UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE INGENIERIA

MODELO NEURONAL FORMAL PARA EL RECONOCIMIENTO DEL LENGUAJE NATURAL

TESIS MANCOMUNADA

QUE PRESENTAN:

GUILLERMO ANTONIO ENRIQUEZ HERNANDEZ

ADRIANA I. TOKUN HAGA LOPEZ

PARA OBTENER EL TITULO DE:

INGENIERO EN COMPUTACION

DIRECTORES: DR. FRANCISCO CERVANTES PEREZ

DR. JESUS FIGUEROA NAZUNO

MEXICO, D. F. 1994

UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

Asesores: Dr. Jesús Figueroa Nazuno

Dr. Francisco Cervantes Pérez

Sustentantes: Adriana Tokun Haga López

_____

Antonio Enríquez Hernández

_____

A mi madre con todo mi amor y con una infinita admiración por ser el punto de inicio y fin de todo el recorrido de mi vida.

A mi padre por su cariño y apoyo. Compartiendo este momento, que simboliza al que él hubiera querido llegar.

A Juan, Eric y Héctor, tratando de corresponder en algo al gran cariño y apoyo que siempre me han brindado, y diciéndoles que todo lo que haga siempre lo harán conmigo.

ADRIANA

A mi tío Angel y a mi abuelita Paulina por su gran cariño, sobre todo a mi abuelita por esperar tanto.

ADRIANA

A mis padrinos por ser más que mis segundos padres.

A mis primos Hugo, Laura y Gustavo, por su confianza y cariño.

ADRIANA

A Javier que siempre tiene una palabra de aliento, sobre todo para los momentos difíciles y que sabe que es al único al que podría recurrir.

ADRIANA

A mis mejores amigos Adriana, Araceli, Lourdes y Héctor, por quererme, apoyarme y compartir los mejores momentos de mi despertar a la vida.

ADRIANA

A Liborio por cruzarce en mi camino.

A Guillermo por conocer mi mundo e ideales.

A Antonio por estar conmigo y compartir los mejores momentos de nuestras vidas.

ADRIANA

Con todo mi amor y admiración para la persona a quien debo en mucho este logro: Mi Madre

Para mi hermana Yolanda, por todo su amor, comprensión y apoyo en mi vida.

Con todo mi respeto y cariño a Carlos, por todas las enseñanzas que me ha forjado.

ANTONIO

A Javier González, Luis Clorio, Javier Bárcenas, Lorena Lemus por compartir mis ideales, mi mundo y toda una vida.

ANTONIO

A Adriana, por que ella sabe todo lo que significa para mi.

ANTONIO

A todas aquellas personas que sin quererlo o saberlo nos han ayudado a llegar hasta este momento:

Miguel Angel Cruz, Margarita Hernández, Francisco Pastén, José Tapia, Fernando Gay, Marcos Martínez, Jorge Pi, Luis Flores, José Luis Martínez, Ma. Teresa Bautista ... y tantas y tantas personas que con el simple hecho de coincidir en nuestro camino nos han dado grandes conocimientos.

ADRIANA Y ANTONIO

A quien se le negó la oportunidad de dar éste paso y ahora que nos toca darlo lo estará haciendo con nosotros; en tú memoria GERMAN.

ADRIANA Y ANTONIO

A la Facultad de Ingeniería y todos aquellos maestros que nos enseñaron la belleza de los secretos de la Naturaleza.

ADRIANA Y ANTONIO

Agradecemos al Dr. Jesús Figueroa y al Dr. Francisco Cervantes por su amistad, apoyo y orientación en la realización de este trabajo.

ADRIANA Y ANTONIO

Agradecemos a Eduardo Gómez su valiosa colaboración en los algoritmos utilizado en la presente.

ADRIANA Y ANTONIO

Todas las cosas son artificiales puesto que la Naturaleza es el arte de Dios.

Thomas Browne.

Yo quiero conocer como Dios creó este mundo. No estoy interesado en éste o aquel fenómeno, en el espectro de éste o aquel elemento. Quiero conocer sus pensamientos; el resto son detalles.

Albert Einstein

INDICE TEMATICO

INDICE TEMATICO

CAPITULO 2. TECNICAS DE ANALISIS Y PROCESAMIENTO PARA LA CLASIFICACION DE LAS SEÑALES DE VOZ

CAPITULO 3. LA NEUROCOMPUTACION

CAPITULO 4. MEMORIA DISTRIBUIDA ESPARCIDA

CAPITULO 5. DISEÑO DE UN SISTEMA PARA EL RECONOCIMIENTO DE VOZ

INTRODUCCION

INTRODUCCION

Tomando en cuenta la complejidad que involucra el reconocimiento del lenguaje natural, y la gran importancia que representa el realizarlo de forma automática por alguna máquina, el presente trabajo propone el diseño e implementación de un sistema por medio del cual se pueden obtener resultados satisfactorios en cuanto a capacidad de reconocimiento, y que además es práctico, fácil de manejar, sin utilizar tecnología muy compleja y sobre todo sin incurrir en grandes costos.

Es indudable que este problema ya ha sido tratado en muchas ocasiones pero con ideas y herramientas convencionales y generalmente costosas. De esta manera el sistema aquí desarrollado tiene como característica el no utilizar tales conceptos, permitiéndonos asimismo obtener adecuados porcentajes de reconocimiento no muy lejanos de los obtenidos por los sistemas anteriores.

