d(x_n) \neq j_n) \tag{4.5}$$

   Y de forma iterativa se va cambiando el conjunto L_v

De los tres métodos expuestos anteriormente se ha utilizado el método del punto 2 en las distintas pruebas y experimentos. Se divide el conjunto L en dos subconjuntos, L_{train} de tamaño 32 repeticiones por locutor y palabra y L_{test} formada por 100 repeticiones por locutor y palabra. Se ha de considerar que el conjunto de datos disponibles es limitado por lo que es interesante estudiar la dependencia de la capacidad de generalización con los datos de entrenamiento.

Cuando la disponibilidad de los datos es limitada, de los tres métodos propuestos, el más antiguo y popular que se emplea con el fin de obtener una mejor generalización

durante el aprendizaje es el tercer método, el denominado validación cruzada *(cross-validation)*.

Se han realizado algunas pruebas aplicando la validación cruzada para estudiar la sensibilidad de los modelos ante la disponibilidad de los datos.

Un inconveniente que presenta el utilizar validación cruzada en el entrenamiento es que al dividir el conjunto de muestras en L_{train} y L_v puede ocurrir que la secuencia de entrenamiento no disponga de muestras suficientes de una determinada clase pudiéndo resultar un empeoramiento de la capacidad de generalización. Para evitar este incoveniente sin renunciar a las ventajas de generalización del entrenamiento se propone una modificación al método propuesto en el punto 3, al que se ha denominado **validación cruzada aleatoria.**

4.6.1 Validación cruzada aleatoria

Este método es una modificación del método general de validación cruzada. Y consiste en:

En cada iteración se eligen de manera aleatoria las muestras que constituirán las muestras de entrenamiento y validación. Si la secuencia de entrenamiento tiene longitud L y está formada por la secuencia de observación $O = O_1, O_2, ..., O_i, ...O_L$, en cada iteración se elegirán de forma aleatoria los subíndices de la secuencia cumpliéndose que $L_{train} = \frac{2}{3} \cdot L$ y $L_v = \frac{1}{3} \cdot L$. Este proceso se repite en cada iteración. Con esta modificación, a la cual nos referimos como validación cruzada aleatoria, se han conseguido en media resultados similares que con la validación cruzada convencional con carga computacional significativamente menor.

4.6.2 Resultados y experimentos

4.6.2.1 Señal de voz

En la tabla 4.16 aparecen los resultados de entrenar los modelos ocultos de Markov con validación cruzada aplicando inicialización aleatoria y la de igual ocupación de estados y criterios de parada por umbral y por iteraciones. Los mejores resultados aparecen para inicialización con igual ocupación de estados y con criterio de parada por umbral.

En la tabla 4.17 aparecen los resultados de entrenar los modelos ocultos de Markov con validación cruzada alaeatoria aplicando inicialización aleatoria y la de igual ocupación de estados para criterios de parada por umbral y por iteraciones. Los mejores resultados

Método	Loc 1	Loc 2	Loc 3	Loc 4	Loc 5	Media	Varianza
A+VC+Umbral	92.5 %	94.8 %	94.9 %	92.7 %	90.3 %	93.0 %	1.9
IOE+VC+Umbral	95.4 %	96.4 %	97.1 %	95.8 %	91.5 %	95.2 %	2.2
IOE+VC+MaxIter	95.5 %	96.0 %	97.2 %	91.9 %	90.2 %	94.2 %	3.0

Tabla 4.16: Tasa de reconocimiento de voz con validación cruzada (VC) aplicando dos inicializaciones: aleatoria (A) e igual ocupación de estados (IOE); y criterios de parada: umbral y por máximo de iteraciones igual a 4.

Método	Loc 1	Loc 2	Loc 3	Loc 4	Loc 5	Media	Varianza
A+VCA+Umbral	94.1 %	96.0 %	96.1 %	93.4 %	90.4 %	94.0 %	2.3
IOE+VCA+Umbral	95.8 %	95.8 %	96.4 %	95.8 %	93.2 %	95.4 %	1.3
IOE+VCA+MaxIter	96.1 %	97.0 %	97.1 %	96.0 %	91.7 %	95.6 %	2.2

Tabla 4.17: Tasa de reconocimiento de voz con validación cruzada aleatoria (VCA) aplicando dos inicializaciones: aleatoria (A) e igual ocupación de estados,(IOE) ; y diferentes criterios de parada: umbral y por máximo de iteraciones igual a 4.

aparecen para inicialización con igual ocupación de estados y con criterio de parada por umbral.

En la tabla 4.18 se comparan los experimentos con modelos HMM inicializados con igual ocupación de estados para validación cruzada y validación cruzada aleatoria con criterio de para umbral. De entre los dos métodos de validación se obtienen mejores resultados con el de la validación cruzada aleatoria.

Método	Loc 1	Loc 2	Loc 3	Loc 4	Loc 5	Media	Varianza
IOE+Umbral	96.9 %	97.1 %	97.3 %	96.9 %	92.1 %	96.0 %	2.2
IOE+VC+Umbral	95.4 %	96.4 %	97.1 %	95.8 %	91.5 %	95.2 %	2.2
IOE+VCA+Umbral	95.8 %	95.8 %	96.4 %	95.8 %	93.2 %	95.4 %	1.3

Tabla 4.18: Comparación de las tasas de reconocimiento de voz con el método de inicialización, IOE, para entrenamientos que utilizan validación cruzada y validación cruzada aleatoria.

4.6.2.2 Texto manuscrito

En la tabla 4.19 aparecen los resultados de entrenar los modelos ocultos de Markov con validación cruzada. Se realizaron dos pruebas dependiendo de la inicialización: aleatoria y la de igual ocupación de estados. El criterio de parada utilizado es el método umbral.

Método	Media	Varianza
A+VC+Umbral	85.9 %	7.13
IOE+VC+Umbral	89.1 %	6.36

Tabla 4.19: Tasa de reconocimiento de texto manuscrito aplicando validación cruzada (VC), con inicializaciones: aleatorio (A) e igual ocupación de estados (IOE); y con criterios de parada por umbral.

En la tabla 4.20 se aplica la modificación de validación cruzada aleatoria y se compara nuevamente los métodos de inicialización y criterio de parada por umbral.

Método	Media	Varianza
A+VCA+Umbral	88.9 %	7.9
IOE+VCA+Umbral	91.1 %	5.6

Tabla 4.20: Tasa de reconocimiento de texto manuscrito aplicando validación cruzada aleatoria (VCA), con inicializaciones: aleatorio (A) e igual ocupación de estados (IOE); y con criterios de parada por umbral.

Por último se compara los resultados de validación cruzada y validación cruzada aleatoria para la inicialización de igual ocupación de estados en la tabla 4.21. Se comprueba que los mejores resultados se obtienen para el primer método, donde se utilizan todos los datos del conjunto de entrenamiento. Entre los métodos de validación cruzada y la validación cruzada aleatoria, es esta última la que tiene la tasa más alta, ésto se debe a que en el método de validación cruzada aleatoria, los datos del conjunto de entrenamiento se van modificando en cada iteración y el modelo aprende de la diversidad de los patrones de entrada.

4.6.2.3 Firmas

En la tabla 4.22 aparecen los resultados de entrenar los modelos ocultos de Markov con validación cruzada. Se realizaron dos pruebas dependiendo de la inicialización: aleatoria y la de igual ocupación de estados. El criterio de parada utilizado es el método umbral.

Método	Media	Varianza
IOE+Umbral	96.6 %	2.8
IOE+VC+Umbral	89.1 %	6.4
IOE+VCA+Umbral	91.1 %	5.6

Tabla 4.21: Comparación de las tasas de reconocimientode texto manuscrito con el método de inicialización, IOE, para entrenamientos que utilizan validación cruzada y validación cruzada aleatoria.

Método	Media	Varianza
A+VC+Umbral	56 %	19.3
IOE+VC+Umbral	62 %	17.7

Tabla 4.22: Tasa de reconocimiento de firmas aplicando validación cruzada (VC), con inicializaciones: aleatorio (A) e igual ocupación de estados (IOE); y con criterios de parada por umbral.

Método	Media	Varianza
A+VCA+Umbral	60 %	18.8
IOE+VCA+Umbral	75.5 %	13.2

Tabla 4.23: Tasa de reconocimiento de firmas aplicando validación cruzada aleatoria (VCA), con inicializaciones: aleatorio (A) e igual ocupación de estados (IOE); y con criterios de parada por: umbral.

Método	Media	Varianza
IOE+Umbral	92.0 %	6.3
IOE+VC+Umbral	62.0 %	17.7
IOE+VCA+Umbral	75.5 %	13.2

Tabla 4.24: Comparación de las tasas de reconocimiento con el método de inicialización, IOE, para entrenamientos que utilizan validación cruzada y validación cruzada aleatoria.

Dentro de los métodos de validación cruzada trabaja mejor el método donde todas las muestras del conjunto de entrenamiento y validación varían en cada iteración, dando así la posibilidad de mostrar la variedad de los datos al modelo. Por supuesto la tasa

de reconocimiento está muy por debajo de los modelos que sí han aprendido de todo el conjunto de entrenamiento.

A la vista de los resultados se observa que el reconocimiento de firmas es sensible a la cantidad de información suministrada durante el aprendizaje.

4.7 Estudio del cálculo computacional/tasa de reconocimiento

En este apartado se ha querido recoger la carga computacional (FLOPS, número de operaciones en punto flotante) empleado en el entrenamiento de cada uno de las modificaciones estudiadas en este capítulo. La carga computacional ha sido obtenida a través de la instrucción FLOPS que proporciona el paquete software MATLAB sobre el cual se han realizado las simulaciones. Este estudio se ha realizado sólo para la señal de voz.

4.7.1 Experimentos y resultados

4.7.1.1 Señal de voz

En la tabla 4.25 se compara la tasa de reconocimiento (media y varianza) con carga computacional cuando los HMMs son inicializados aleatoriamente. Estos datos aparecen visualizados en la figura 4.8

Método	Media	Varianza	Carga Computacional
Aleatorio	93.6 %	2.3	3.57865e+7
Equiprobable	93.8 %	3.8	4.18852e+7
IOE	96.0 %	2.2	4.33726e+7
No. Fijo de iteraciones	94.9 %	1.8	2.38548e+7
Media-Varianza	94.2 %	2.2	3.22453e+7
VC	93.0 %	1.9	1.81816e+7
VCA	94.0 %	2.3	1.76323e+7

Tabla 4.25: Tabla comparativa del cálculo computacional con la tasa de reconocimiento para los distintos tipos de inicialización y criterios de parada.

En la tabla 4.26 se compara la tasa de reconocimiento (media y varianza) con carga

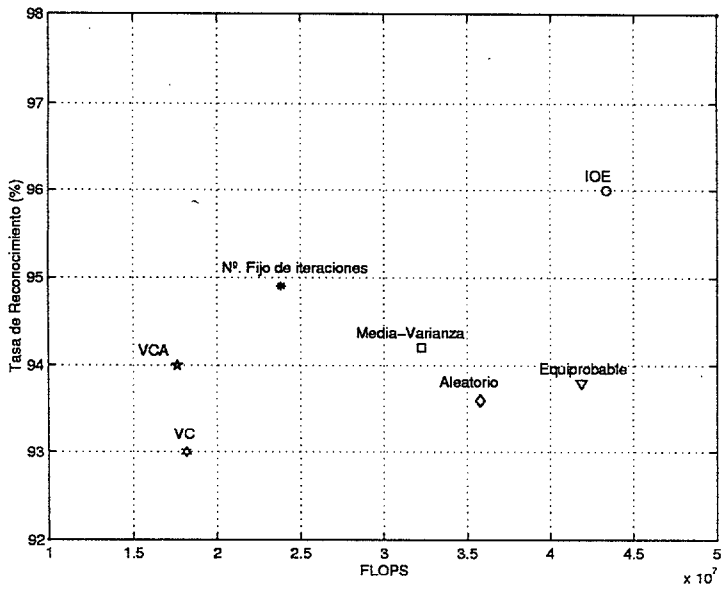

Figura 4.8: Tasa de reconocimiento/Carga computacional para distintos tipos de inicialización y criterios de parada

computacional, cuando los modelos han sido inicializados con el método de igual ocupación de estados (IOE). Estos datos aparecen visualizados en la figura 4.9

En la tabla 4.25 no se han incluído los métodos *voting model* y *reentrenamiento*, ya que la tasa de reconocimiento dependerá del número de modelos utilizados y por lo tanto, el coste computacional vendrá en función del número de modelos utilizados. Por supuesto, la tasa de reconocimiento es mejor pero el coste computacional y el tiempo invertido es mucho más elevado que cualquiera de los métodos propuestos en la tabla 4.25.

Método	Media	Varianza	Carga Computacional
Aleatorio	93.6 %	2.3	3.57865e+7
IOE	96.0 %	2.2	4.33726e+7
No. Fijo de iteraciones	94.7 %	2.5	2.51873e+7
Media-Varianza	95.8 %	2.2	3.63319e+7
VC	95.2 %	2.2	3.91808e+7
VCA	95.4 %	1.3	3.46259e+7

Tabla 4.26: Tabla comparatativa del cálculo computacional con la tasa de reconocimiento para los distintos criterios de parada con modelos inicializados con el método igual ocupación de estados (IOE).

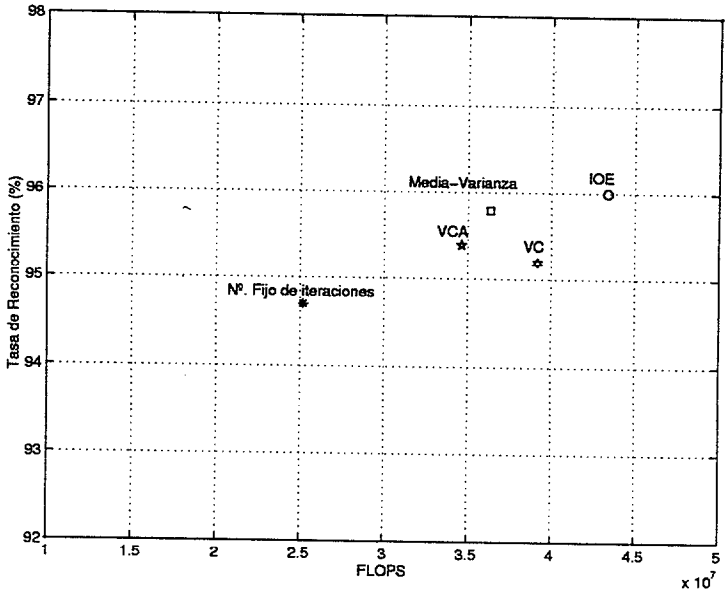

Figura 4.9: Tasa de reconocimiento/Carga computacional para modelos inicializados con el método igual ocupación de estados.)

Observando la figura 4.8, los puntos se encuentran más dispersos, encontrándonos con modelos como los de validación cruzada (VC) y validación cruzada aleatoria (VCA) en la parte central-izquierda. Esto indica que se tratan de modelos donde la relación Tasa de Reconocimiento/Coste computacional es media. Los métodos que se encuentren en la parte superior son los que mejor tasa de reconocimiento tienen. Y los que se encuentran más a la derecha implican un coste computacional más elevado. El método de inicialización de *igual ocupación de estados* se encuentra en la parte superior-derecha indicando que es un método de inicialización con una tasa de reconocimiento alta pero la carga computacional es la más elevada de todos los métodos.

En la figura 4.9, los puntos se encuentran más concentrados en la zona central superior. Esto nos indica que el método de inicialización propuesto *de igual ocupación de estados* funciona bastante bien con los métodos de parada. En líneas generales aumenta la tasa de reconocimiento a costa de un coste computacional, obteniéndose un rendimiento similar.

El comportamiento de las tablas 4.8 y 4.9 es similar para las aplicaciones de texto manuscrito y firmas. El cálculo computacional aumenta, cuando se aplica el método de inicialización de igual ocupación de estados, aunque al mismo tiempo se obtiene un incremento en la tasa de reconomiento, los puntos tienden a aglomerarse en una zona.

Capítulo 5

Sistemas Híbridos: HMM/NN

5.1 Introdución

Los modelos ocultos de Markov representan ser una herramienta efectiva para la clasificación de secuencias temporales sin embargo presentan poca capacidad de discriminación. Por otro lado, las redes neuronales presentan ser buenas clasificadoras de secuencias estáticas con una gran capacidad de discriminación. Sin embargo para secuencias temporales se requiere de estructuras neuronales recurrentes, las cuales necesitan de algoritmos de entrenamiento complejos, inestables y no las hacen tan atractivas para aplicaciones de este tipo. Un campo de trabajo de esta tesis ha sido el estudio de sistemas que resultan de la combinación de dos estructuras diferentes por un lado los HMMs y las NNs con el fin de obtener las ventajas de ambos. Esta combinación forma lo que se conoce por un sistema híbrido. Este sistema, utiliza por un lado la capacidad de clasificación de secuencias temporales de los HMMs y por otro, la capacidad de discriminación de las NNs.