El sistema propuesto, (que reconoce palabras aisladas con independencia de la persona que hable, lo cual es una diferencia substancial con respecto a muchos otros modelos ya estudiados), está integrado por dos subsistemas principales, una etapa de hardware que obtiene y transforma la señal de voz para ser manipulada por una computadora, y otra de software que hace uso de diferentes algoritmos computacionales para el proceso de reconocimiento de la señal de voz.

Para la manipulación de la señal fué necesario el uso de diversas técnicas de proceso y análisis de señales, tales como los diferentes tipos de transformaciones para pasar la información de la señal del dominio del tiempo a una representación en el dominio de la frecuencia. Estas técnicas son herramientas adecuadas para resaltar y obtener algunas características significativas (y para nuestro caso muy útiles) de una señal de voz determinada.

No obstante que esta parte del sistema puede situarse dentro de los procesos convencionales para el reconocimiento de voz, la idea más importante del diseño es el utilizar técnicas neurocomputacionales. Estos modelos neuronales formales, involucran en sí mismos, un enfoque nuevo y poderoso para tratar problemas complejos. Dentro de estos modelos podemos ubicar la Memoria Distribuida Esparcida, la cual es la parte más importante o núcleo de nuestro sistema para lograr de manera eficiente su tarea de reconocimiento del lenguaje.

El reconocimiento automático del lenguaje natural ofrece sin lugar a dudas una gran variedad de aplicaciones en diferentes áreas de interés humano, y si bien es cierto que, como ya se ha mencionado, el problema se ha intentado resolver en muchas ocasiones y de diferentes maneras, también lo es que con el acelerado avance de la ciencia y la tecnología, surgen nuevos métodos, nuevos modelos e ideas que aunque están asociados a nuevos conceptos nos son muy útiles en la resolución de antiguos problemas; tal es el caso de las redes neuronales que con su nuevo enfoque han dado un paso gigantesco para *"empezar"* a tratar de comprender y posiblemente imitar el funcionamiento del cerebro humano.

Finalmente, esperamos que la labor de investigación y desarrollo despierte interés en todas aquellas personas, que crean que se puede hacer cosas importantes en México y por mexicanos.

CAPITULO I

HISTORIA Y ANTECEDENTES DEL RECONOCIMIENTO DEL LENGUAJE

HISTORIA Y ANTECEDENTES DEL RECONOCIMIENTO DEL LENGUAJE

1.1 Introducción

El poder transmitir información a cualquier dispositivo electrónico, no representa gran problema ya que existen diferentes métodos indirectos de hacerlo, el escribir por medio de un teclado, el leer de discos o cintas las instrucciones que se desean realizar, etc; pero finalmente siempre es necesario valerse de un paso intermedio para lograrlo.

La forma más directa de recibir la información, es por medio de la voz, con la cual se vendrían a simplificar notablemente cualquier tipo de procesos, pero paradójicamente a esto, el implementar tal tipo de función para una máquina resulta ser un problema muy complicado, siendo que el humano la realiza inconscientemente. Es por esto que se han desarrollado a través de muchos años diversos prototipos para poder resolver esta compleja tarea.

1.2 La voz

Para poder definir una posible solución al problema del reconocimiento del lenguaje, primero es necesario conocer los mecanismos por los cuales se produce una señal de voz y las formas de representarla.

Para transmitir información a un receptor, un emisor o persona produce una señal de voz en forma de ondas acústicas. Esta señal consta de variaciones de presión como una función del tiempo y ésta se modera directamente en la parte frontal de la boca, fuente del sonido primario (aunque el sonido también emana de la ventana de la nariz, la garganta y de la lengua).

Las variaciones de amplitud para cada señal corresponden a las derivaciones de la presión atmosférica causadas durante el viaje de las ondas. La señal de voz es no-estacionaria, o variante en el tiempo, ya que sus características cambian conforme los músculos vocales se contraen o se relajan.

La voz puede ser dividida en segmentos de sonido, los cuales comparten varias propiedades acústicas con otros durante un pequeño intervalo de tiempo.

Desde que el emisor desea producir una secuencia de sonidos correspondientes al mensaje que será transmitido, los principales músculos vocales realizan movimientos involuntarios. Para cada

sonido, hay un posicionamiento de cada una de las diferentes articulaciones vocales: cuerdas, lengua, labios, dientes, velo del paladar y mandíbula.

Los sonidos generalmente se dividen en dos grandes clases:

(a) vocales: las cuales no restringen el flujo de aire en el aparato vocal, y las

(b) consonantes: que restringen el flujo de aire en varios puntos.

La forma en que se produce la voz puede relacionarse con el funcionamiento de un filtro, en el cual un sonido fuente excita un filtro del tracto vocal.(O'Shaughenessy,1987)

La fuente del sonido se produce en la laringe, en la base de los músculos vocales, donde el flujo de aire puede ser interrumpido periódicamente por las cuerdas vocales. Los soplos de aire producidos por la abducción y la aducción (abertura y cerradura respectivamente) de las cuerdas vocales generan una excitación periódica de los músculos vocales.

Para las dos excitaciones, sonoras y sordas, los músculos vocales actúan como filtros que amplifican las frecuencias de ciertos sonidos mientras que atenúan las de otros.

Como en una señal periódica, la voz tiene un espectro consistente de armónicas de la frecuencia fundamental de la vibración de la cuerdas vocales, esta frecuencia denotada como **F0**,

corresponde físicamente al tono que será percibido. Las armónicas son concentraciones de energía múltiples de **F0**. Una señal realmente periódica tiene un espectro lineal discreto, pero los músculos vocales raramente permanecen fijos en el tiempo, por lo que los sonidos vocales son cuasi-periódicos.

Por todo lo anterior, podemos finalizar que el producir una señal de voz, por simple que parezca, involucra una gran cantidad de conceptos y complicadas interacciones biomecánicas, que se pueden traducir en *modelos* con el fin de poder representar, explicar y simplificar el complejo proceso del habla. Todo esto nos proporciona herramientas para el procesamiento y análisis formal de la señal de voz.

1.3 Historia del reconocimiento automático del lenguaje

La evolución del reconocimiento de voz, involucra una larga lista de experimentos y diseños de sistemas automáticos. Los primeros intentos para la construcción de máquinas que pudiesen reconocer voz se realizaron alrededor de los años 40's. El problema que dio pauta a una seria investigación en esta área fué el evitar la necesidad de depender de una operadora cuando se deseaba establecer una comunicación telefónica. Se pensaba que si se pudiera reconocer por una máquina los dígitos hablados del número telefónico solicitado, sería mucho más eficiente y menos costoso.

No obstante, los intentos para la construcción de tales máquinas para el reconocimiento de una extensa variedad de voces, fallaron, por lo que se buscaron soluciones alternativas y no tan ambiciosas. Posteriormente se introdujeron los sistemas de telefonía con disco, los cuales han cambiado muy poco, desde su principio hasta nuestros días.

En los 50's las computadoras digitales empiezan a ser utilizadas en una diversidad de tareas. Los datos se introducían por un teclado y los resultados se obtenían vía impresora. Se argumentaba, por otra parte, que si la voz era el modo de comunicación natural entre los humanos, éste también podría ser utilizado en la comunicación hombre máquina, lo cual dio nuevos ímpetus para la investigación del reconocimiento automático de la voz.

1.3.1 Antes de 1960 : El enfoque acústico

Las experiencias obtenidas por un espectrógrafo para señal un sonido demostraron que diferentes palabras pronunciadas tienen grandes diferencias en sus patrones acústicos. Se creía que toda la información contenida para el reconocimiento de voz residía en la acústica de la señal. Este punto de vista fué reforzado por trabajos contemporáneos de síntesis de voz, los cuales fueron demostrando que son estos patrones de movimiento los que el humano utiliza para la percepción de la voz.

Uno de los primeros intentos para la construcción de un reconocedor basado en patrones acústicos fué hecho por Davis en 1952. Su dispositivo reconocía dígitos hablados, utilizando filtros. Un sistema similar fué desarrollado en 1958 por Dudley y Balashek el cual realizaba un análisis espectral de la señal de voz. Este sistema produjo buenos resultados cuando el parlante era el mismo que había generado los patrones acústicos, pero no fueron tan buenos para otros parlantes. El problema de estos sistemas no fué tanto el análisis empleado sino que no se pensó en las diferencias y variedad de patrones que pueden producir diferentes parlantes.

Otro sistema para el reconocimiento automático del lenguaje fué el llamado 'Escritor Fonético' (phonetic typewriter) de Olson y Belar (1956). Este sistema utilizaba también un banco de filtros pero asimismo usaba un **compresor** el cual ajustaba el nivel medio de la señal a una misma intensidad tanto para aquellas personas que su pronunciación fuera débil como para las que hablaran más fuerte, de esta amnera se reducía algo de la variabilidad.

Un sistema basado en la teoría de características fué diseñado por Wiren y Stubbs en 1956. El principio de ese reconocedor es el siguiente : primero se divide la palabra en sonidos **sonoros** y **sordos**. Entonces los sonidos **sordos** se subdividen en **fricativos**[1] y **plosivos**. Este principo de clasificación binaria se repite hasta que un simple fonema queda aislado. Todos estos trabajos fueron muy limitados.

[1] Dícese de las letras f,s,x,j cuya articulación hace salir el aire con cierto roce en la boca.

1.3.2 1960-1968 : El enfoque del reconocimiento de patrones

Las necesidades reales mostraron que una simple comparación de patrones acústicos tenía una aplicación limitada. Los patrones acústicos de una palabra pronunciada difiere en duración e intensidad y la misma palabra pronunciada por diferentes personas produce patrones acústicos los cuales contiene también diferencias en el contenido de frecuencias. Así, se sugirió que el reconocimiento de palabras habladas se podría realizar de manera análoga al reconocimiento de objetos por patrones ópticos. Los objetos producen diferentes patrones sobre la retina dependiendo de la distancia y perspectiva desde las cuales son observados. Los patrones producidos por un mismo objeto, sin embargo, puede ser comparado contra cada uno de los otros si se aplican las transformadas geométricas apropiadas.(Flanagan)

Uno de los primeros sistemas de reconocimiento de voz con diferentes parlantes fué realizado por Forgie (1959). Ellos notaron que existe una relación inversa entre el tono de la voz y la extensión de la vocal, así la frecuencia fundamental puede utilizarse para normalizar las demás frecuencias de la señal. Su reconocedor consistió de un banco de filtros de 35 canales conectados a una computadora.

Por estas fechas se hicieron estudios para el reconocimiento de períodos de voz mayores a las vocales y dígitos aislados. Esto involucra segmentación de la señal de voz en pequeñas unidades tales como fonemas. Este proceso es muy difícil de realizarse satisfactoriamente debido

a que muchos sonidos se mezclan en las fronteras mismas sin un límite físico aparente.