En [Sharkey99] se comenta las experiencias de combinar redes neuronales para abordar un problema determinado. Existen dos maneras de combinar modelos para dar una solución a una tarea: comités (también conocido por *committee or ensembles*) y modularmente. El primer método, comités, combina varios sistemas, donde cada uno de ellos, da una respuesta ante un patrón de entrada y la salida final viene determinada por la contribución de cada sistema. El segundo método denominado *modular*, el problema a resolver se divide en diferentes subtareas, donde cada subtarea es desempeñada por módulos diferentes. La solución final vendrá dada por la contribución de cada módulo.

La diferencia entre ambos métodos está en la presencia o ausencia de información redundante. El método comité utiliza información redundante mientras que los segundos no.

En este capítulo se recogen las experiencias, simulaciones y resultados principalmente sobre un modelo híbrido modular. También se recogen experimentos de comités de modelos híbridos. El objetivo es la combinación de dos sistemas diferentes para abordar un problema, con el fin de obtener las ventajas de ambos.

5.2 Modelo híbrido

El modelo híbrido con el que se va a trabajar es un sistema de combinación modular. Está formado por la conexión en cascada de los modelos ocultos de Markov, seguidos por una estructura neuronal o red de conexión, ver la figura 5.1. Cuando se habla de estructura neuronal, se hace referencia tanto a una red neuronal como a la combinación de redes.

Esta estructura del híbrido que se propone, fue ya propuesto en [Cho97] para el reconocimiento de texto manuscrito, en él se sugiere un híbrido HMM/NN, aplicando las variables *forward*, $\alpha_T(i,l)$, ver ecuación 5.1 como entradas a una MLP. Esta tesis recoge los experimentos basados en esta misma estructura pero aplicada a la señal de voz, con una ligera modificación, en este caso las entradas a la red no son las variables $\alpha_T(i,l)$, sino su suma acumulativa, o lo que es lo mismo la variable probabilidad, ver ecuación 5.1.

$$P(O|i) = \sum_{l=1}^{N} \alpha_T(i,l) \tag{5.1}$$

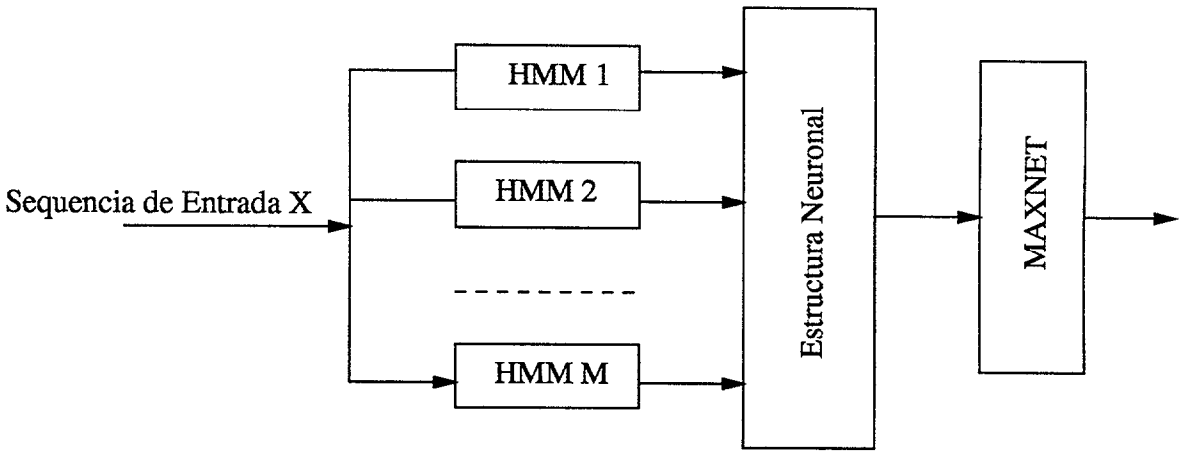

Figura 5.1: Modelo Híbrido: HMM y Estructura Neuronal.

Bajo la clasificación realizada por [Sharkey99] a la hora de combinar modelos: comités o modularmente; el modelo híbrido de la figura 5.1 se corresponde con los del segundo tipo. Las tareas se encuentran separadas y perfectamente definidas. Los HMMs son los

responsables de tratar el carácter temporal de la señal y las NNs de aumentar la capacidad de discriminación de los HMMs, corrigiendo errores cometidos por la primera etapa.

A partir de esta estructura modular que en principio parece sencilla se pueden obtener estructuras híbridas más complejas, ya que es posible a su vez crear comités de HMMs y comités de NNs, sus posibles combinaciones y a su vez, conectarlas modularmente.

El motivo principal de la combinación de estos dos sistemas, los HMMs y las NNs es la de mejorar la capacidad de generalización o dicho de otra manera corregir los errores que comete la primera etapa que compone el modelo híbrido.

En las estructuras modulares hay 4 modos de combinar los módulos: cooperativo, competitivo, secuencial y supervisión.

1. Cooperación, se entiende que todos los elementos que intervienen en el sistema modular contribuyen por igual o de manera ponderada en la decisión final.
2. Competitivo, se entiende que dependiendo de la entrada actuará un módulo u otro.
3. Secuencial, el proceso de funcionamiento es sucesivo.
4. Supervisión, en este caso un módulo puede ser usado para supervisar el funcionamiento de otro módulo.

El modo utilizado en el sistema híbrido que se propone utilizar en esta tesis es el de modo secuencial, donde las salidas del primer módulo son las entradas al segundo módulo. Es decir, las probabilidades de estimación de cada modelo HMM, $P(O|\lambda_i)$, ante una secuencia de observación son las entradas a la estructura neuronal o red de conexión.

5.2.1 Modelos ocultos de Markov del modelo híbrido

Según lo visto en el capítulo 3, los modelos ocultos de Markov asignan un modelo de N estados por clase a reconocer. Los modelos ocultos de Markov pueden ser analizados como una red neuronal donde cada estado l de cada modelo i, puede ser visto como un nodo l de cada red i, ver figura 5.2. Cuya salida es $\alpha_T(i,l)$ (variable forward) en el tiempo T, es decir, la probabilidad de que la secuencia de entrada pertenezca al modelo. Esta variable se calcula mediante el algoritmo *forward-backward* [Rabi92]. Sumando esta variable para todos los nodos de la red obtenemos la probabilidad acumulada por todos los nodos, ver ecuación 5.2.

$$P(O|i) = \sum_{l=1}^{N} \alpha_T(i,l) \tag{5.2}$$

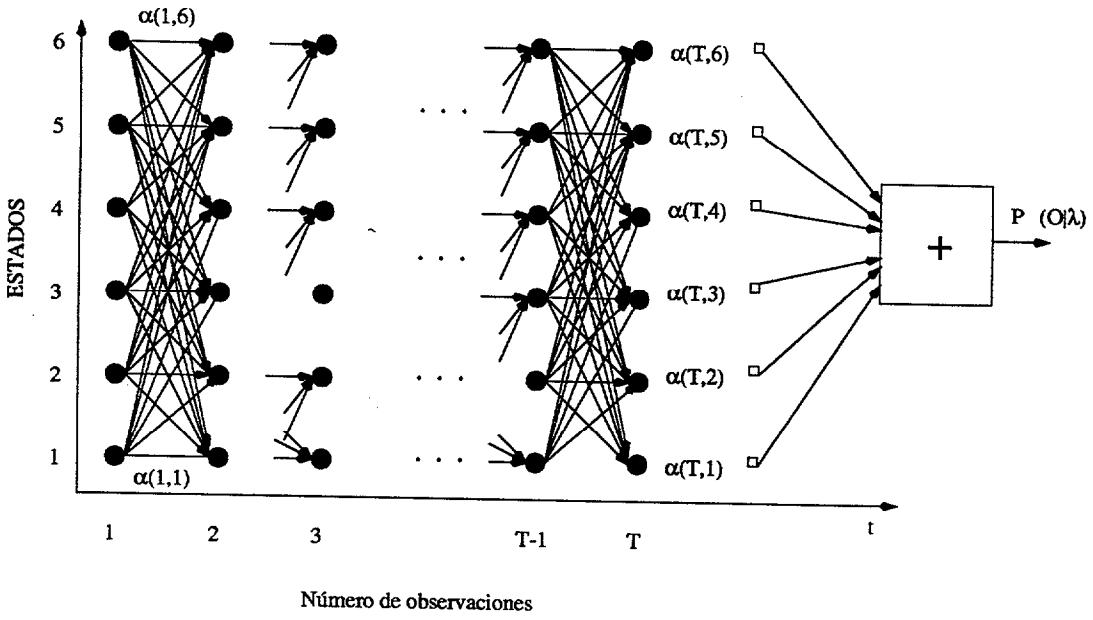

Figura 5.2: Modelos ocultos de Markov vistos como una estructura neuronal.

En las primeras pruebas se trabajó directamente con la variable $P(O|\lambda_i)$, como entradas a la red neuronal. Una segunda prueba fue la de utilizar como entrada a la red neuronal, la variable $\alpha_T(i,l)$, método propuesto en [Cho97]. Esto implica que el número de entradas aumenta, ya que por cada modelo hay N (estados) o variables α_T. Los resultados de utilizar esta entrada no presentaron ninguna mejoría respecto al método anterior, por lo que se decidió desechar este método debido a que, tanto la estructura de la red, como el cálculo computacional aumenta sin mejoría en los resultados.

5.2.2 Estructura neuronal del modelo híbrido

La estructura neuronal puede estar formada por una única red, o múltiples redes.

- Al primer método se le conoce con el término *All Class One Net*, (ACON), todas las clases a una única red. Esta estructura neuronal utiliza una única red, la cual tiene que reconocer o discriminar todas las clases. La salida de la red tendrá tantos nodos como clases a clasificar o modelos ocultos de Markov haya. Esta estructura se puede ver en la figura 5.1.

- El segundo método utiliza una estructura neuronal más compleja la cual puede estar formada por un conjunto de redes neuronales. Éstas se pueden organizar de dos maneras diferentes según la tarea que realicen:

  1. **Comités de redes.** Esta arquitectura trabaja con múltiples redes ACON. Este tipo de estructura se reconocen porque el conjunto de redes desarrollan

la misma tarea, todas las redes actúan para dar la misma salida, la misma respuesta. Se trata de una estructura formada por varias redes ACON trabajando en paralelo. Las salidas de todas las redes, deben ser analizadas para seleccionar una clase. Con esta primera idea, se puede pensar, que si todas las redes son iguales, todas están entrenadas con el mismo fichero de entrenamiento, con el mismo algoritmo, etc., la salida de todas las redes neuronales será la misma, ésto implica que todas las redes proporcionan la misma información. Pues bien, este tipo de modalidad consiste en trabajar con múltiples redes ACON pero cada una ellas presenta alguna diferencia, por ejemplo, aplicando diferentes inicializaciones en sus pesos, usando distintos criterios de parada, variando la topología, diferentes algoritmos de entrenamiento, usando diferentes conjuntos de entrenamiento, etc. De esta manera cada red neuronal aporta información importantísima a la hora de clasificar. Por supuesto este tipo de estructuras lleva un cálculo computacional más elevado que la configuración con una sóla red.

2. **Arquitecturas modulares.** Este tipo de estructura, utiliza redes *One Class One Net*, (OCON), una clase, una red. Este tipo de arquitecturas utiliza el principio de que un problema se puede descomponer en problemas más pequeños. Llevando este principio a la estructura neuronal supondría la utilización de redes más sencillas, donde cada red se encarga de clasificar una sola clase. Para llevar a cabo esta descomposición, es necesario conocer bien la tarea a desarrollar. Para el problema que se está tratando, una posible solución sería dividir la red MLP (ACON) en tantas redes neuronales como clases deseemos clasificar. Es lógico pesar que esta estructura de múltiples redes trabajando sobre un mismo problema permite que el sistema sea más discriminativo ya que una red neuronal sólo tiene que generar una frontera de decisión. Comprobaremos que realmente este tipo de estructuras de múltiples redes funcionan mejor que una única red. Ya en [Sharkey96] se pone de manifiesto que la descomposición de un problema supone una mejora en el funcionamiento del sistema propuesto.

   Al tratarse de un conjunto de redes, el entrenamiento de todas ellas implica un aumento en la carga computacional pero no tan elevada con en el caso de la arquitectura de redes ACON, ya que estas redes tienen una estructura más sencilla.

5.3 Híbrido HMM con estructura neuronal MLP ACON

Este primer experimento se centra en una red MLP de una única capa oculta con L nodos. Cada modelo HMM, estima una probabilidad de que una secuencia de observación haya sido generada por el modelo y éste la reconozca como suya. Es justamente esta variable $P(O|\lambda_i)$, previa normalización respecto al máximo, la entrada a la red neuronal. Esta probabilidad es la suma acumulativa de la variable *forward* $\alpha_T(i,l)$ de cada modelo, ver ecuación 5.2.

La estructura de la red queda perfectamente definida por tener tantos nodos de entrada, como clases a clasificar o modelos ocultos de Markov hayan. El mismo número de nodos se utiliza en la salida, ya que cada nodo de salida representa una clase. Los nodos de la capa intermedia se obtuvieron experimentalmente mediante prueba y error.

La estructura de la red neuronal utilizada es:

1. Capa de entrada: 10 nodos.
2. Capa oculta, puede tener entre 15 y 20 nodos.
3. Capa de salida: 10 nodos.

La función de cada nodo es la función *sigmoidal*, ver figura 5.3. La red neuronal fue entrenada con el algoritmo clásico de gradiente conjugado *backpropagation*. El algoritmo de gradiente conjugado puede buscar el mínimo de una función mucho más rápido que el método convencional *backpropagation*, ver [Kung93]. Las muestras de entrenamiento fueron dadas a la red en bloque. Este método se utiliza sobre todo cuando no es necesario un entrenamiento en tiempo real.

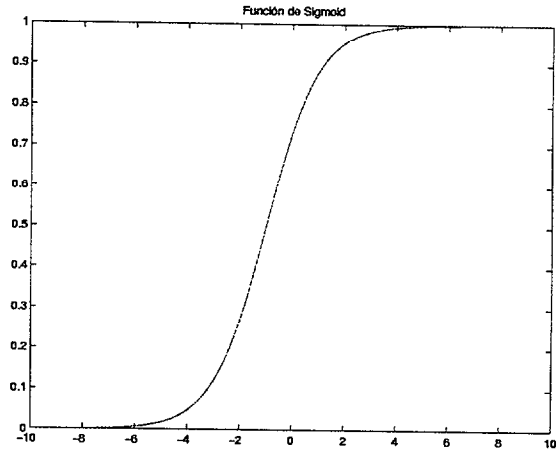

Figura 5.3: Función de transferencia sigmoidal

La salida de esta red vendría dada por la ecuación 5.3

$$O(i|X) = f\left\{\sum_{l=1}^{L} w_2(i,l) \cdot f\{\sum_{j=1}\sum_{k=1}^{N} w_1(l,j,k) \cdot \alpha_T(j,k)\}\right\} \tag{5.3}$$

donde $w_2(i,l)$ es el peso o valor de la conexión del nodo l de la capa oculta hacia el nodo i de la capa de salida, el peso $w_1(l,j,k)$ es el peso que define la conexión del estado

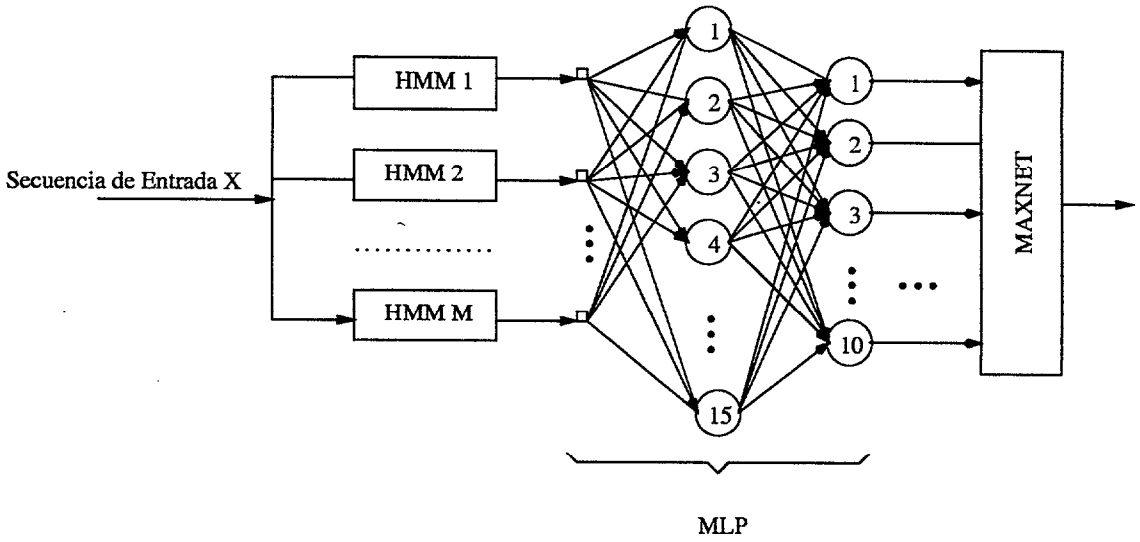

Figura 5.4: Modelo Híbrido: HMM/MLP

l del modelo HMM j al nodo k de la capa oculta y f es la función global de activación, típicamente se utiliza la función sigmoidal o tangente hiperbólica.

En las primeras pruebas de entrenamiento llevadas a cabo con la MLP ACON, se observó que la red neuronal empeoraba los resultados obtenidos, por los HMMs utilizando únicamente los HMMs. No se consiguía mejoría alguna. Ésto se debió a que se utilizó los mismos ficheros de entrenamiento de los modelos ocultos de Markov, pero tomados a la salida de éstos. La mayoría de los modelos entrenados alcanzan una tasa de reconocimiento del 100% con este conjunto. Cuando ésto sucede, la red neuronal es entrenada con datos sin errores, por lo que la red no puede aprender a corregir errores cometidos por los HMMs, ya que carece de esa información en el fichero de entrenamiento. Es por lo que el sistema híbrido formado por la unión en cascada de los modelos ocultos de Markov (HMM) con la red neuronal (NN) no puede dar resultados mejores que los HMMs. Para solventar este problema se tuvo que añadir errores intencionadamente en los ficheros de entrenamiento. Con esto se consiguió una mejora, y así la red pudo dar respuestas adecuadas ante entradas con errores.

5.3.1 Experimentos y resultados

5.3.1.1 Señal de voz

Los resultados que aquí presentamos han sido obtenidos con la base de datos de 5 locutores con 132 realizaciones. La secuencia de test son las 100 primeras realizaciones y las 32 restantes para entrenamiento. Presentamos dos tipos de pruebas de esta estruc-

tura híbrida, las cuales utilizan modelos ocultos de Markov diferentes dependientes de la inicialización de estos modelos:

1. En la tabla 5.1 aparecen los resultados de aplicar el modelo híbrido HMM/MLP ACON. Estos resultados son los valores promedios de entrenar 10 estructuras híbridas.

2. La segunda prueba se utilizó, los modelos optimizados con el método de inicilización de *igual ocupación de estados.* Recordemos que con este tipo de inicialización se obtuvieron modelos con la tasa de reconocimiento más alta. Los resultados de aplicar la estructura híbrida HMM/MLP ACON se encuentran en la tabla 5.2. Estos resultados son los valores promedios de 10 estructuras híbridas.

Modelo	Loc 1	Loc 2	Loc 3	Loc 4	Loc 5	Media	Varianza
HMM	94.6 %	93.2 %	96.9 %	91.4 %	91.8 %	93.6%	2.3
HMM/MLP	98.3 %	98.1%	95.6%	98.6%	96.7%	97.5%	1.3

Tabla 5.1: Tasa de reconocimiento de voz del modelo híbrido HMM/MLP ACON. Los HMMs inicializados aleatoriamente).

Modelo	Loc 1	Loc 2	Loc 3	Loc 4	Loc 5	Media	Varianza
HMM	96.9 %	97.1 %	97. 3 %	96.9 %	92.1 %	96.0%	2.2
HMM/MLP	97.8 %	98.6 %	95.8 %	98.6 %	97.1 %	97.6%	1.2

Tabla 5.2: Tasa de reconocimiento de voz del modelo híbrido HMM/MLP ACON. Los HMMs inicializados con el método de igual ocupación de estados, (IOE).

Para ambas pruebas se consiguen mejoras frente a los HMMs trabajando individualmente, mejoras tanto en media como en varianza. Sin embargo se puede observar que la mejoría presentada por la primera prueba es cuantitativamente más grande que para la segunda prueba. Esto se debe principalmente a que los modelos ocultos de Markov entrenados aleatoriamente tienen una tasa de error más alta que los inicializados por el método de *igual ocupación de estados*, esto significa que el fichero de entrenamiento de la red neuronal tiene más datos, más información de errores cometidos por los HMMs. Cuanto mejor funcionen los HMMs peor será la mejora introducida por la red neuronal. Se ha de destacar que en ambas pruebas la varianza es menor en el modelo híbrido indicando un comportamiento uniforme entre los distintos locutores.

Modelo	Media	Varianza
HMM	92.5 %	7.9
HMM/MLP	96.8 %	5.6

Tabla 5.3: Tasa de reconocimiento de texto manuscrito del modelo híbrido HMM/MLP ACON. Los HMMs están inicializados aleatoriamente.

Modelo	Media	Varianza
HMM	96.6 %	2.8
HMM/MLP ACON	98.8 %	1.3

Tabla 5.4: Tasa de reconocimiento de texto manuscrito del modelo híbrido HMM/MLP ACON. Los HMMs están inicializados con el método igual ocupación de estados.

5.3.1.2 Texto manuscrito

Para la aplicación de texto manuscrito se utilizaron 20 escritores. De cada escritor se disponían de 12 realizaciones. La distribución de las realizaciones para el entrenamiento y test es como sigue, 8 para entrenamiento y 4 para test. Como resultado, cada modelo se entrenó con 160 muestras y se procedió a realizar el test con 80 realizaciones. Se realizó los mismos dos experimentos que para la señal de voz. El primero con modelos inicializados aleatoriamente y el segundo con modelos inicializados con el método de igual ocupación de estados. En las tablas 5.3 y 5.4 aparecen los resultados de ambas pruebas.

5.3.1.3 Firmas

Para la aplicación de firmas se utilizaron 10 escritores. De cada escritor se disponían de 24 realizaciones. La distribución de las realizaciones para el entrenamiento y test es como sigue, 14 para entrenamiento y 10 para test. Se realizó los mismos dos experimentos que para la señal de voz. El primero con modelos inicializados aleatoriamente y el segundo con modelos inicializados con el método de igual ocupación de estados. En las tablas 5.5 y 5.6 aparecen los resultados de ambas pruebas.

Método	Media	Varianza
HMM	84.0 %	13.5
HMM/MLP	93.4 %	6.4

Tabla 5.5: Tasa de reconocimiento de firmas del modelo híbrido HMM/MLP ACON. Los HMMs inicializados aleatoriamente

Método	Media	Varianza
HMM	92.0 %	6.3
HMM/MLP	97.8 %	2.3

Tabla 5.6: Tasa de reconocimiento de firmas del modelo híbrido HMM/MLP ACON. Los HMMs inicializados con el método igual ocupación de estados

5.4 Híbrido HMM con estructura neuronal RBF ACON

Otro tipo de red empleado es la RBF. Como es sabido las redes de base radial tienen una función de base que definen una hiperesfera. Se trata de un tipo de red que combina funciones locales. Esta red tiene dos capas, una primera, que aplica una función de activación Gaussiana y una segunda capa, la cual es lineal. La salida de cualquier nodo i viene dada por la ecuación 5.4. Analizando esta expresión matemática, la parte exponencial se corresponde con el cálculo de la primera capa, la función Gaussiana; al producto de esta parte por los pesos de la segunda capa se corresponde con la parte lineal.

$$O(i|X) = \sum_{l=1}^{L} w(i,l) \exp\left\{ -\sum_{j=1}^{M}\sum_{k=1}^{N} \frac{(\alpha_t(j,k) - \mu(l,j,k))^2}{2\sigma^2(l,j,k)} \right\} \tag{5.4}$$

Donde $w(i,l)$ son los pesos del nodo l de la capa Gaussiana al nodo de salida i, $\mu(l,j,k)$ y $\sigma^2(l,j,k)$ son la media y varianza respectivamente de la función Gaussiana.

Este tipo de red divide el espacio en esferas donde cada una de ellas viene definida por un centroide. De la ecuación 5.4 los centroides vienen dados por $\mu(l,j,k)$. Éstos se obtienen bien por un entrenamiento competitivo como puede ser el algoritmo de las K-medias (*K-means*) o bien seleccionando adecuadamente estos centroídes, en función de la aplicación. La varianza σ^2 puede estimarse a partir de las muestras incluídas en cada clase, veáse 5.5 y $\mu(l,j,k)$ como una varianza.

$$\sigma_k^2 = \frac{1}{\#\mu_k} \sum_{\alpha \in \mu_k} \|\alpha - \mu_k\| \tag{5.5}$$

Las RBFs resultan ser eficaces en problemas de clasificación ya que combina funciones locales Gaussianas para construir las regiones de decisión a diferencia de las MLP que combinan funciones globales sigmoidal.

5.4.1 Experimentos y resultados

Se observará que la mejora introducida por esta red es cuantitativamente inferior a la introducida por la red MLP para las distintas señales.

5.4.1.1 Señal de Voz

Los resultados que aquí presentamos han sido obtenidos con la base de datos de 5 locutores con 132 realizaciones. La secuencia de test son las 100 primeras realizaciones y las 32 restantes para entrenamiento.

Al igual que en el experimento anterior, presentamos dos tipos de pruebas de esta estructura híbrida, dependientes de la inicialización de los modelos ocultos de Markov:

1. La primera se llevó a cabo con modelos ocultos de Markov inicializados de manera aleatoria. En la tabla 5.7 aparecen los resultados de aplicar el modelo híbrido HMM/RBF ACON. Estos resultados son los valores promedios de entrenar 10 estructuras híbridas.

2. La segunda prueba se utilizó, los modelos optimizados con el método de inicilización de *igual ocupación de estados.* Recordemos que con este tipo de inicialización se obtuvieron modelos con la tasa de reconocimiento más alta. Los resultados de aplicar la estructura híbrida HMM/RBF ACON se encuentran en la tabla 5.8. Estos resultados son los valores promedios de 10 estructuras híbridas.

Los resultados de estas pruebas presentan una cierta mejoría de más de **1 punto**, respecto a los modelos HMMs, al igual que en la varianza.

Modelo	Loc 1	Loc 2	Loc 3	Loc 4	Loc 5	Media	Varianza
HMM	94.6 %	93.2 %	96.9 %	91.4 %	91.8 %	93.6 %	2.3
HMM/RBF	97.8 %	98.6 %	95.8 %	98.6 %	97.1 %	97.6 %	1.2

Tabla 5.7: Tasa de reconocimiento de voz del modelo híbrido HMM/RBF. Los HMM han sido inicializados aleatoriamente.

Modelo	Loc 1	Loc 2	Loc 3	Loc 4	Loc 5	Media	Varianza
HMM	96.9 %	97.1 %	97.3 %	96.9 %	92.1 %	96.0 %	2.2
HMM/RBF	97.8 %	98.6 %	95.8 %	98.6 %	97.1 %	97.6 %	1.2

Tabla 5.8: Tasa de reconocimiento de voz del modelo híbrido HMM/RBF. Los HMMs han sido inicializados con el método de igual ocupación de estados (IOE).

5.4.1.2 Texto manuscrito

Los resultados que aquí presentamos han sido obtenidos con la base de datos de 20 escritores. La secuencia de test es de tamaño 80 y la de entrenamiento 160.

Al igual que en los experimentos anteriores, presentamos dos tipos de pruebas de esta estructura híbrida, dependientes de la inicialización de los modelos ocultos de Markov:

1. La primera se llevó a cabo con modelos ocultos de Markov inicializados de manera aleatoria. En la tabla 5.9 aparecen los resultados de aplicar el modelo híbrido HMM/RBF ACON.
2. La segunda prueba se utilizó, los modelos optimizados con el método de inicilización de *igual ocupación de estados*. Los resultados de aplicar la estructura híbrida HMM/RBF ACON se encuentran en la tabla 5.10.

Modelo	Media	Varianza
HMM	92.6 %	7.9
HMM/RBF	93.5 %	5.1

Tabla 5.9: Tasa de reconocimiento de texto manuscrito del modelo híbrido HMM/RBF. Los HMM han sido inicializados aleatoriamente.

La mejora introducida por esta red es cuantitativamente inferior a la introducida por la red MLP.

Modelo	Media	Varianza
HMM	96.6 %	2.8
HMM/RBF	97.0 %	2.6

Tabla 5.10: Tasa de reconocimiento de texto manuscrito del modelo híbrido HMM/RBF. Los HMMs han sido inicializados con el método de igual ocupación de estados (IOE).

5.4.1.3 Firmas

Los resultados que aquí presentamos han sido obtenidos con la base de datos de 10 firmantes. Cada modelo fue entrenado con 14 firmas y el test se realizó con 10 firmas diferentes a las de entrenamiento.

Al igual que en los experimentos anteriores, presentamos dos tipos de pruebas de esta estructura híbrida, dependientes de la inicialización de los modelos ocultos de Markov:

1. La primera se llevó a cabo con modelos ocultos de Markov inicializados de manera aleatoria. En la tabla 5.11 aparecen los resultados de aplicar el modelo híbrido HMM/RBF ACON.
2. La segunda prueba se utilizó, los modelos optimizados con el método de inicilización de *igual ocupación de estados*. Los resultados de aplicar la estructura híbrida HMM/RBF ACON se encuentran en la tabla 5.12.

Modelo	Media	Varianza
HMM	84.0 %	13.5
HMM/RBF	90.1 %	9.4

Tabla 5.11: Tasa de reconocimiento de firmas del modelo híbrido HMM/RBF. Los HMM han sido inicializados aleatoriamente.

Modelo	Media	Varianza
HMM	92.0 %	6.3
HMM/RBF	95.7 %	5.8

Tabla 5.12: Tasa de reconocimiento de firmas del modelo híbrido HMM/RBF. Los HMMs han sido inicializados con el método de igual ocupación de estados (IOE).

5.5 Híbrido con comités de redes neuronales

Hasta el momento se han visto dos tipos de estructuras híbridas, los formados por las redes perceptron multicapa, (MLP) y las redes de funciones de base radial , (RBF). A continuación se estudia sistemas híbridos cuya estructura neuronal está formada por comités de estas redes. Recordemos que esta arquitectura trabaja con múltiples redes ACON. Este tipo de estructura se reconocen porque el conjunto de redes desarrollan la misma tarea, todas las redes actúan para dar la misma salida, la misma respuesta. Se trata de una estructura formada por varias redes ACON trabajando en paralelo.

La razón más importante a la hora de combinar y formar nuevas estructuras es que un sistema, cuando es entrenado para dar una determinada salida puede cometer errores frente a determinadas entradas. Esto se produce porque el conjunto de entrenamiento es limitado. La estimación de un conjunto de estimadores imperfectos puede pensarse como una forma de manejar las limitaciones de cada estimador por separado. Es decir, cada estimador comete sus errores pero combinados de una determinada manera se pueden minimizar dichos errores. Es evidente que no se obtienen ventajas, con un comité formado por sistemas idénticos ya que todos generalizan de la misma manera, cometen los mismos errores. En principio, los sistemas pueden ser diferentes en cuanto a estructura, parámetros, inicializaciones, etc. pero si comenten los mismos errores no estamos mejorando el sistema, simplemente añadiendo más carga computacional. Lo interesante de la combinación de estructuras neuronales es la de encontrar modelos o sistemas que generalizan de manera diferente. Que los errores que comenten cada sistema estén incorrelados. De esta manera cada red neuronal aporta información importantísima a la hora de clasificar. Por supuesto este tipo de estructuras lleva un cálculo computacional más elevado que la configuración con una sóla red.

En [Sharkey99] propone diferentes métodos para conseguir este efecto:

1. **Inicialización**. Los modelos pueden ser inicializados de manera aleatoria manteniendo fijo el conjunto de entrenamiento.

2. **Topología**. Los modelos pueden tener una arquitectura o topología diferente, manteniendo fijo el conjunto de entrenamiento. Una posibilidad que puede darse es que al variar la topología de manera radical, pueda derivarse en que los errores estén incorrelados.

3. **Algoritmo**. El algoritmo de entrenamiento utilizado para entrenar los modelos podría ser diferente, manteniendo constante el conjunto de entrenamiento.