Hughes (1961) intentó resolver este problema **rastreando** las características acústicas utilizando un esquema similar al de Wiren y Stubss (1956). Trabajos similares fueron hechos por Sakai, Doshita y Gold (1966). Sakai y Doshita utilizaron equipo electrónico especializado que se adecuaba mejor que un computadora de propósitos general.

Otros trabajos durante este período se realizaron intentando el **reconocimiento de patrones por neuronas artificiales** o elementos con umbrales adaptativos. La idea fué simular las habilidades humanas para el reconocimiento de patrones por medio de redes de elementos que poseyeran propiedades similares a las neuronas cerebrales.[2] Esta técnica fué utilizada por Talbert en 1963 y Nelson en 1967.

1.3.3 1969-1976 : El enfoque lingüístico

Aunque la señal de voz contiene toda la información necesaria para que un receptor entienda un mensaje hablado, el mensaje no se entenderá sin que el parlante y el receptor hablen el mismo lenguaje. La persona que recibe la señal hace uso de su conocimiento del lenguaje para **decodificar** el mensaje. La mayoría de los primeros sistemas de reconocimiento del lenguaje descuidaron este detalle. Una excepción fué el construido por Fry y Denes(1958). Ellos tomaron

[2] Ver capítulo III La Neurocomputación.

ventaja del hecho de que la probabilidad de un fonema siga a otro, depende de la identidad del primero. Su sistema consistió de un analizador de espectros, un comparador de patrones y un diagrama almacenado de probabilidades de los fonemas en inglés. El sistema no fué muy bueno, pero el uso de un diagrama de probabilidades incrementó el porcentaje de reconocimiento de 24% a 44%.

Todos los reconocedores utilizan obviamente un conocimiento acústico, pero en los primeros sistemas el conocimiento acústico no fué de fundamental importancia. No obstante, el uso de conocimiento léxico es casi obligatorio. Es necesario conocer la estructura de cada palabra del vocabulario, ya que este puede ser representado como una secuencia de fonemas. Por otra parte, la pronunciación de las palabras cambian de acuerdo a su contexto. Por ejemplo,'did' se pronuncia normalmente '*dId*' y 'you' como '*jou*', pero 'did you' se pronuncia frecuentemente como '*dIdzê*'. Estos cambios sistemáticos son conocidos como reglas fonológicas y pueden ser usadas para incrementar la capacidad de reconocimiento (Oshika 1974). Lea (1973) investigó como características prosódicas se pueden extraer de los contornos de la frecuencia fundamental de la señal de voz.

Tappert (1974) desarrolló un sistema que hacía uso de un **conocimiento sintáctico.** El procesador acústico producía fonemas **ruidosos**, de ellos se formó un árbol el cual representaba las posibles secuencias de palabras dada esa cadena determinada. Restricciones sintácticas fueron utilizadas para escoger la ruta más parecida a través del árbol. En un experimento reportado por Tappert (1974), usando un vocabulario de 250 palabras y una gramática predefinida, se

obtuvieron porcentajes de reconocimiento para una farse del 54% lo cual correspondía a un 91% de reconocimiento por palabras. Estos resultados se obtuvieron por un solo parlante después del entrenamiento.

Otros sistemas utilizaron conocimiento semántico. SPEECHLLS (Woods 1975), por ejemplo, contenía una red que representaba asociaciones entre palabras y conceptos. En este sistema, si la palabra 'química' se reconoce , el procesador semántico sugerirá que palabras como 'análisis' y 'elemento' pueden ser encontradas en la señal de voz.

La mayoría de los sistemas desarrollados en esta época usaban un módulo diferente para cada fuente de conocimiento lingüístico. Uno de los grandes problemas involucrados fué cómo integrar estas diferentes fuentes de conocimiento.

1.3.4 1977 a mediados de 80's : El enfoque pragmático

El mayor avance en cuanto a reconocimiento de palabras aisladas es el uso de algoritmos de programación dinámica. Ello ha permitido introducir distorsiones no lineales en un proceso de comparación y clasificación. Un algoritmo de este tipo fue utilizado por Velichko y Zalgoruyko (1970) en un sistema que reconocía 200 palabras rusas con un porcentaje de error del 5%. El poder del método, sin embargo, fué demostrado por Sakoe y Chiba (1978), quienes optimizaron un numero de algoritmos de programación dinámica y alcanzaron un error del 0.2% para dígitos

japoneses con su mejor sistema.

Otros algoritmos basados en modelos estocásticos fueron desarrollados dando resultados alentadores (Jelinel, 1976). Un sistema basado en modelos de Markov fué desarrollado por Levison en (1973), alcanzando un 96% de reconocimiento después de ser entrenado para 1000 dígitos, y pronunciados posteriormente por 50 hombres y 50 mujeres. Algunos sistemas basados en modelos de Markov han aparecido en el mercado, incluyendo el Verbex 1800 y el Sistema Dragón.(Flanagan).

Durante este período se siguió trabajando con clasificadores basados en lingüística. Klatt(1979) ha propuesto un sistema basado en unidades fonéticas en las cuales todas las posibles pronunciaciones son representadas como una ruta a través de una red. De Mori y Laface (1980) han aplicado algoritmos de lógica difusa (fuzzy-set) al problema de voz continua. El KEAL, sistema de reconocimiento de voz ha sido desarrollado para el lenguaje francés (Mercier , 1980) y Niemann (1982) ha desarrollado un sistema para el reconocimiento del lenguaje continuo alemán.