4. **Conjunto de entrenamiento**. Los métodos más utilizados para crear comités es justamente variar el conjunto de entrenamiento. Hay numerosas técnicas para hacer esto:

- *Sampling Data.* Cada modelo es entrenado con un subconjunto del conjunto de entrenamiento. Dentro de estos métodos se encuentran el de validación cruzada, *bootstrapping* and *smootbootstrapping.*
- *Conjuntos de entrenamiento disconjuntos.* Se utiliza conjuntos de entrenamiento disjuntos. No hay solape entre los datos utilizados en el entrenamiento de los distintos modelos.
- *Boosting and Adaptative Resampling.* [Schapire99] demostró que una serie de modelos débiles pueden convertirse en un modelo fuerte como resultado de entrenamiento de cada uno de los módulos del comité con patrones que con frecuencia son mal clasificados, se trata de un entrenamiento adaptativo.
- *Fuente de datos diferentes.* Bajo ciertas circunstancias es posible disponer de fuentes diferentes.
- *Preprocesado.* Los datos pueden ser modificados utilizando otras técnicas de preprocesado, extrayendo carácterísticas diferentes o bien distorsionar los datos de entrada aplicando transformaciones no lineales, añadiendo ruido, etc.

5.5.1 Híbrido HMM con comité de MLPs ACON

Como hemos comentado anteriormente, una posibilidad es la aplicación de múltiples redes MLPs ACON, donde cada red contribuyen en la salida final. Cada red neuronal estima una clase. Siendo la clase más votada, la elegida. Esto significa que varias redes neuronales aportan su información para dar una salida correcta. Por supuesto este tipo de estructura requiere un cálculo computacional más elevado. Si el número de redes es **n**, el cálculo computacional es **n** veces más que el de una red neuronal MLP ACON. En la figura 5.5 aparece esta estructura.

Un problema que se encontró al trabajar con este tipo de estructura es la elección del número de redes y las clases a detectar. El número de redes utilizadas debe ser superior a 2, a no ser que la salida de las redes estén ponderadas dando preferencia a una de las dos redes. El motivo de que el número de redes sea superior a 2, es que al trabajar con dos redes, la tarea de clasificación puede terminar en un empate de clases y elegir la clase incorrecta. Sin embargo cuando el número de redes que participan en la decisión final es impar e igual a 3, esta situación no aparece, eligiéndose la clase más votada, siempre y cuando el número de clases a detectar sea inferior a 3. Si el número de clases es superior a 3, puede ocurrir que las tres redes proporcionen salidas diferentes, por lo que tenemos una nueva situación de empate de clases.

Cuando el número de redes aumenta, el que se dé una situación de empate puede ocurrir tanto para estructuras con un número par e impar de redes. Si la estructura tiene un conjunto par de redes, para que se dé una situación de empate es que las redes que intervienen en la toma de decisión final se pueden agrupar en conjuntos pares de redes de **igual tamaño**, votando a clases diferentes. Si el número de redes es impar, puede darse

que se originen varios conjuntos pares de **igual tamaño** y conjuntos pares e impares de redes de menor tamaño los cuales no intervienen en la decisión final.

Por supuesto con un número impar, la probabilidad de que se den situaciones de empate es menor. También disminuye cuando el número de redes de la estructura ACON aumenta. Y por supuesto el mejor método es que las redes generalicen de manera diferente de tal forma que los errores estén incorrelados.

5.5.2 Experimentos y resultados

Esta prueba sólo se realizó para la señal de voz

5.5.2.1 Señal de voz

Los resultados que aquí presentamos han sido obtenidos con la base de datos de 5 locutores con 132 realizaciones. La secuencia de test son las 100 primeras realizaciones y las 32 restantes para entrenamiento.

La estructura de las redes es la misma que para los híbridos (HMM/MLP ACON). Diez nodos en la capa de entrada, quince nodos en la capa oculta y diez nodos en la capa de salida. La clase elegida por cada red neuronal es contabilizada, eligiéndose la clase más votada.

Al igual que en los experimentos anteriores, presentamos dos tipos de pruebas de esta estructura híbrida, dependientes de la inicialización de los modelos ocultos de Markov. En estas tablas aparecen los resultados cuando intervienen varias redes MLPS, hasta un máximo de 10 MLPs, en la toma de decisión final. Los resultados de ambas tablas son los valores promedios de entrenar 10 estructuras híbridas:

1. La primera se llevó a cabo con modelos ocultos de Markov inicializados de manera aleatoria. En la tabla 5.13 aparecen los resultados de aplicar el modelo híbrido HMM/Comité MLP ACON cuando los modelos ocultos de Markov han sido inicializados aleatoriamente.

2. La segunda prueba se utilizó, aplicando a los modelos el método de inicilización de *igual ocupación de estados.* Recordemos que con este tipo de inicialización se obtuvieron modelos con la tasa de reconocimiento más alta. Los resultados de aplicar estos modelos a la estructura híbrida HMM/Comité MLP ACON se encuentran en la tabla 5.14.

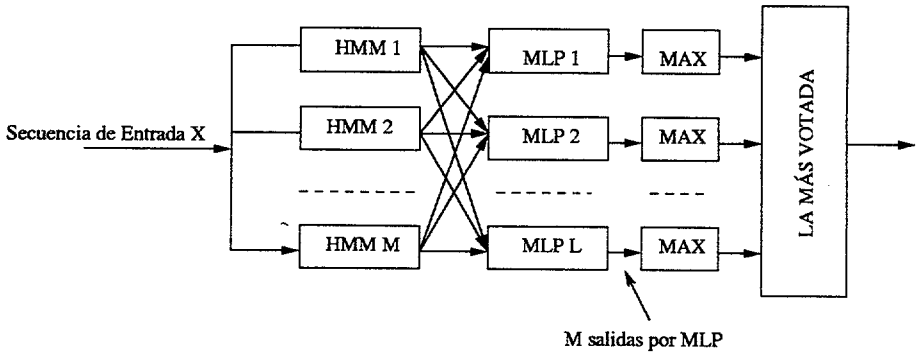

Figura 5.5: Modelo Híbrido: HMM/Comités MLP

En la tabla 5.13 aparecen los resultados de aplicar distintas redes neuronales MLPs. Cada red ha sido inicializada aleatoriamente.

HMM/Comité MLP	Loc 1	Loc 2	Loc 3	Loc 4	Loc 5	Media	Varianza
HMM/1 MLP	97.0 %	97.9 %	95.5 %	96.6 %	97.1 %	96.8 %	0.9
HMM/2 MLPs	95.6 %	97.9 %	95.4 %	96.0 %	96.5 %	96.3 %	1.0
HMM/3 MLPs	98.9 %	99.3 %	97.7 %	98.1 %	96.9 %	98.2 %	1.0
HMM/4 MLPs	98.5 %	99.4 %	98.5 %	98.3 %	97.3 %	98.4 %	0.7
HMM/5 MLPs	99.0 %	99.5 %	97.9 %	99.0 %	97.4 %	98.6 %	0.9
HMM/6 MLPs	98.8 %	99.6 %	97.9 %	99.0 %	98.1 %	98.7 %	0.7
HMM/7 MLPs	99.2 %	99.5 %	97.6 %	99.1 %	98.6 %	98.8 %	0.7
HMM/8 MLPs	99.0 %	99.6 %	97.8 %	99.0 %	98.1 %	98.7 %	0.7
HMM/9 MLPs	99.0 %	99.6 %	97.5 %	99.0 %	98.8 %	98.8 %	0.8
HMM/10 MLPs	99.0 %	99.6 %	97.6 %	99.1 %	98.2 %	98.7 %	0.8

Tabla 5.13: Tasa de reconocimiento de voz del modelo híbrido HMM/Comité MLP aplicando distintos números de MLPs. Los HMMs están inicializados aleatoriamente.

Lo comentado anteriormente sobre la incidencia de considerar en la estructura híbrida con comité de redes neuronales, un número par bajo de MLPs, se puede observar en las tablas 5.13 y 5.14. En la primera se produce para la mayoría de los locutores, la tasa de reconocimiento cuando se utiliza 2 MLPs es inferior a la de una de las redes MLP que intervienen en el comité. Para la segunda tabla, sucede únicamente para el locutor 1.

En la tabla 5.15 aparecen los resultados de aplicar distintas redes con inicialización, topología, algoritmo y conjuntos de entrenamiento diferentes. Los resultados se han obtenido a partir de 10 estructuras neuronales.

Como se puede observar, en la tabla 5.15, los resultados son muy buenos, con medidas de tasas de reconocimiento altas y las varianzas bajas. Con la aplicación de estos métodos: inicialización, topología, algoritmo y conjunto de entrenamiento se consiguen resultados

HMM/Comité MLP	Loc 1	Loc 2	Loc 3	Loc 4	Locutor 5		
HMM/1 MLP	96.5 %	97.5 %	93.0 %	97.3 %	96.2 %	96.1 %	1.8
HMM/2 MLPs	96.0 %	97.6 %	95.6 %	97.8 %	96.8 %	96.8 %	1.0
HMM/3 MLPs	97.5 %	98.9 %	97.4 %	98.2 %	98.2 %	98.0 %	0.6
HMM/4 MLPs	97.3 %	98.9 %	96.9 %	98.4 %	97.8 %	97.9 %	0.8
HMM/5 MLPs	98.0 %	98.9 %	97.6 %	98.8 %	98.7 %	98.4 %	0.6
HMM/6 MLPs	97.7 %	99.5 %	97.3 %	98.7 %	98.8 %	98.4 %	0.9
HMM/7 MLPs	97.8 %	99.6 %	97.4 %	98.8 %	98.9 %	98.5 %	0.9
HMM/8 MLPs	97.9 %	99.6 %	97.4 %	98.5 %	98.8 %	98.4 %	0.8
HMM/9 MLPs	98.2 %	99.0 %	97.4 %	98.8 %	98.7 %	98.4 %	0.6
HMM/10 MLPs	98.4 %	99.4 %	97.5 %	98.6 %	98.8 %	98.5 %	0.7

Tabla 5.14: Tasa de reconocimiento de voz del modelo híbrido HMM/Comité MLP aplicando distintos números de MLPs. Los HMMs están inicializados con el método de igual ocupación de estados.

HMM/Comité MLP	Loc 1	Loc 2	Loc 3	Loc 4	Loc 5	Media	Varianza
Inicialización	98.0 %	98.3 %	91.4 %	97.8 %	96.1 %	96.3 %	2.9
Topología	98.9 %	98.7 %	98.4 %	99.3 %	98.6 %	98.8 %	0.34
Algoritmo	98.5 %	97.9 %	95.8 %	97 %	97.2 %	98.3 %	1.0
Conjunto	97.4 %	98.0 %	97.3 %	97.4 %	96.7 %	97.4 %	0.4

Tabla 5.15: Tasa de reconocimiento de voz del modelo híbrido HMM/Comité MLP aplicando los distintos métodos.

muy buenos, ésto se debe a que los parámetros de estas redes neuronales se encuentren más incorreladas debido a que una estructura diferente en la red, inicializaciones diferentes, conjuntos de entrenamiento distintos, hacen que las redes evolucionen en el aprendizaje de distinta manera y entre ellas se puedan corregir errores.

5.6 Híbrido HMM con estructura neuronal modular MLP OCON

Otra alternativa de estructura neuronal, es la descomposición del problema en múltiples redes, como ya se ha comentado en apartados anteriores. Esta estructura requiere de tantas redes MLPs como modelos ocultos de Markov hayan o clases a clasificar. Esta

estructura requiere de un cálculo computacional más elevado que si utilizasemos un única red, sin embargo el entrenamiento de dichas redes es más rápido debido a que su estructura es más sencilla. En la figura 5.6 aparece la estructura de este modelo híbrido.

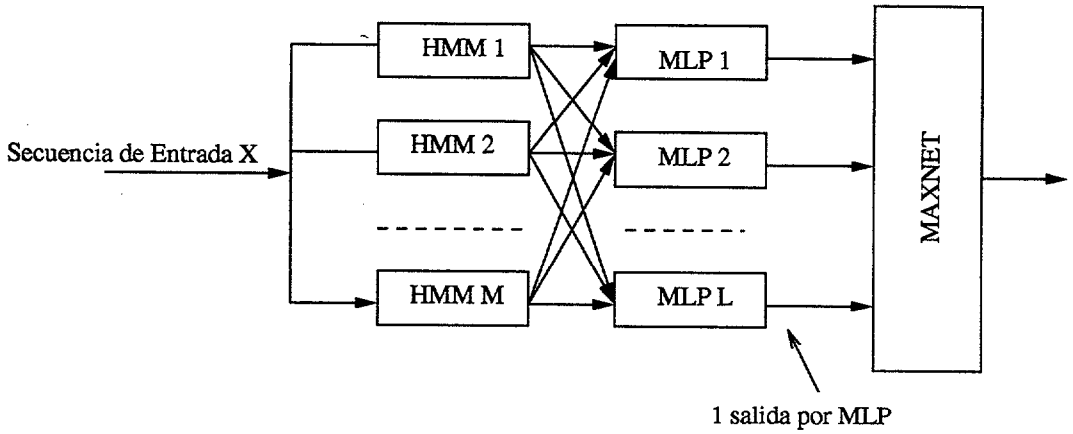

Figura 5.6: Modelo Híbrido: HMM con Estructura Neuronal Modular MLP OCON

La estructura de cada red sería mucho más sencilla:

1. Capa de entrada: 10 nodos.
2. Capa oculta: 15 nodos.
3. Capa de salida: 1 nodo

La tarea de cada red es simplemente detectar una clase de patrones y el resto discriminarlos.

5.6.1 Experimentos y resultados

5.6.1.1 Señal de voz

Los resultados que aquí presentamos han sido obtenidos con la base de datos de 5 locutores con 132 realizaciones. La secuencia de test son las 100 primeras realizaciones y las 32 restantes para entrenamiento.

Presentamos dos tipos de pruebas de esta estructura híbrida, las cuales utilizan modelos ocultos de Markov diferentes dependientes de la inicialización de los modelos:

1. La primera se llevó a cabo con modelos ocultos de Markov inicializados de manera aleatoria. En la tabla 5.16 aparecen los resultados de aplicar el modelo híbrido HMM/MLP OCON. Estos resultados son los valores promedios de entrenar 10 estructuras híbridas.

2. La segunda prueba se utilizó, los modelos optimizados con el método de inicilización de *igual ocupación de estados*. Recordemos que con este tipo de inicialización se obtuvieron modelos con la tasa de reconocimiento más alta. Los resultados de aplicar la estructura híbrida HMM/MLP OCON se encuentran en la tabla 5.17. Estos resultados son los valores promedios de entrenar 10 estructuras híbridas.

	Loc 1	Loc 2	Loc 3	Loc 4	Loc 5	Media	Varianza
HMM	94.6 %	93.2 %	96.9 %	91.4 %	91.8 %	93.6 %	2.3
HMM/MLP OCON	95.9 %	97.0 %	94.9 %	95.7 %	94.0 %	95.5 %	1.1

Tabla 5.16: Tasa de reconocimiento voz del modelo híbrido HMM/MLP OCON. Los HMMs inicializados aleatoriamente.

	Loc 1	Loc 2	Loc 3	Loc 4	Loc 5	Media	Varianza
HMM	96.9 %	97.1 %	97.3 %	96.9 %	92.1 %	96.0 %	2.2
HMM/MLP OCON	96.6 %	98.3 %	96.2 %	96.4 %	95.3 %	96.6 %	1.1

Tabla 5.17: Tasa de reconocimiento de voz del modelo híbrido HMM/MLP OCON. Los HMMs inicializados con el método IOE.

5.6.1.2 Texto manuscrito

Los resultados que aquí presentamos han sido obtenidos con la misma base de datos que en pruebas anteriores, 20 escritores, el fichero de entrenamiento de tamaño 160 y el test con 80 muestras.

Presentamos dos tipos de pruebas de esta estructura híbrida, las cuales utilizan modelos ocultos de Markov diferentes dependientes de la inicialización de los modelos ocultos de Markov:

1. La primera se llevó a cabo con modelos ocultos de Markov inicializados de manera aleatoria. En la tabla 5.18 aparecen los resultados de aplicar el modelo híbrido HMM/MLP OCON.

2. La segunda prueba se utilizó, los modelos optimizados con el método de inicilización de *igual ocupación de estados.* Los resultados de aplicar la estructura híbrida HMM/MLP OCON se encuentran en la tabla 5.19.