1.3.5 Mediados de 80's - 90's : El enfoque neurocomputacional

La idea del reconocimiento de voz a través de modelos de neuronas artificiales que simularan el comportamiento de la red neuronal cerebral del humano fue retomada a raíz del éxito de tales

modelos en la resolución de problemas muy complicados, por ejemplo en el reconocimiento de patrones visuales.

Como hemos observado, los anteriores desarrollos de sistemas que reconociesen de forma automática una señal de voz tienen ciertos inconvenientes y sobre todo son costosos tanto en material, equipo y tiempo involucrado. Son sistemas que dependen en demasía de la persona que hable (y sobre todo de las diferencias entre parlantes masculinos y femeninos) y generalmente restringidos a palabras específicas que tienen validez dentro de un contexto determinado o dentro de un lenguaje específico. No obstante, como veremos, las características de los **modelos neurocomputacionales formales** , tienden a discriminar los efectos antes mencionados, por lo que son un fuerte candidato entre la gran variedad de sistemas automáticos de reconocimiento.

Son tales y tan poderosas las particularidades de estos modelos que se ha llegado a construir, un sistema capaz de **leer** y **pronunciar** un texto impreso en un libro (NetTalk,Sejnowsky), pero sobre todo, este sistema es capaz de **aprender**, (en situaciones experimentales de laboratorio).

1.4 Algunos problemas que deben considerarse en un proceso de reconocimiento de voz

Existen un gran número de palabras que se pronuncian diferente y que tienen diferente significado, pero que se escuchan, no obstante, parecidas. Por ejemplo consideremos las palabras

'casa' y 'caza'. Estos palabras homófonas pueden causar problemas para un sistema clasificador. Esta clase de problemas no pueden ser resueltos en un nivel acústico ni fonético, deben ser estudiados por niveles avanzados de lingüística, estructura y semántica.

Otro problema consiste en la localización de los límites de una palabra. Esto es, el significado de la palabra puede cambiar dependiendo del momento en que se finalice, por ejemplo : "Contenedor" (recipiente) y "Con tenedor" (uso del cubierto).

De manera semejante a lo anterior, si una secuencia de fonemas puede ser reconocida y correctamente segmentada en palabras, existirá una ambigüedad de significado hasta que todas la palabras se agrupen dentro de unidades sintácticas apropiadas. 'El niño saltó sobre la arroyo con el pez' puede significar que el niño con el pez saltó sobre el arroyo o que el arroyo en la cual el pez nadaba fué saltado por el niño. La correcta interpretación de este tipo de frases es frecuentemente únicamente posible si el contexto de la frase está disponible. Consideremos por ejemplo, la frase 'Leche de vaca en polvo'. Esta frase ha sido segmentada en palabras sin ambigüedad alguna y no tiene más que un significado válido, pero en primera instancia, no se sabe si lo que es polvo es la vaca o la leche. Otro ejemplo sería, 'Sombreros para niños de paja'; los sombreros deben ser los de paja y no los niños, pero eso solo lo sabemos (nosotros) por su significado dentro de un contexto. Esto es muy difícil sino que imposible que un sistema automático lo pudiera identificar.

1.5 Alcances

En la última década las máquinas que reconocen el lenguaje se han movido fuera del laboratorio y se han colocado dentro del mercado. Aunque es todavía poco común encontrar máquinas de reconocimiento del lenguaje, existen y están siendo usadas cada vez más. Las máquinas en el mercado no tienen todas las capacidades como se pudiera desear, pero aún así son usadas en ciertas aplicaciones. Mucho falta por hacer, pero avances substanciales han permitido un gran progreso, logrando establecer una tecnología formal de reconocimiento del lenguaje.

Uno de los avances que han permitido el reconocimiento de patrones en general, y el reconocimiento de voz en particular fue la disponibilidad de computadoras digitales. Esto permite desarrollar algoritmos más rápidamente y ser probados con grandes cantidades de información. Este desarrollo está aún en progreso, y avances en la velocidad y capacidad de almacenamiento permitirán asimismo desarrollar algoritmos más sofisticados. Sin embargo, lo más importante es la aparición de nuevas estructuras, nuevos lenguajes de programación y sobre todos **nuevas teorías** apropiados para aplicaciones en el reconocimiento de patrones.

1.6 Problemas Actuales

Ahora que los sistemas de reconocimiento del lenguaje están siendo probados en situaciones prácticas sus defectos pueden ser estudiados. Indudablemente uno de los aspectos molestos de los reconocedores de palabras aisladas es la necesidad de hacer una pausa entre las palabras. La emisión de palabras aisladas es una forma menos natural de comunicación. Los reconocedores de palabras aisladas tienen obviamente una ventaja, que es su alto porcentaje de reconocimiento.

Un problema obvio que debe ser superado es el hecho del efecto del ruido. En un estudio reciente Wilpon (1985) mostró que con datos grabados bajo condiciones de laboratorio sobre un línea telefónica privada, un sistema reconocedor del lenguaje tenía una precisión de reconocimiento del 98.4% en una tarea de reconocimiento de dígitos, mientras tanto usando el mismo sistema, una precisión de únicamente 94.4% se obtuvo con datos colectados sobre una línea telefónica pública. Algunas de estas diferencias pueden ser debidas a las variaciones en la pronunciación, pero esquemas separados fueron formados para cada conjunto de datos, el ruido externo de la línea debe haber contado mucho en la diferencia. Para muchas tareas la limitante de un vocabulario pequeño crea un problema. Esto es particularmente para aplicaciones del tipo de la máquina de escribir operada por voz. Esto es exigido por uno de los mejores sistemas actuales (Bahl, 1984), el cual está diseñado para ser usado en la mecanografía de cartas, puede operar con un vocabulario de 5,000 palabras, pero con una sintaxis artificial impuesta. El

problema es que para ese tipo de aplicaciones un incremento drástico en el tamaño del vocabulario es requerido para problemas que van más allá de los típicos reconocedores de dígitos.

Con un vocabulario grande, el problema del entrenamiento llega a ser importante. Por lo tanto es aconsejable adoptar **técnicas independientes del parlante**, de esta manera el entrenamiento no necesita ser ejecutado para cada usuario individual. La combinación de una técnica independiente del parlante y un vocabulario grande crean problemas los cuales se empiezan ahora a estudiar[3].

1.7 Sistemas comerciales y hardware

En los últimos años se ha visto un avance en los sistemas de reconocimiento automático de voz (SRAV) y un decremento en su costo. Junto con los nuevos algoritmos improvisados para SRAV's y la declinación del costo de microprocesadores y memoria, algunos chips ha sido diseñados especialmente para reconocimiento automático de voz (RAV). Aunque el RAV puede ser implementado en un procesador microprocesador estandard, el uso de pocas multiplicaciones, incluso, y el acceso a la arquitectura de la memoria del chip (típicamente de 64 Kbytes - 1 Mbyte) reduce la capacidad de los sistemas simples de operar en tiempo real. En particular, la parametrización y la comparación son repetitivas, tareas que consumen tiempo y que ahora pueden ser desarrolladas por microcircuitos de propósito específico.

---

[3] Refiérase al capitulo V Diseño de un Sistema para el Reconocimiento de voz.

Algunos chips NMOS de Iterstate, Weiktek, Voice Control Systems, General Instrument, Intel, Matsushita, Oki y Nippon Electric (NEC) son capaces de hacer un análisis completo para reconocimiento de palabras aisladas (RPI), para 340 palabras con dependencia del parlante y 27 palabras con independencia de la persona que hable. Mientras que la mayoría de estos chips utilizan análisis espectral, otras técnicas son también utilizadas, por ejemplo, Matsushita y NEC son eficientes computacionalmente hablando, ya que no utilizan multiplicaciones al realizar la transformación del dominio del tiempo al de la frecuencia, sino que usan la transformada de Walsh.

Comercialmente los sistemas reconocedores están más disponibles en la forma de tarjetas (circuito impreso) para servidores o terminales. Tales sistemas (RPI) generalmente tienen un costo de unos 1000 dólares y unos cientos de dólares para una terminal. Estos reconocen un vocabulario de 100-500 palabras en modo dependiente del parlante y únicamente 10-30 palabras en modo independiente del parlante. Virtualmente todas las manufacturas afirman que su producto excede un porcentaje de reconocimiento del 90%, pero pruebas comparativas imparciales son raramente realizadas para comprobarlo. Algunas compañías (como TI, Interstate, ITT, Audec, Logical, Votan, Tecmar, NEC, Future Solutions, Audopilot) fabrican reconocedores para su uso en máquinas personales (IBM PC, Apple). No obstante, sistemas que acepten un vocabulario arriba de unos pocos cientos de palabras son muy raros, aunque dos compañías (Kurzweil e IBM) planean tener productos de 10,000 palabras pronto.

Para realizar sistemas prácticos, con un vocabulario extenso y en tiempo real, es necesario

un **procesamiento en paralelo;** por ejemplo, el sistema Kurzweil utiliza 25 chips que actúan como filtros de alta precisión y 64 microprocesadores Motorola 6800 implementando múltiples "expertos" en paralelo, alcanzando una rapidez de cálculo del algunos miles de millones de instrucciones por segundo.

Seis compañías han estado activas recientemente en el área más difícil, el reconocimiento de lenguaje continuo (RLC) : Verbex (modelos 3000 y 4000), Votan, Texas Instruments (un sistema usando el chip TMS320), NEC (DP-200 y SR-100), Interstate y Dragon Systems (Mark I y II). Con un costo de alrededor de 12,000 dólares, el Verbex 3000 y el NEC DP-200 aceptan típicamente aceptan pronunciaciones del alrededor de 5 palabras conectadas de un vocabulario de 50- 360 palabras. El modelo menos costoso (2,000 dls.), el SR-100 requiere de pausas de 30 ms. entre cada palabra. El sistema TI para RLC es realmente un reconocedor de lenguaje continuo, ya que reconoce frases de 21 palabras pero identifica tales palabras dentro de un vocabulario de tan solo 50 palabras. Por último, Dragon proporciona la mejor opción en RLC al vender algoritmos de software mas que hardware.

1.8 Conclusiones

El reconocimiento de voz puede proporcionar un camino práctico para introducir datos discretos, tal como texto a una computadora, para aplicaciones donde los errores no tienen consecuencias catastróficas. Aún con la pronunciación de palabras separadas con pausas muy

marcadas, la mayoría de las personas pueden introducir datos por esta vía más rápido que escribiendo (arriba de 150 por minuto vía voz contra 50 palabras por minuto vía escritura). El hecho de que el porcentaje de error se incrementa con el tamaño del vocabulario, no debe representar una dificultad mayor ya que un pequeño vocabulario (digamos 50 palabras) es suficiente para la mayoría de las aplicaciones actuales del reconocimiento de voz.