Método	Media	Varianza
HMM	92.5 %	7.9
HMM/MLP OCON	98.8 %	2.3

Tabla 5.18: Tasa de reconocimiento de texto manuscrito del modelo híbrido HMM/MLP OCON. Los HMMs inicializados aleatoriamente.

Método	Media	Varianza
HMM	96.6 %	2.8
HMM/MLP OCON	99.2 %	1.6

Tabla 5.19: Tasa de reconocimiento de texto manuscrito del modelo híbrido HMM/MLP OCON. Los HMMs inicializados con el método IOE.

5.6.1.3 Firmas

Los resultados que aquí presentamos han sido obtenidos con la misma base de datos que en pruebas anteriores, 10 firmantes. Cada modelo fue entrenado con 14 muestras por firma y el test se realizó con 10 muestras de firmas.

Presentamos dos tipos de pruebas de esta estructura híbrida, las cuales utilizan modelos ocultos de Markov diferentes dependientes de la inicialización de los modelos ocultos de Markov:

1. La primera se llevó a cabo con modelos ocultos de Markov inicializados de manera aleatoria. En la tabla 5.20 aparecen los resultados de aplicar el modelo híbrido HMM/MLP OCON.

2. La segunda prueba se utilizó, los modelos optimizados con el método de inicilización de *igual ocupación de estados.* Los resultados de aplicar la estructura híbrida HMM/MLP OCON se encuentran en la tabla 5.21.

Método	Media	Varianza
HMM	84.0 %	13.5
HMM/MLP OCON	96.3 %	2.3

Tabla 5.20: Tasa de reconocimiento de firmas del modelo híbrido HMM/MLP OCON. Los HMMs inicializados aleatoriamente.

Método	Media	Varianza
HMM	92.0 %	6.3
HMM/MLP OCON	97.5 %	1.9

Tabla 5.21: Tasa de reconocimiento de firmas del modelo híbrido HMM/MLP OCON. Los HMMs inicializados con el método IOE.

5.7 Híbrido HMM con estructura neuronal modular de redes polinómicas, GMDH

En el capítulo 2 se realizó una breve introducción de este tipo de redes neuronales: estructura, entrenamiento, etc. En este apartado se exponen los resultados de aplicar este tipo de red a la estructura híbrida propuesta. Este tipo de red tiene un único nodo de salida por lo tanto será necesario entrenar tantas redes como clases a clasificar. Cada red GMDH ha sido entrenada con la finalidad de detectar una clase determinada.

La estructura neuronal es una estructura modular formada por diez redes polinómicas GMDH OCON. Cada red tiene la función de detectar una clase. Los nodos que componen la red GMDH realizan una función polinómica de sus entradas del tipo, ver ecuación 5.6.

$$Z_l^k = a_l^k \times \left(Z_j^{k-1}\right)^2 + b_l^k \times \left(Z_i^{k-1}\right) \times \left(Z_j^{k-1}\right) + c_l^k \times \left(Z_i^{k-1}\right)^2 + d_l^k \times Z_j^{k-1} + e_l^k \times Z_i^{k-1} + f_l^k \quad (5.6)$$

El entrenamiento de esta red, trata de buscar los coeficientes $a_l^k, b_l^k, c_l^k, d_l^k, e_l^k y f_l^k$ que mejor se ajusten al problema. Este entrenamiento se realiza tal como se especificó en el capítulo 2, aplicando una serie de modificaciones al algoritmo de entrenamiento:

- La primera modificación se realizó en el fichero de entrenamiento. En [Hecht91] recomienda utilizar distintos ficheros de entrenamiento para la creación de cada capa. Debido a la escacez de datos se utilizó siempre el mismo fichero de entrenamiento.

- La segunda modificación se realizó en el criterio de parada del entrenamiento. En [Hecht91] recomienda que el entrenamiento de la red se finalice, cuando el MSE aumenta y aplica a continuación un procedimiento de poda quedándose con un sólo nodo en la capa de salida. La modificación consiste en parar el entrenamiento cuando al aplicar el procedimiento de poda gana un sólo nodo.

Con estas modificaciones se procede a exponer el entrenamiento de las redes GMDHs. La estructura de la red se define durante el entrenamiento. En cada iteración del entrenamiento se define un nivel con sus nodos. Cuando un nivel es añadido, se evalua el error cometido por todos los nodos para el conjunto de entrenamiento. Antes de construir un nuevo nivel es necesario eliminar los nodos que no aportan información para la clase a clasificar. Para ello es necesario definir un umbral que permita seleccionar aquellos nodos cuyo **MSE** estén por debajo del umbral definido. Éstos serán los nodos ganadores en el proceso de poda. Se trata de un entrenamiento competitivo. Los nodos tienen que competir unos con otros durante la poda. El proceso de construir la red capa a capa, continúa hasta que durante un proceso de poda gane un único nodo. La definición de este umbral implica definir un criterio de parada. Uno de los objetivos que se ha de conseguir con el umbral es que el número de nodos que ganen el proceso de poda disminuya a medida que aumenta el número de capas. Con esto se consigue que la red converja en un único nodo de salida.

Se sabe que a medida que se van añadiendo capas a la red, el número de nodos por nivel aumenta como resultado de la combinación de $M_{nivel-1}$ nodos tomados de dos en dos. Si durante el proceso de poda, el número de nodos por nivel no decrece respecto a la capa anterior, la estructura de la red se desborda y no se consigue la convergencia. La elección del **umbral** es muy importante ya que de ello depende el que la red, converja o no.

Cuando se ha alcanzado un único nodo, se realiza un segundo proceso de poda, eliminar aquellos nodos que no están conectados con el nodo de salida reduciendo enormemente la estructura de la red.

Se ha trabajado buscando un umbral de tal forma que la estructura de la red decrezca a medida que el número de capas aumenta, con el fin de obtener un sólo nodo de salida. Después de numerosas pruebas se obtuvo experimentalmente el umbral definido en la ecuación 5.7.

$$umbral = (Max(MSE) - Min(MSE)) \times 0.03 + Min(MSE) \tag{5.7}$$

Las entradas a la red GMDH son las diez salidas de los modelos ocultos de Markov, una salida por modelo, los cuales dan a su salida la probabilidad de que una secuencia de observación O pertenezca al modelo λ_i, $P(O/\lambda_i)$. El logaritmo de estas probabilidades, $log(P(O/\lambda_i))$, constituyen la entrada de la red GMDH, previamente normalizadas y escaladas entre 0 y 1.

Analicemos el procedimiento de entrenamiento. Esta red utiliza un entrenamiento supervisado, por lo que se debe proporcionar a la red, los pares de entrada-salida. Sea $I_{10\times R}$, el conjunto de entrenamiento de entrada e $Y_{1\times R}$ la salida deseada de la red para cada entrada, los pares de entrada-salida será $i_{10\times j}, y_{1\times j}$. El valor de **R** hace referencia al tamaño del fichero de entrenamiento. El subíndice 10 de la entrada i, hace referencia al número de entradas a la red, y el subíndice 1 de la variable y, al número de salidas de la red.

La primera capa se forma con la combinación de las 10 entradas tomadas de 2 en 2, $(C(10,2))$, dando como resultado 45 nodos o elementos de procesos para la primer nivel. El siguiente paso será calcular los coeficientes a_l^1, b_l^1, c_l^1, d_l^1, e_l^1, y f_l^1 para los 45 nodos, $l = 1, 2, 3, ...45$. En la ecuación 5.8, aparece la notación matricial de la función de transferencia de un nodo. Siendo la matriz A_l^1, la matriz de coeficientes.

$$Z_l^1 = \begin{bmatrix} \left(Z_j^0\right)^2 & \left(Z_i^0\right) \times \left(Z_j^0\right) & \left(Z_i^0\right)^2 & Z_j^0 & Z_i^0 & 1 \end{bmatrix} \times \begin{bmatrix} a_l^1 \\ b_l^1 \\ c_l^1 \\ d_l^1 \\ e_l^1 \\ f_l^1 \end{bmatrix} = X_l \cdot A_l^1 \qquad (5.8)$$

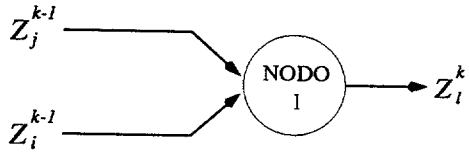

Figura 5.7: Nodo o elemento de proceso

El objetivo es encontrar el conjunto de coeficientes o matriz A_l^1 que satisfacen la ecuación 5.8, para cada uno de los nodos. Éstos se calculan durante la fase de entrenamiento.

Como se ha comentado ya, el entrenamiento de la red es supervisado. Los coeficientes de entrada se calculan a partir de la salida que deberían dar los nodos, esto es, de la ecuación 5.8 se sustituye el valor de Z_l^1 por el valor $y_{1\times j}$. Éste puede ser un valor real o bien una señal binaria 0 ó 1. Todos los nodos de una misma capa son entrenados para dar la misma salida, lo más cercana al valor de $y_{1\times r}$.

La matriz A_l^1 se calculan para el conjunto de entrenamiento, es decir, los datos, tanto de entrada como de salida, son presentados a la red en bloque. El conjunto de ecuaciones no tendrá una solución exacta sino aproximada con el fin de obtener un conjunto de coeficientes que minimicen el MSE (Mean Square Error). Se ha probado que cuando la matriz A_l^k se calcula a partir de la pseudoinversa de X_l, el MSE es mínimo.

Ejemplo

Veamos a continuación un ejemplo del principio de funcionamiento de la red GMDH. Se sabe que hay tantas redes como clases o dígitos a clasificar. Se analizará el caso particular del **dígito 0**, es decir la red GMDH 0. Las entradas a la red son las salidas de los 10 modelos ocultos de Markov. Combinando las 10 entradas de 2 en 2 y como ya se ha adelantado en apartados anteriores originan la formación de 45 nodos. Es lógico pensar que tratándose de la red **GMDH 0**, sea la probabilidad del modelo oculto de Markov para el dígito 0, $P(0/HMM_0)$, la que teóricamente sea más alta salvo en aquellos casos donde los HMMs comenten un error de clasificación. Por lo tanto, aquellos nodos donde participa la variable, $P(0/HMM_0)$, son los que deberían dar a su salida un valor próximo a la unidad y en aquellos nodos donde no participa, probablemente dará un valor no tan cercano a la unidad.

En la figura 5.8 se observa el error cuadrático medio cometido por los 45 nodos de la primera capa. Se observa que son los primeros nodos los que comenten el error más bajo debido a que una de sus entradas es justamente la $P(0/HMM_0)$, ya que se combinan de la siguiente manera:

- Nodo 1: entradas $P(0/HMM0)$ y $P(0/HMM1)$
- Nodo 2: entradas $P(0/HMM0)$ y $P(0/HMM2)$
- ...
- Nodo 9: entradas $P(0/HMM0)$ y $P(0/HMM9)$
- ...
- Nodo 45: entradas $P(0/HMM8)$ y $P(0/HMM9)$

Una vez medido el MSE cometido por cada nodo, se procede a seleccionar aquellos nodos que comenten el error más bajo.

Evaluando el resultado del MSE para este caso, figura 5.8, se observa que los nodos que presentan un MSE pequeño, son los nodos del 1 al 9. Tiene sentido que esto ocurra así, ya que se está evaluando el dígito 0 y es importante la contribución de la $P(0|HMM_0)$ en aquellos nodos donde participe, precisamente, los nodos numerados del 1 al 9 de la figura 5.8.

Una vez evaluado el error que comete cada nodo, se pasa al siguiente paso, la **poda**. Para ello se ha de fijar un umbral que seleccione aquellos nodos que menor MSE presenten. El calculado y obtenido bajo la experiencia para este trabajo es el dado en la ecuación 5.7. Los nodos ganadores son: 3, 4, 5, 6, 7, y 8. El resto son eliminados.

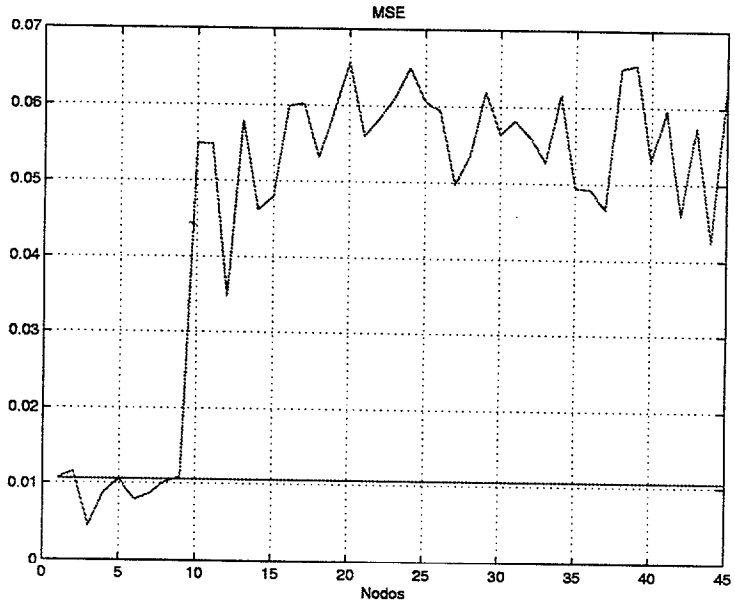

Figura 5.8: Error Cuadrático medio por nodos, capa 1

Con los 6 nodos ganadores de este primer nivel, se pasa a la formación del segundo nivel con 15 nodos, resultado de aplicar la combinación de 6 nodos tomadas de 2 en 2, $C(6,2)$. En la figura 5.9 aparece el MSE para el nivel 2. En este caso son los nodos numerados del 1 al 5 los que presentan un MSE por debajo del umbral.

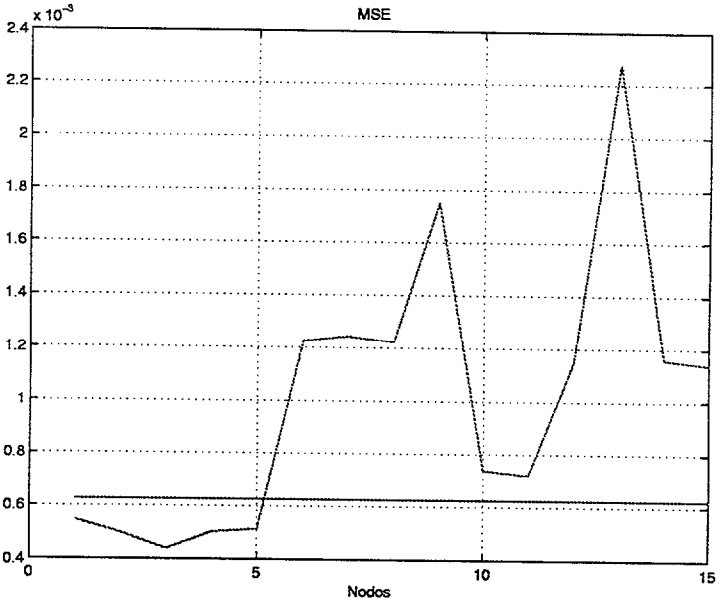

Figura 5.9: MSE para la segunda capa

El proceso de poda es similar para esta segundo nivel, se aplicará el umbral y todos los nodos cuyo MSE estén por encima serán eliminados. Para el ejemplo que tratamos

son los 5 primeros nodos los ganadores, que pasarán formar parte del tercer nivel.

Esta nueva capa estará formada por la combinación de 5 nodos tomados de 2 en 2, C(5,2), dando como resultado 10 nodos, de los cuales, según la gráfica 5.10 resulta el nodo 9 el que comete menos error. Este nodo es el que en definitiva clasificará al dígito **0**.

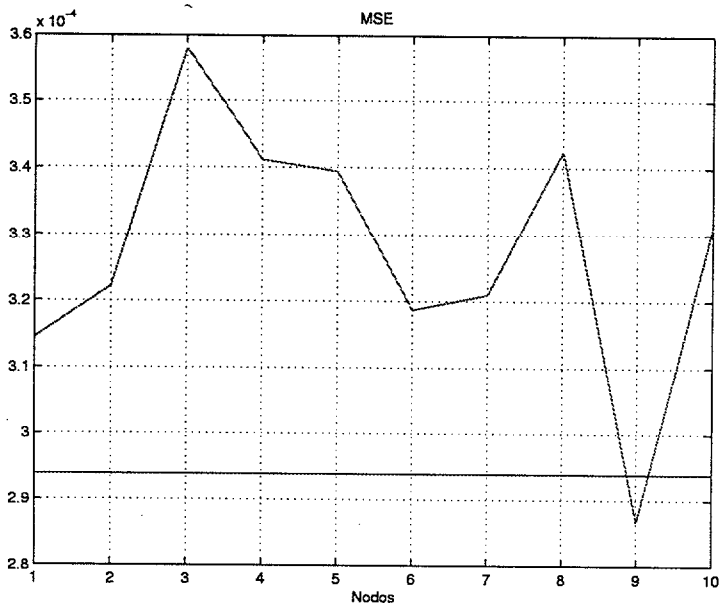

Figura 5.10: MSE para la tercera capa

Una vez obtenido el único nodo de salida se procede a una última poda eliminando aquellos nodos que no intervienen en el nodo ganador. En la figura 5.11 aparece la estructura de la red y sus dependencias con otras entradas.

Hasta el momento y para todas las pruebas realizadas, la red GMDH con el umbral definido en 5.7 tiende a decrecer en su estructura y finalizar el entrenamiento con 3 y en muy contados casos en 4 niveles.

5.7.1 Experimentos y resultados

5.7.1.1 Señal de voz

Los resultados que aquí presentamos han sido obtenidos con la base de datos de 5 locutores con 132 realizaciones. La secuencia de test son las 100 primeras realizaciones y las 32 restantes para entrenamiento.

Presentamos dos tipos de pruebas de esta estructura híbrida, las cuales utilizan modelos ocultos de Markov diferentes dependientes de la inicialización de los modelos:

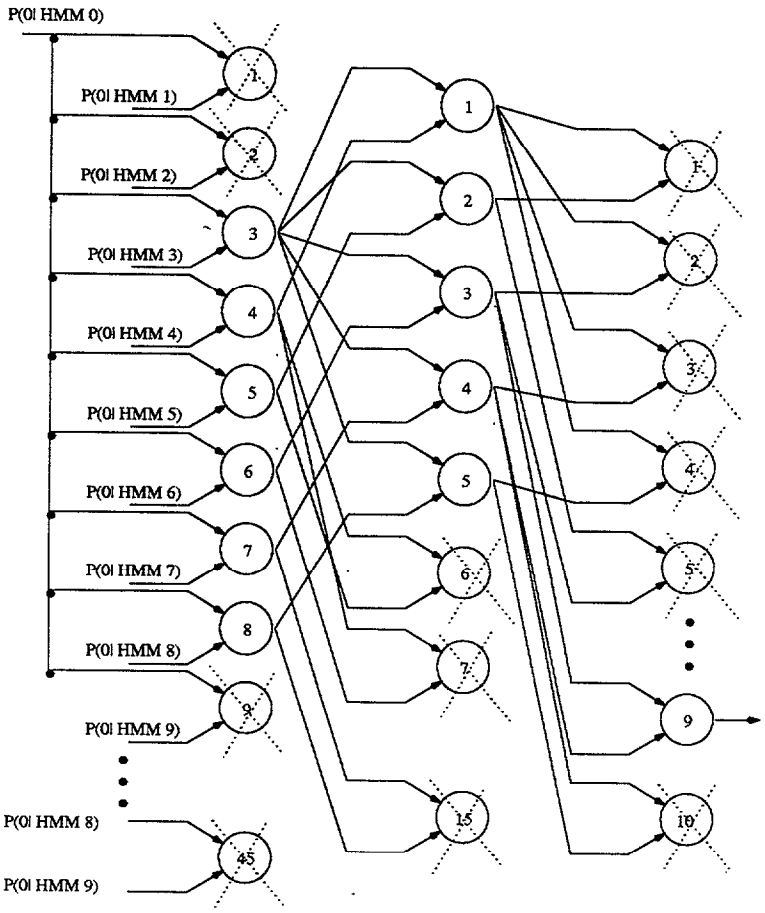

Figura 5.11: Estructura de la red GMDH para el dígito 0

1. La primera se llevó a cabo con modelos ocultos de Markov inicializados de manera aleatoria. En la tabla 5.22 aparecen los resultados de aplicar el modelo híbrido HMM/GMDH. Estos resultados son los valores promedios de entrenar 10 estructuras híbridas.

2. La segunda prueba se utilizó, los modelos optimizados con el método de inicilización de *igual ocupación de estados.* Recordemos que con este tipo de inicialización se obtuvieron modelos con la tasa de reconocimiento más alta. Los resultados de aplicar la estructura híbrida HMM/GMDH se encuentran en la tabla 5.23. Estos resultados son los valores promedios de entrenar 10 estructuras híbridas.

5.7.1.2 Texto manuscrito

Los resultados que aquí presentamos han sido obtenidos con la misma base de datos que en pruebas anteriores, 20 escritores, el fichero de entrenamiento es de tamaño 160 y el test de 80 muestras.

Método	Loc 1	Loc 2	Loc 3	Loc 4	Loc 5	Media	Varianza
HMM	94.6 %	93.2 %	96.9 %	91.4%	91.8%	93.6 %	2.3
HMM/GMDH	95.4 %	95.9 %	95.3 %	93.9 %	90.0 %	94.1 %	2.4

Tabla 5.22: Tasa de reconocimiento de voz del modelo híbrido HMM/GMDH. Los HMMs están inicilalizados aleatoriamente.

Método	Loc 1	Loc 2	Loc 3	Loc 4	Loc 5	Media	Varianza
HMM	96.9 %	97.1 %	97.3 %	96.9 %	92.1 %	96.0 %	2.3
HMM/GMDH	97.9 %	98.8 %	97.7%	98.3 %	95.2 %	97.6 %	1.4

Tabla 5.23: Tasa de reconocimiento de voz del modelo híbrido HMM/GMDH. Los HMMs han sido inicializados con el método IOE

Presentamos dos tipos de pruebas de esta estructura híbrida, las cuales utilizan modelos ocultos de Markov diferentes dependientes de la inicialización de los modelos:

1. La primera se llevó a cabo con modelos ocultos de Markov inicializados de manera aleatoria. En la tabla 5.24 aparecen los resultados de aplicar el modelo híbrido HMM/GMDH.
2. La segunda prueba se utilizó, los modelos optimizados con el método de inicilización de *igual ocupación de estados*. Los resultados de aplicar la estructura híbrida HMM/GMDH se encuentran en la tabla 5.25.

Método	Media	Varianza
HMM	92.5 %	7.9
HMM/GMDH	94.5 %	5.8

Tabla 5.24: Tasa de reconocimiento de texto manuscrito del modelo híbrido HMM/GMDH. Los HMMs inicializados aleatoriamente.

5.7.1.3 Firmas

Los resultados que aquí presentamos han sido obtenidos con la misma base de datos que en pruebas anteriores, 10 firmantes con 24 repeticiones cada una.

Método	Media	Varianza
HMM	96.6 %	2.8
HMM/GMDH	97.0 %	2.3

Tabla 5.25: Tasa de reconocimiento de texto manuscrito del modelo híbrido HMM/GMDH. Los HMMs inicializados con el método IOE.

Presentamos dos tipos de pruebas de esta estructura híbrida, las cuales utilizan modelos ocultos de Markov diferentes dependientes de la inicialización de los modelos:

1. La primera se llevó a cabo con modelos ocultos de Markov inicializados de manera aleatoria. En la tabla 5.26 aparecen los resultados de aplicar el modelo híbrido HMM/GMDH.

2. La segunda prueba se utilizó, los modelos optimizados con el método de inicilización de *igual ocupación de estados.* Los resultados de aplicar la estructura híbrida HMM/GMDH se encuentran en la tabla 5.27.

Método	Media	Varianza
HMM	84.0 %	13.5
HMM/GMDH	92.3 %	9.8

Tabla 5.26: Tasa de reconocimiento de firmas del modelo híbrido HMM/GMDH. Los HMMs inicializados aleatoriamente.

Método	Media	Varianza
HMM	92.0 %	6.3
HMM/GMDH	93.2 %	5.4

Tabla 5.27: Tasa de reconocimiento de firmas del modelo híbrido HMM/GMDH. Los HMMs inicializados con el método IOE.

5.8 Híbrido HMM con estructura neuronal polinómica

Otra posibilidad de modelo híbrido, es la de utilizar una red de interconexión basado en la combinación de múltiples redes GMDH de estructura tan sencilla que sólo están formadas

por un único nodo. La función de transferencia es la misma que la que utilizan los nodos GMDH, una función polinómica de sus entradas, ver ecuación 5.9.

$$Z_l^1 = \left[\left(Z_j^0\right)^2 \quad \left(Z_i^0\right) \times \left(Z_j^0\right) \quad \left(Z_i^0\right)^2 \quad Z_j^0 \quad Z_i^0 \quad 1 \right] \times \begin{bmatrix} a_l^1 \\ b_l^1 \\ c_l^1 \\ d_l^1 \\ e_l^1 \\ f_l^1 \end{bmatrix} = X_l \cdot A_l^1 \qquad (5.9)$$

Como se ha podido ver, el utilizar una estructura neuronal con redes GMDH en el híbrido implica emplear tantas redes como clases se desee clasificar. Con esta propuesta de utilizar mútiples redes GMDH, muy sencillas, se aprovecha por un lado las capacidades de las redes polinómica, y por otro lado, se reemplaza las 10 redes GMDH del híbrido anterior por una estructura relativamente más sencilla. Esto nos conduce a un cálculo computacional más bajo. Se trata de una alternativa novedosa de estructura neuronal basada en redes GMDH.

La estructura neuronal está formada por múltiples redes GMDH de estructura tan sencilla que sólo están formadas por un nodo o elemento de proceso. Cada red o nodo trabaja de manera independiente y paralela. Se puede decir que estos nodos o simples redes se encuentran distribuidos en una única capa.

Se ha comentado que cada nodo o elemento de proceso tiene dos entradas y una salida, ver figura 5.12. Las señales, que se desean evaluar son las 10 salidas procedentes de los modelos ocultos de Markov. Es necesario aplicar el mismo procedimiento de formación de capas de un red GMDH general, es decir combinar estas señales de 2 en 2, por lo que se necesita un total de 45 nodos para examinar la combinación de todos los pares posibles de entradas.

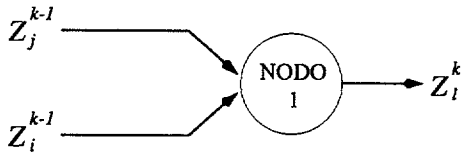

Figura 5.12: Nodo o elemento de proceso

En la figura 5.13 aparece la estructura o distribución de las sencillas redes GMDH.

La principal diferencia con la red GMDH clásica está en el entrenamiento de estas redes. A pesar de que la formación de esta red de interconexión es similar a la formación de una capa de una red GMDH. En esta estructura, cada nodo es considerado como una red GMDH. Y el entrenamiento de cada nodo es independiente al de sus nodos o redes vecinas. Cada red o nodo es entrenada para dar una respuesta diferente a los nodos

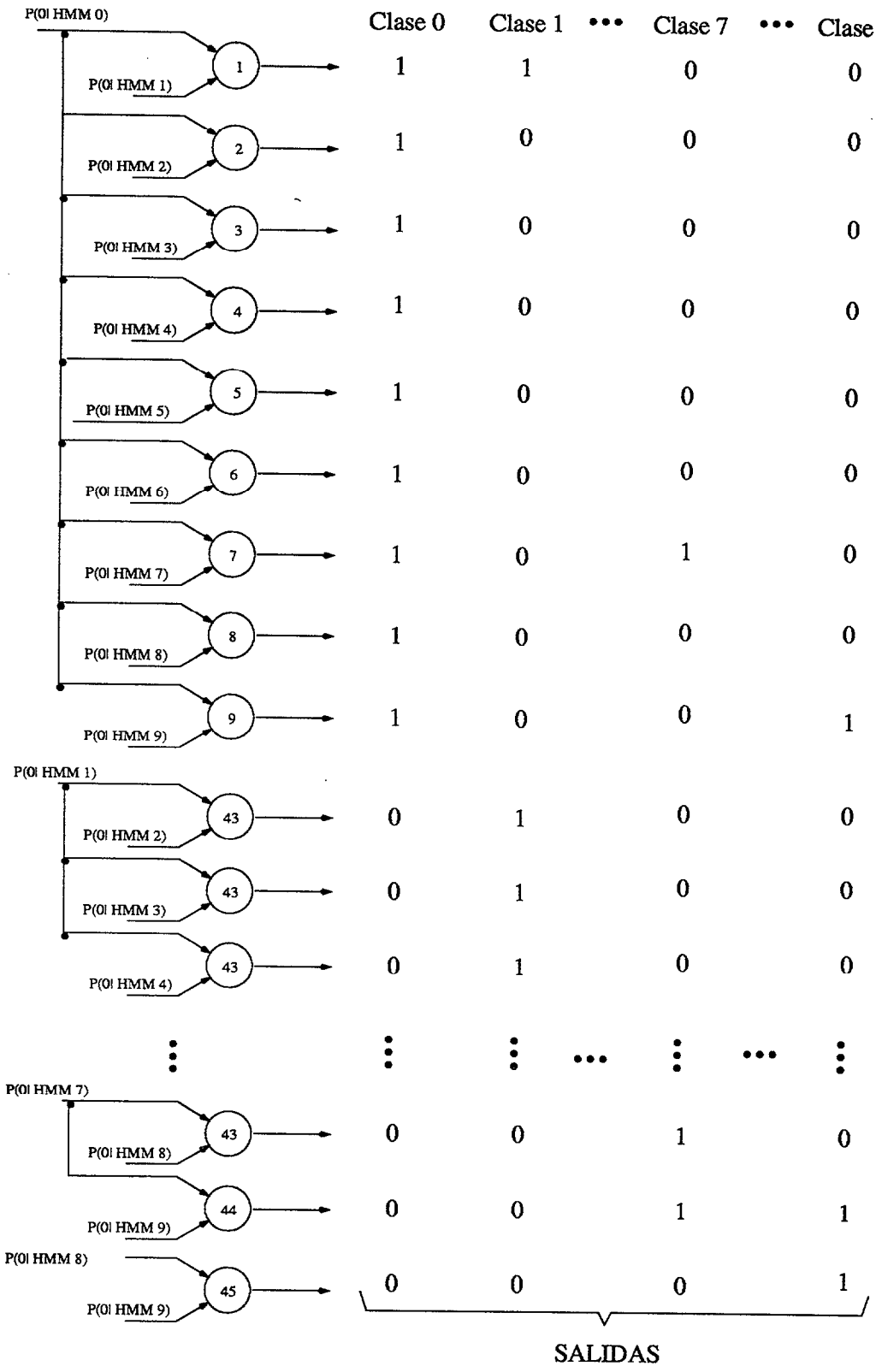

Figura 5.13: Estructura de la Red GMDH

restantes y no tienen que competir unos con otros en un proceso de poda como ocurría en la red GMDH clásica. Esta red de interconexión polinómica se entrena de tal forma que haga corresponder un patrón de entrada con uno de salida formado por la salida de cada red o nodo.

El método de entrenamiento utilizado es el siguiente, cuando un patrón de entrada se corresponde con un dígito x, todos aquellos nodos donde participa, como entrada la probabilidad de este dígito, $P(X|\lambda_x)$, tendrá a su salida un 1 y el resto de los nodos estarán a 0. En la figura 5.13, se observa la estructura de esta red de interconexión, con algunos patrones de salida.

A esta estructura neuronal polinómica la hemos denominado Red GMDH.

5.8.1 Resultados y experimentos

5.8.1.1 Señal de voz

Los resultados que aquí presentamos han sido obtenidos con la base de datos de 5 locutores con 132 realizaciones. La secuencia de test son las 100 primeras realizaciones y las 32 restantes para entrenamiento. Una comparación de los resultados obtenidos con el sistema HMM se tienen en las tablas 5.28 y 5.29. En ambas tablas se representan los valores promedios de entrenar 10 sistemas híbridos. En la primera tabla se ha trabajado con los modelos ocultos de Markov inicializados de manera aleatoria. La segunda tabla, 5.29 se compara los resultados con los modelos ocultos de Markov inicializados con el método de igual ocupación de estados (IOE).

Método	Loc 1	Loc 2	Loc 3	Loc 4	Loc 5	Media	Varianza
HMM	94.6 %	93.2 %	96.9 %	91.4%	91.8%	93.6 %	2.3
HMM/Red GMDH	96.5 %	95.0 %	95.0 %	96.3 %	95.7 %	95.7 %	0.7

Tabla 5.28: Tasa de reconocimiento de voz del modelo híbrido HMM/Red GMDH. Los HMMs están inicilalizados aleatoriamente.

5.8.1.2 Texto manuscrito

Los resultados que aquí presentamos han sido obtenidos con la misma base de datos que en pruebas anteriores, 20 escritores, el fichero de entrenamiento de tamaño 160 y el test con 80 muestras.