La mayoría de los reconocedores utilizan un reconocimiento estadístico de patrones, aplicando modelos generales (redes) como estructuras para incorporar conocimiento acerca de la señal e voz en términos de marcos de referencia. Los parámetros de los modelos son estimados durante el proceso de entrenamiento en el cual los parlantes pronuncian palabras o frases, las que pueden ser repetidas posteriormente durante el reconocimiento. Empleado primariamente en el reconocimiento de voz continua, el enfoque cognitivo para el reconocimiento de patrones, analiza la voz desde el punto de vista de la producción de la misma. La investigación sobre métodos cognitivos examina modelos específicos de voz,los cuales conducen a un mejor entendimiento de cómo los humanos generan la voz. Aunque tales modelos debieran ser más eficientes que los esquemas clásicos de comparación contra una referencia , hoy en día, han tenido menos éxito que los métodos estadísticos debido a nuestro inadecuado conocimiento de cómo se produce la voz.

Desde el punto de vista de la Inteligencia Artificial, la mejores simulaciones del proceso humano se encuentran supuestamente en aquellos modelos que imitan mejor como funcionan los humanos. De esta manera, la síntesis y reconocimiento de voz se debe derivar de una mejor comprensión de cómo el humano produce y percibe la voz.

Los sistemas prácticos tanto para el reconocimiento de voz continuo como para los que sean independientes del usuario, requerirán un progreso más significativo en el entendimiento del lenguaje natural. Así, el futuro del los reconocedores automáticos de voz posiblemente combinarán las técnicas estadísticas con sistemas expertos avanzados que exploten todas las características que hacen que una señal de voz sea diferente de cualquier otra o que al menos así lo interpretemos.

Hemos intentado mostrar a grandes rasgos los grandes problemas que los modernos y futuros sistemas tendrán que resolver. Hasta hoy, y por regla general, es claro el inconveniente, de estar limitados en cuanto a la velocidad de respuesta, a la extensión del vocabulario que pueden manejar y sobre todo, en cuanto a las diferencias que pueden existir entre las diferentes pronunciaciones de una persona a otra. Algunos sistemas han podido resolver uno u otro punto, pero a costa de descuidar los demás, no se ha logrado integrar en un solo sistema todas las características deseables de éste. No obstante, la búsqueda continúa y una nueva propuesta en la cual sea más factible la integración de cada uno de los factores mencionados es utilizando paradigmas neurocomputacionales.

CAPITULO II

TECNICAS DE ANALISIS Y PROCESAMIENTO PARA LA CLASIFICACION DE LAS SEÑALES DE VOZ

TECNICAS DE ANALISIS Y PROCESAMIENTO PARA LA CLASIFICACION DE LAS SEÑALES DE VOZ

2.1 Introducción

En el capítulo anterior, se examinó la forma en como se produce el lenguaje natural y se describieron aspectos de las señales de voz que son muy importantes para el desarrollo de los procesos de comunicación. Las aplicaciones del procesamiento de voz (codificación, síntesis, reconocimiento, etc.) utilizan éstas propiedades para complementar sus tareas.

Para el almacenamiento o reconocimiento de la voz, lo más deseable es eliminar la redundancia contenida en la señal para obtener una eficiente representación de las características esenciales de la voz en forma de parámetros y para simplificar la manipulación de los datos. El más relevante de los parámetros para el reconocimiento de la voz es la consistencia entre los parlantes; ya que ellos producirán valores similares (pero no iguales) al pronunciar un mismo fonema.

Para este capítulo se investigaron cuales son los métodos que se utilizan para el análisis de la voz, tanto en el dominio del tiempo (los que interactúan directamente con la señal de voz), como en la frecuencia (involucrando la transformación espectral de la voz). El objetivo es obtener

una mejor representación de la señal de voz en términos de los parámetros que contienen la información más relevante dentro de un formato eficiente.

2.2 La señal de voz

Existen varias formas de caracterizar el potencial de comunicación de la voz. Una alta cuantización desarrollada en términos de teoría de la información fué introducida por Shannon, la cual nos dice que la voz se puede representar en términos del mensaje contenido en la información. Una manera alternativa para caracterizar la voz es en términos del contenido de información en el mnsaje de la señal.

Aunque las ideas de la teoría de la información juegan un papel muy importante en los sistemas sofisticados de comunicación podremos ver que es la representación basada en la forma de onda de la señal y varios modelos paramétricos los que se usan más en las aplicaciones prácticas.

Al considerar los procesos de comunicación de la voz, es útil el empezar a pensar que un mensaje se representa en varias formas abstractas en el cerebro del emisor. A través de un complejo proceso de producción de voz, la información en que el mensaje es convertido finalmente es una señal acústica. La información del mensaje puede ser representada de numerosas y diferentes maneras en el proceso de la producción de la voz. Por ejemplo, la

información del mensaje primero se convierte en un conjunto de señales neuronales que controlan un mecanismo articulado (para el movimiento de la lengua, los labios, la cuerdas vocales, etc.). Las articulaciones se mueven en respuesta de estas señales neuronales para producir una secuencia de gesticulaciones, dando como resultado final, una onda acústica, que es la que contiene la información del mensaje original.

En los sistemas de comunicación, la señal de voz es transmitida, almacenada y procesada de diferentes maneras.En general, hay dos principales aspectos en cualquier sistema:

1. Preservación del contenido del mensaje de la señal de voz.
2. Representación de la señal de voz, en una forma que sea conveniente para la transmisión y/o el almacenamiento, o en la forma que sea flexible, de manera que las modificaciones que se hagan a la señal de voz no afecten seriamente al contenido del mensaje.

La representación de la señal de voz debe ser tal que la información pueda ser extraída fácilmente por los humanos que la escuchan, y en nuestro caso, por una máquina.