Método	Loc 1	Loc 2	Loc 3	Loc 4	Loc 5	Media	Varianza
HMM	96.9 %	97.1 %	97.3 %	96.9 %	92.1 %	96.0 %	2.3
HMM/Red GMDH	97.8 %	96.4 %	95.8 %	97.2 %	95.2 %	96.4 %	0.9

Tabla 5.29: Tasa de reconocimiento del modelo híbrido HMM/Red GMDH. Los HMMs han sido inicializados con el método IOE

Presentamos dos tipos de pruebas de esta estructura híbrida, las cuales utilizan modelos ocultos de Markov diferentes dependientes de la inicialización de los modelos ocultos de Markov:

1. La primera se llevó a cabo con modelos ocultos de Markov inicializados de manera aleatoria. En la tabla 5.30 aparecen los resultados de aplicar el modelo híbrido HMM/Red GMDH.
2. La segunda prueba se utilizó, los modelos optimizados con el método de inicilización de *igual ocupación de estados*. Los resultados de aplicar la estructura híbrida HMM/Red GMDH se encuentran en la tabla 5.31.

Método	Media	Varianza
HMM	92.5 %	7.9
HMM/Red GMDH	93.8 %	6.9

Tabla 5.30: Tasa de reconocimiento de texto manuscrito del modelo híbrido HMM/Red GMDH. Los HMMs inicializados aleatoriamente.

Método	Media	Varianza
HMM	96.6 %	2.8
HMM/Red GMDH	98.0 %	1.6

Tabla 5.31: Tasa de reconocimiento de texto manuscrito del modelo híbrido HMM/Red GMDH. Los HMMs inicializados con el método IOE.

5.8.1.3 Firmas

Los resultados que aquí presentamos han sido obtenidos con la misma base de datos que en pruebas anteriores, 10 firmantes, el fichero de entrenamiento utiliza de tamaño 160 y el test con 80 muestras.

Presentamos dos tipos de pruebas de esta estructura híbrida, las cuales utilizan modelos ocultos de Markov diferentes dependientes de la inicialización de los modelos ocultos de Markov:

1. La primera se llevó a cabo con modelos ocultos de Markov inicializados de manera aleatoria. En la tabla 5.32 aparecen los resultados de aplicar el modelo híbrido HMM/Red GMDH.
2. La segunda prueba se utilizó, los modelos optimizados con el método de inicialización de *igual ocupación de estados*. Los resultados de aplicar la estructura híbrida HMM/Red GMDH se encuentran en la tabla 5.33.

Método	Media	Varianza
HMM	84.0 %	13.5
HMM/Red GMDH	86.5 %	9.5

Tabla 5.32: Tasa de reconocimiento de firmas del modelo híbrido HMM/Red GMDH. Los HMMs inicializados aleatoriamente.

Método	Media	Varianza
HMM	92.0 %	6.3
HMM/Red GMDH	95.8 %	3.6

Tabla 5.33: Tasa de reconocimiento de firmas del modelo híbrido HMM/Red GMDH. Los HMMs inicializados con el método IOE.

5.9 Combinación de HMMs con el método de *voting model*

En este apartado queremos recoger los resultados de la combinación de varios modelos para dar una respuesta. Ya en el capítulo 4, se adelantaba de un método de inicialización con resultados muy buenos, utilizar varios modelos inicializados con diferentes semillas y utilizar todos los modelos en la fase de reconocimiento. También puede ser visto como un comité de HMMs.

La clase ganadora es la más votada. Se trata de otra forma de clasificar, en la que un conjunto o comité de modelos ocultos de Markov participan en la decisión final.

Lo importante cuando se utiliza este tipo de estructuras es que los modelos estén incorrelados. Es decir, los modelos no deben comportarse de la misma manera para todas las entradas. Si esto sucede, la estructura pierde su objetivo, aumentar la capacidad de generalización.

5.9.1 Experimentos y resultados

En este caso sólo se han realizado los cálculos para el reconocimiento de voz.

5.9.1.1 Señal de voz

Los resultados que aquí presentamos han sido obtenidos con la base de datos de 5 locutores con 132 realizaciones. La secuencia de test son las 100 primeras realizaciones y las 32 restantes para entrenamiento.

Al igual que en los experimentos anteriores, presentamos dos tipos de pruebas de este comité de HMMs, dependientes de la inicialización de los modelos ocultos de Markov. En las tablas 5.34 y 5.35 aparecen los resultados cuando intervienen varias modelos HMM, hasta un máximo de 10, en la toma de decisión final. La primera fila de ambas tablas son los valores promedios de entrenar 10 modelos ocultos de Markov:

1. La primera se llevó a cabo con modelos ocultos de Markov inicializados de manera aleatoria. Los resultados se recogen en la tabla 5.34.
2. La segunda prueba se utilizó, aplicando a los modelos el método de inicilización de *igual ocupación de estados.* Los resultados se recogen en la tabla 5.35.

De ambas tablas se deduce que en valores promedios en media la tasa de reconocimiento aumenta a medida que se incorporan los modelos HMMs en la toma de decisión final. Y la varianza también disminuye a medida que aumenta el número de los modelos HMMs. Si analizamos los datos de manera particular por locutor podemos observar que este efecto no se mantiene, por ejemplo, el locutor 1, cuando se emplean 6 u 8 modelos en la toma de decisión, la tasa de reconocimiento sufre una pequeña disminución respecto al sistema que utiliza 5 modelos. Esto puede suceder por empate de modelos a la hora de elegir la clase. Seleccionando la clase no correcta.

N modelos	Loc 1	Loc 2	Loc 3	Loc 4	Loc 5	Media	Varianza
HMM	94.6 %	93.2 %	96.9 %	91.4%	91.8%	93.6 %	2.3
2 HMMs	97.1 %	92.2 %	98.1 %	96.9 %	93.1 %	95.5 %	2.6
3 HMMs	97.0 %	95.1 %	97.2 %	97.3 %	92.2 %	95.8 %	2.2
4 HMMs	97.3 %	98.1 %	97.9 %	97.4 %	91.6 %	96.5 %	2.7
5 HMMs	97.8 %	98.0 %	98.2 %	97.4 %	94.6 %	97.2 %	1.5
6 HMMs	97.5 %	98.1 %	98.3 %	97.5 %	95.5 %	97.4 %	1.1
7 HMMs	97.8 %	97.8 %	98.4 %	97.7 %	95.5 %	97.4 %	1.1
8 HMMs	97.7 %	98.2 %	98.9 %	97.4 %	95.5 %	97.6 %	1.1
9 HMMs	97.7 %	98.2 %	98.9 %	97.4 %	95.9 %	97.6 %	1.1
10 HMMs	97.3 %	97.9 %	98.5 %	97.5 %	95.9 %	97.4 %	1.0

Tabla 5.34: Tasa de reconocimiento del comité de modelos HMMs inicilalizados aleatoriamente.

N Modelos	Loc 1	Loc 2	Loc 3	Loc 4	Loc 5	Media	Varianza
HMM	94.6 %	93.2 %	96.9 %	91.4%	91.8%	93.6 %	2.3
2 HMMs	97.1 %	97.2 %	98.4 %	97.3 %	94.2 %	96.8 %	1.6
3 HMMs	97.2 %	97.8 %	98.5 %	97.2 %	94.1 %	97.0 %	1.7
4 HMMs	97.5 %	97.5 %	98.8 %	97.8 %	95.4 %	97.4 %	1.2
5 HMMs	97.1 %	97.8 %	98.5 %	97.8 %	94.5 %	97.1 %	1.6
6 HMMs	97.6 %	97.8 %	98.4 %	97.8 %	95.5 %	97.4 %	1.1
7 HMMs	97.7 %	97.7 %	98.5 %	97.8 %	96.2 %	97.6 %	0.8
8 HMMs	97.7 %	97.8 %	98.3 %	97.8 %	96.1 %	97.5 %	0.8
9 HMMs	97.8 %	97.8 %	98.3 %	97.9 %	95.3 %	97.4 %	1.2
10 HMMs	97.6 %	98.3 %	98.4 %	97.9 %	94.8 %	97.4 %	1.5

Tabla 5.35: Tasa de reconocimiento de voz del comité de modelos HMMs inicilalizados con el método IOE.

5.10 Estudio del cálculo computacional/tasa de reconocimiento

El estudio se ha realizado únicamente para el reconocimiento de voz.

Método	Media	Varianza	Entrenamiento HMM+NN (Flops)	Test HMM+NN (Flops)
HMM/MLP ACON	97.5 %	1.3	1.1875e+010	9.2698e+6
HMM/MLP OCON	95.5 %	1.1	7.85792e+8	9.8773e+6
HMM/GMDH	94.1 %	2.4	2.9609e+9	9.8428e+5
HMM/Red GMDH	95.7 %	0.7	1.16355e+8	1.57707e+6

Tabla 5.36: Estudio comparativo de los distintos modelos híbridos. Los modelos HMMs están inicilalizados aleatoriamente.

	Media	Varianza	Entrenamiento HMM+NN (Flops)	Test HMM+NN (Flops)
HMM/MLP ACON	97.6 %	1.2	1.3271e+10	9.49969e+6
HMM/MLP OCON	96.6 %	1.1	4.60482e+8	9.3845e+6
HMM/GMDH	97.6 %	1.4	2.54601e+9	8.3804e+5
HMM/Red GMDH	96.4 %	0.9	1.16351e+8	1.57707e+6

Tabla 5.37: Estudio comparativo de los distintos modelos híbridos. Los modelos HMMs están inicilalizados con el método IOE.

De las tablas 5.36 y 5.37 se observa que para la estructura HMM/MLP ACON es la que invierte más cálculo computacional que el resto de los híbridos, también consigue la tasa de reconocimiento más alta. La estructura HMM/Red GMDH es la que menos carga computacional emplea en su entrenamiento. El híbrido HMM/GMDH también invierte una gran carga computacional durante el entrenamiento sin embargo en la fase de test, es el que menos carga computacional emplea. Ésto se debe a la estructura optimizada durante el entrenamiento. En cuanto a los resultados de la tasa de reconocimiento es similar entre las distintas estructuras.

Capítulo 6

Conclusiones y Líneas Futuras

6.1 Conclusiones

El comienzo de esta tesis estuvo enfocado hacia el reconocimiento de voz, estudiando la estructura óptima de los modelos. También se estudió problemas tales como: inicialización, los criterios de parada, conjuntos de entrenamiento, etc. De esta primera parte resumimos, que a la hora de trabajar y abordar un problema, es conveniente buscar la estructura del modelo más idónea antes de comenzar a realizar las distintas pruebas y experimentos. A veces no es suficiente buscar la estructura óptima sino estudiar cómo influyen otros parámetros en el comportamiento de un sistema. Se ha comprobado que utilizando una inicialización adecuada, aplicando criterios de parada, y eligiendo conjuntos de entrenamiento adecuados, se puede conseguir resultados satisfactorios con un coste adicional en cuanto al cálculo computacional.

Se ha probado que dentro de los métodos de inicialización estudiados, es el método de *inicialización con igual ocupación de estados (IOE)*, con el que se ha obtenido modelos iniciales, que convergen en el entrenamiento, en modelos mejores que los obtenidos con la aplicación de métodos de inicialización tradicionales.

Por otro lado se ha comprobado que el método de validación cruzada para el reconocimiento de voz y texto manuscrito, puede ser una alternativa cuando se dispone de conjuntos de datos reducidos. Y que combinados con el método de inicialización de *igual ocupación de estados* dan resultados satisfactorios. Se probó que al utilizar este método interesa, que los conjuntos de entrenamiento y validación vayan cambiando su composición a medida que se entrena.

La siguiente línea de trabajo de esta tesis está centrada en el estudio de estructuras

híbridas. Éstos presentan una característica muy importante y es que varios métodos o sistemas participan en la decisión final, enriqueciendo en gran medida el comportamiento de un clasificador. Como ya se ha visto, un modelo híbrido puede tener un comportamiento modular o de comité. Los primeros descomponen la tarea entre los distintos modelos y los segundos agrupan o combinan varios modelos diferentes en cuanto a inicialización, estructuras, entrenamientos, etc., a la hora de tomar la decisión.

Los modelos híbridos son métodos, que tanto trabajando de manera modular como en comité, aprovechan al máximo las capacidades de cada modelo. En los primeros, cada sistema tiene una tarea asignada y ésto mejora el rendimiento de cada uno de ellos. Y los segundos son capaces de corregir errores en la etapa de decisión, que de una manera democrática eligen la clase más votada.

A partir de los primeros experimentos llevados a cabo sobre el sistema híbrido propuesto, se concluyó en el hecho, de que no se puede utilizar el mismo fichero de entrenamiento para los modelos ocultos de Markov, y para las redes neuronales[1]. Ésto se debe a que puede ocurrir que los modelos HMMs tenga una tasa de reconocimiento del 100 %, (para el fichero de entrenamiento), y la red no puede aprender a corregir errores cometidos por los HMMs, porque carece de esa información en el fichero de entrenamiento. Es por lo que, el sistema híbrido no puede dar resultados mejores que los HMMs. No se consigue mejoría alguna con la estructura híbrida. Para solventar este problema se ha de preparar cuidadosamente el fichero de entrenamiento de las redes neuronales. La solución adoptada en las pruebas presentadas en esta tesis fue, la de añadir errores intencionadamente en dichos ficheros, para conseguir una mejora, y así la red pueda dar respuestas adecuadas ante entradas con errores y no vistas.

Otra de las conclusiones obtenidas a la hora de trabajar con la estructura híbrida propuesta, es que cuando los modelos ocultos de Markov funcionan muy bien, la mejoría introducida por la red neuronal es menor que para modelos con tasas de reconocimiento inferiores. La diferencia cuantitativa puede ser aproximadamente de 4 puntos. Es decir, para modelos con una tasa de reconocimiento, del 93.6 %, resultado obtenido con los modelos inicializados aleatoriamente, se consigue un mejora de casi 4 puntos con la unión en cascada de una MLP. Sin embargo, si empleásemos la misma estructura híbrida pero con modelos optimizados, con una inicialización adecuada, la mejoría introducida puede ser sólo de medio punto con la unión en cascada de las redes neuronales. Ésto conduce a estudiar, qué método compesa más, optimización de modelos o aplicación de estructuras híbridas, a nivel del carga computacional y tiempo intertido en el entrenamiento y en la fase de test de ambos métodos.

La razón más importante a la hora de combinar y formar nuevas estructuras es que un sistema, cuando es entrenado para dar una determinada salida puede cometer errores frente a determinadas entradas debido a que el conjunto de entrenamiento es limitado. La combinación de un conjunto de estimadores imperfectos puede pensarse como una forma

[1]Los datos de entrenamiento de las redes neuronales son los tomados a la salida de los HMMs

de manejar las limitaciones de cada estimador por separado. Es decir, cada estimador comete sus errores pero combinados de una determinada manera, se pueden minimizar. Es evidente, que no se obtienen ventajas, con un comité formado por sistemas idénticos ya que todos generalizan de la misma manera, cometen los mismos errores. En principio, los sistemas pueden ser diferentes en cuanto a estructura, parámetros, inicialización, etc. pero si comenten los mismos errores no estamos mejorando el sistema, simplemente añadiendo más carga computacional. Lo interesante de la combinación de estructuras neuronales es encontrar modelos o sistemas que generalicen de manera diferente, (que los errores que comente cada sistema estén incorrelados). De esta manera cada red neuronal aporta información diferente a la hora de clasificar. Por supuesto, este tipo de estructuras lleva un carga computacional más elevada que la configuración con una sóla red.

Un problema que se encontró al trabajar con la combinación de múltiples redes, es la elección del número de redes que intervienen. No siempre se consigue resultados mejores al aumentar el número de modelos porque se puede producir una situación de empate de clases, eligiéndose la clase errónea. Se recomienda que el número de redes utilizadas sea superior a 2 para evitar esta situación. Sin embargo cuando el número de redes que participan en la decisión final es impar e igual a 3, esta situación puede no aparecer, eligiéndose la clase más votada, a no ser que las tres redes tomen decisiones diferentes. Cuando el número de redes aumenta, el que se produzca una situación de empate puede ocurrir, tanto para estructuras con un número par o impar de redes. Por supuesto, con un número impar, la probabilidad puede ser menor, si el número de redes es superior al número de clases.

Otra de las conclusiones obtenidas en esta tesis fue la utilización de redes polinómicas en estructuras híbridas con resultados satisfactorios. El sistema híbrido HMM/GMDH es una estructura más compleja que el híbrido HMM/MLP, ya que se necesita de tantas redes GMDH como clases a clasificar. La ventaja de las redes GMDH, es que crean su propia estructura durante el entrenamiento, por lo tanto están optimizadas para la aplicación.

Por otro lado, el sistema híbrido formado por HMM/Red GMDH es una estructura única, donde las redes GMDH son tan sencillas que están formadas por nodos o elementos de proceso GMDH. Esta estructura única, se forma con tantos nodos como combinaciones tomadas de dos en dos, de las señales a procesar, su estructura es fija y depende únicamente del número de entradas. Cada nodo da una respuesta independiente al resto de los nodos. Siendo esto una ventaja ya que varios nodos participan en la decisión final de clasificación y ésto hace que se obtengan resultados aceptables con esta estructura.

Por otro lado el coste computacional de este tipo de redes polinómicas es inferior al del híbrido HMM/MLP con tasas de reconocimiento similares.

Resumiendo cuando se aborda un problema de clasificación de secuencias, y se va a emplear los modelos HMMs, es importante buscar la estructura más adeacuada a la aplicación. Con una buena inicialización de los modelos, aplicar criterios de parara adecuados, etc. se puede conseguir resultados buenos sin necesidad de aplicar reentrenamientos, con

el fin de conseguir modelos óptimos. Si se desea obtener modelos a partir de conjuntos de datos reducidos, es conveniente estudiar la posibilidad de aplicar técnicas de validación cruzada combinadas con los métodos de inicialización y criterios de parada.

Por otro lado, cuando determinadas estructuras adolecen de ciertas limitaciones pero presentan otras ventajas, como el caso de los HMMs que presentan poca capacidad de discriminación y sin embargo modelan el comportamiento temporal de una señal o las redes neuronales que son estructuras con una gran habilidad de clasificación pero son menos eficientes a la hora de trabajar con secuencias temporales o con datos secuenciales, es posible la combinación de ambas estructuras con el fin de obtener las ventajas de ambos modelos.

La última comprobación de nuestro estudio fue la de experimentar con otro tipo de señales diferentes a la de voz. Para ello se aplicaron las distintas pruebas al reconocimiento de texto manuscrito y firmas. Como se ha podido comprobar a través de los resultados obtenidos, éstos tienen un comportamiento similar al de la señal de voz en cuanto al método de inicialización, criterios de parada, validación cruzada, estructuras híbridas, etc.

6.2 Líneas Futuras

A lo largo de esta tesis, se ha adelantado algunas pruebas preliminares de aplicación de los métodos de optimización y modelos híbridos sobre otro tipo de aplicaciones tales como el reconocimiento de dígitos manuscritos o reconocimiento de firmas. Pero la aplicación de los modelos híbridos puede llevarse a diferentes campos o áreas de clasificación, de diagnóstico, etc. siempre aplicando las condiciones particulares de cada señal.

En esta tesis sólo se ha trabajado con una estructura híbrida formada por la conexión en cascada de los modelos ocultos de Markov y una estructura neuronal que puede tomar diferentes formas. Otro posible campo de trabajo es la aplicación de nuevas estructuras híbridas.

Las redes neuronales que forman parte de esta estructura híbrida han sido entrenadas mediante los algoritmos clásicos con lo que cabe la posibilidad de aplicación de otro tipo de algoritmos de entrenamiento o redes con distintas estructuras, y analizar más en detalle la estructura óptima en las redes MLP con la aplicación de técnicas de crecimiento o de poda.

Otro posible campo de acciones futuras podría ser el trabajar con otro tipo de estructuras híbridas no modulares sino en comité con las redes neuronales. Donde las redes neuronales participen en la tarea de clasificación temporal de la secuencia de entrada. Es decir, la clase a clasificar vendrá determinada por la salida de los HMMs y la salida de las

NNs, ponderando la salida de cada uno de ellos a la hora de tomar una decisión. Para ello será necesario trabajar con redes dinámicas lo que conduce a estructuras y entrenamientos complejos o de una forma más inmediata, trabajar con redes estáticas pero previamente se ha de aplicar técnicas de alinemiento temporal, (DTW), a los patrones de entrada, donde todos los patrones tengan la misma duración.

Otro campo de trabajo mucho más laborioso sería el trabajar con modelos de IOHMM, donde los modelos HMMs son visto o implementados mediante redes neuronales, se trata de una estructura recurrente, ver [Bengio96].

Bibliografía

[Bengio96] Yoshua Bengio and Paolo Frasconi, "Input-Output HMM's for Sequence Processing", *IEEE Trans. on Neural Network*, Vol.7, No. 5, pp 1231-1249, September 1996.

[Boulard96] H. Boulard, H. Hermansky, and N. Morgan "Towards increasing speech recognition error rates", *Speech Communications*, Vol. 8, No.3, pp. 205-231, May 1996.

[Breiman93] Leo Breiman, Jerome H. Friedman, Richard A. Olshen, Charles J. Stone, *Classification and Regression Trees*, Chapman and Hall Inc., 1984, 1993.

[Camino99] C R Morales, C M Travieso, I Alonso and M A Ferrer, "Signature classification by hidden Markov model", 33^{rd} *Annual IEEE Int Carnahn Conference on security technology*, Madrid, SPAIN 1999.

[Camino99-pfc] José Luis Camino Carmona, *Reconocimiento automático de firmas utilizando modelos ocultos de Markov*, ETSIT de Las Palmas de Gran Canaria, junio de 1999.

[Cerf94] P. Le Cerf, W.MA, Van Compernolle, "Multilayer Perceptrons as Labelers for Hidden Markov Models, *IEEE Trans. on Speech and Audio Precessing*, vol. 2, no 1, pp. 175-184, January 1994.

[Cho97] S.B. Cho, "Neural-Network Classifier for Recognizing Totally Unconstrained Handwritten Numerals, *IEEE Trans. On Neural Networks*, vol.8, No. 1, pp 43-53, January 1997.

[Chung95] Yong Joo Chung, Chong Kwan Un, "Multilayer perceptrons for state-dependent wightings of HMM likelihoods", *Speech Communications*, No 18, pp 79-89, 1996.

[Ciro99] C R Morales, C M Travieso, I Alonso and M A Ferrer, "Algoritmo Secuencial de Esqueletización para un sistema de reconocimiento de carácteres", *XIV Symposium Nacional de la Unión Cientifica Internacional de Radio*, pp 199-200, Septiembre 1999.

[Ciro99-pfc] Ciro Roldán Morales, *Reconocimiento de dígitos manuscritos usando modelos ocultos de Markov*, ETSIT de Las Palmas de Gran Canaria, abril de 1999.

[Day93] Day, S.P. and M.R. Davenport "Continuous-time temporal back-propagation with adaptative time delays", *IEEE Transactions on Neural Networks*, vol. 4, pp. 348-354, 1993.

[Duda73] Duda, R.O., And Hart, P.E. *Pattern Classification and Scene Analysis*, Wiley, New York, 1973.

[Farrell94] Kevin R. Farrell, Richard J. Mammone and Khaled T. Assaleh, "Speaker Recognition Using Neural Networks and Conventional Classifiers", *IEEE Trans. on Speech and Audio Processing*, vol. 2, No 1, part II, January 1994.

[Ferng94] Chiy-Ferng Perng, "Nonlinear System Identification with on-line learning Self Organization Neural Networks",*Proceedings of IEEE International Conference on Systems, Man and Cybernetics*, Vol 1, pp 138-141, 1994.

[Furui89] Sadaoki Furui, *Digital Speech Processing Synthesis and Recognition*, Dekker, 1989.

[Gallant90] Stephen I. Gallant, "Perceptron-Based Learning Algorithms, *IEEE Trans. on Neural Network*, vol. 1, no 2, pp. 179-191, June 1990.

[Gray92] A. Gersho, R.M. Gray, "Vector Quantization and Signal Compresion", Kluwer Academic Publishers, Norwell, MA, 1992.

[Haykin94] Simon Haykin, *Neural Network, a Comprehensive Foundation.* IEEE Press, 1994.

[Hecht91] Robert Hecht-Nielsen,*Neurocomputing.* Addison Wesley. Ed. (1991).

[Hush93] Don R. Hush and Bill G. Horne, "Progress in Supervised Neural Networks, What's New Since Lippmann", *IEEE Signal Processing Magazine*, January 1993.

[Itzi95] Itziar Alonso, Miguel A. Ferrer, "Cuantificación vectorial robusta al ruido utilizando redes neuronales", *X Symposium Nacional URSI 1995*, pp 37-40, Valladolid 1995.

[Itzi97] Itziar Alonso, Miguel A. Ferrer, Aníbal Figueiras, " Mejoras en la inicialización de HMMs ", *XII Symposium Nacional URSI 1997*, vol.1 pp 443-446, Bilbao 1997.

[Itzi98] I Alonso, C. Travieso M. Ferrer, A. Figueiras, " A comparative study of different hybrid HMM/NN classifiers for Speech Recognition", *Signal and Image Processing, SIP'98*, pp 259-263, Nevada 1998.

[Itzi99-1] Itziar Alonso, C Travieso M A Ferrer, A Figueiras, " Improving a HMM Speech Recognizer with a Polynomial GMDH Neural Networks", *IEEE International Workshop on Intelligent Signal Processing*, pp 231-235, Budapest, Hungary 1999.

[Itzi99-2] Itziar Alonso, C Travieso M A Ferrer, A Figueiras, " Improving a HMM Recognizer with a Polynomial GMDH Neural Networks", *Signal and Image Processing, SIP'99*, pp 279-283, Bahamas 1999.

[Ivak] http://www.inf.kiev.ua/GMDH-home/.

[Jain97] A.K. Jain, "Guest editorial special issue on artificial networks and statistical patterns recognition", *IEEE Trans. on Neural Networks* vol.8, No 1, pp 1-3, January 1997.

[J.Alonso98] Jesús Alonso, M.A. Ferrer, Itziar Alonso, "Detector de voz", *IV Jornadas Informáticas*, Julio 1998.

[Khaled97] Khaled El-Malch, "Comparison of Voice Activity Detection Algoritms for Wireless Personal Comunications Systems", *IEEE CCECE* 1997.

[Kitter82] Devijver P.A., and Kittler, *Pattern Recognition: A Statistical Approach*, Prentice Hall, Englewood Cliff NJ, 1982.

[Kundu97] A. Kundu and G. Chen, "An integrated hybrid neural network and hidden Markov Model classifier, *IEEE Trans. on Signal Processing*, vol. 45, no 10, pp. 2566-2570, October 1997.

[Kung93] S.Y. Kung *Digital Neural Networks*, Prentice Hall, New Jersey 1993.

[Kung95] S.Y. Kung, "Decision-Based Neural Network with Signal-Image Classification Applications, *IEEE Trans. on Neural Network*, vol. 6, no 1, pp. 170-181, January 1995.

[Lacou97] Yves Lacouture, "Mean-Variance back-propagation: a connectionist learning algorithm with a selective attention mechanism", in *Int. Journal Conf. On Neural Networks* 1997.

[Levin90] Esther Levin, Naftali Tishby, and Sara A. Solla, " A Statistical Approach to Learning and Generalization in Layered Neural Networks", *Proceedings of the IEEE*, vol 78, No. 10, pp. 1568-1574, October 1990.

[Lippmann87] R. Lippmann, "An introducction to computing with neural nets", *IEEE Acoustics, Speech and Signal Processing Magazine*, pp. 4-22, April 1987.

[Lippmann91] R. Lippmann, "A critical overview of neural network and conventional classifiers", In B.H. Juang, S.Y. Kung, and C.A. Kamm, editors, *Neural Networks for Signal Processing: Proceedings of the 1991 IEEE Workshop*, pp. 266-278. IEEE Press, 1991.

[Makhoul85] John Makhoul, Salim Roucos and Herbert Gish, "Vector Quantization in Speech Coding", *Proceeding of the IEEE*, Vol. 73, No 11 pp 1551-1588, November 1985.

[Maren90] Alianna Maren, Craig Harston, Robert Pap, "Handbook of Neural Computing Applications", Academic Press, 1990.

[National94] National Academy of Sciences, *Voice Comunications between humans and machines*, National Academy Press, Washington D.C. 1994.

[Nelson95] Nelson Morgan and Hervé Bourlard, "Continuous Speech Recognition",*IEEE Signal Processing Magazine*, pp 25-42, May 1995.

[Nikos97] Nikos Doukas, "Stability of a Voice Activity Detector Based on Sourse Separation", *IEEE DSP*, 1997.

[O'Gorman95] O'Gorman L., Kasturi R., *Document Image Analysis*, IEEE Computer Society Press, 1995.

[Pico93] J. Picone, " Signal Modeling techniques in speech recognition", *Proccedings of the IEEE*, vol 81. No 9, pp 1215-1247, 1993.

[Rabi92] L. R. Rabiner, " A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition", *Proc. IEEE*, Vol. 77, No2, pp. 257-286, 1992.

[Rabi93] Rabiner, B.H. Juang,*Fundamentals of Speech Recognition*, Prentice Hall, 1993.

[Rabi94] Rabiner, "Applications of Voice Processing to Telecommunications", *Proceeding of the IEEE*, Vol. 82, No 2, February 1994.

[Renals94] S.Renals, N.Morgan, H. Bourlard, M.Cohen and H. Franco, "Connectionist probability estimators in HMM speech recognition", *IEEE Trans.on Speech and Audio Proc.* vol.2 nol pp 161-174, 1994.

[Reed93] "Pruning algorithms - A Survey", *IEEE Trans. on Neural Networks*, vol.4, No 5, pp 740-747, September 1993.

[Robison94] A. Robison, " An application of recurrent nets to phone probability estimation", *IEEE Trans.on Neural Networks*, vol.5 pp. 298-305, 1994.

[Roe93] David B. Roe and Jay G. Wilpon, "Whither Speech Recognition: The Next 25 Years", *IEEE Communications Magazine*, Vol. 31, No.11 pp 54-61, November 1993.

[Rohwer90] R. Rohwer, "The moving targets training algorithm"in *Advances in Neural Information Processing Systems 4*, vol. 2, D. S. Touretzky, Ed. San Mateo, CA:Morhan Kaufmann, pp. 558-565, 1990.

[Sakoe78] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition", *IEEE Trans.on Acoustics, Speech and Signal Processing*, vol. ASSP-26, No. 1, pp 43-49, February 1978.

[Sarkar96] D. Sarkar, "Randomness in Generalization Ability: A source to improve it", *IEEE Trans.on Neural Networks*, vol.7, No 3, pp 676-685, May 1996.

[Schapire99] R.E. Schapire,"The strength of weak learnability, *Machine Learning*, 5:197-227, 1990.

[Sharkey96] Amanda J. C. Sharkey, " On Combining Artificial Neural Nets", Connection Science, Vol. 8. Nos 3 y 4, 1996, pp 299-313.

[Sharkey99] Amanda J. C. Sharkey, *Combining Artificial Neural Nets: Ensemble and Modular Multi-Net Systems*, Springer-Verlag London, 1999.

[Steve96] Steve Young, "A review of Large-vocabulary continuous-speech Recognition",*IEEE Signal Processing Magazine*, Vol 13, pp 45-57, 1996.

[Travi99] C. M. Travieso, C. R. Morales, I. Alonso and M. A. Ferrer, "Handwritten Digits Parameterisation for HMM Based Recognition", *Seventh International Conference on Image Processing and its Applications*, pp 770-774, Manchester UK, July 1999.

[Utans91] Joachin Utans and John Moody "Selecting Neural Network Architectures via the Prediction Risk: Application to Corporate Bond Rating Predition",*First International Conference on Artificial Intelligence Applications on Wall Street. IEEE Computer Society Press*, Los Alamitos, CA, 1991.

[Vaseghi92] S.V. Vaseghi, "State duration modelling in hidden Markov models",*Signal Processing*, 41, pp 31-41, 1995.

[Wada91] Yasuhiro Wada and Mitsuo Kawato, "Estimation of Generalization Capability by Combination of New Information Criterion and Cross Validation",*IJCNN*, 1991.

[Webb94] Andrew R. Webb, "Functional Approximation by Feed-Forward Networks: A Least-Squares Approach to Generalization", *IEEE Trans. on Neural Networks*, vol 5, No. 3, May 1994.

[Waibel89] Waibel, A. T. Hanazawa, G. Hinton, K. Shikano, and K.J. Lang, "Phoneme recognition using time-delay neural networks, *IEEE Transactions on Acoustics, Speech, and Signal Processing* ASSP-37, pp. 328-329, 1989.

[Widrow90] Bernard Widrow and Michael A. Lehr,"30 Years of Adaptive Neural Networks: Perceptron, Madaline, and Backpropagation", *Proc. IEEE*, vol. 78, pp. 1415-1442, 1990.