2.3 Procesamiento de la señal

El mecanismo general de la manipulación y procesamiento general de la señal se muestra en el el siguiente esquema.

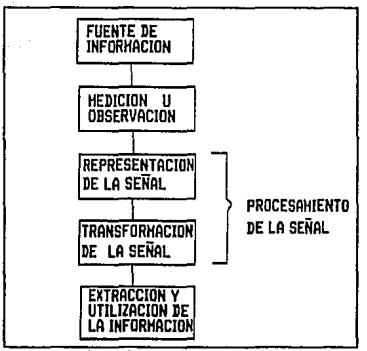


Ilustración 1.

En este caso la señal de voz que emite el humano es la fuente de información. La medición u observación es generalmente la forma de onda acústica.

El procesamiento de la señal involucra primero la obtención de una representación para la señal basada en un modelo dado y entonces la aplicación de varios niveles de transformación ordenada para colocar a la señal en la forma más conveniente. El último paso del proceso es la extracción y utilización del mensaje de la información. Este paso puede ser ejecutado por cada uno del los humanos que escuchan o por máquinas automáticas. A manera de ejemplo, un sistema cuya función es identificar automáticamente la voz de una persona de entre un conjunto, podría usar una representación espectral dependiente del tiempo de la señal de voz. Una posible transformación para la señal tendría que calcular la media espectral, a través de una expresión

completa, comparar la media de espectro con una almacenada para cada posible parlante entonces basado en una similaridad espectral medir la opción de identidad del parlante.

El procesamiento para las señales de voz generalmente involucra dos tareas. La primera, es el vehículo para obtener una representación general de la señal de voz en su forma de onda o de forma paramétrica. La segunda, el procesamiento de la señal cumple la función de ayudar en el proceso de transformación para la representación de la señal en formas alternativas, las cuales son menos generales en naturaleza, pero las más apropiadas para un análisis detallado y para aplicaciones específicas.

2.4 Técnicas para el procesamiento de la voz.

2.4.1 Conversion analógica-digital

El proceso de conversión de analógico a digital es la producción de un conjunto de puntos de valores discretos que representan o mapean a una forma de onda continua con tal precisión que la señal original pueda ser reconstruida. El proceso inverso para producir una forma de onda continua a partir de puntos de valores discretos es conocida como conversión Digital-Analógico (D-A).

Existen dos parámetros en el proceso de conversión: la frecuencia con la que la señal

puede ser muestreada, y la precisión (número de bits) con los cuales la cantidad de muestras se pueden representar. El teorema de muestreo, de Nyquist, establece la regla para no perder información de la señal. Esta no debe ser muestreada a una frecuencia menor que el doble de la frecuencia más alta de la señal.

Para el reconocimiento automático de la voz sólo son las frecuencia de interés las que percibe el rango de audición del oído humano. Este rango varía de 20 Hz hasta 20 KHz, aunque el limite superior se reduce progresivamente con la edad del receptor. En la realidad muchos de los niveles de energía para los sonidos de voz se encuentran por debajo de los 5 KHz, aunque varios sonidos fricativos, como la /s/, tienen energías superiores a los 10 KHz. Por esta razón frecuentemente se utilizan filtros paso-bajas con una frecuencia de corte de 5 KHz seguido por el convertido A-D con lo cual se pueden alcanzar hasta 10 KHz de muestreo, para representar digitalmente de los sonidos de la voz. El segundo parámetro el cual puede ser considerado en la conversión A-D es la exactitud con la cual la señal podrá ser muestreada. El rango de intensidad para la audición humana que va del umbral de audición hasta el de molestia o dolor, es de aproximadamente 120 decibeles (dB).

Por lo tanto si el rango esta representado por n bits:

$$\text{Rango en}_{dB} = 20 \log_{10} 2^n = 120$$

entonces

$$n = 120/(20 \log_{10} 2) = 20 \text{ bits}$$

por lo que la precisión requerida para cubrir el rango de percepción del oído humano es aproximadamente de 20 bits.

Las señales de voz, tienen un rango de duración de aproximadamente 70 dB, así ...

$$n = 70/(20 \log_{10} 2) = 12 \text{ bits}$$

De esta manera para obtener la representación digital de una señal que cubra todo el rango de audición humana puede emplearse una frecuencia de muestreo de 40 KHz con una precisión de 20 bits. (Ainsworth,1988).

2.4.2 Análisis en frecuencia

2.4.2.1 Transformada de Fourier (FT)

La información transmitida por los sonidos de la voz se codifican en frecuencias de vibración que son el resultado de las formas de onda. El métodos más común para el análisis de las frecuencias contenidas en la señal es el de Fourier.

El Teorema de Fourier establece que cualquier señal periódica puede ser representada mediante una la suma de una serie infinita de armónicas:

$$f(t) = a_0 + \sum_{n-1}^{\infty} a_n \sin(nwt + \phi_n)$$

donde a_n son los coeficientes y ϕ_n es la fase angular. La frecuencia angular es $w = 2\pi/T$, donde T es el período de la señal. Esto lo podemos escribir también como la suma de dos series infinitas:

$$f(t) = a_0 + \sum_{n-1}^{x} (a_n \cos nwt + b_n \sin nwt)$$

y como se puede observar los coeficientes se pueden calcular de:

$$a_0 = 1/2\pi \int_{-T/2}^{T/2} f(t)\ dt$$

$$a_n = 1/2\pi \int_{-T/2}^{T/2} f(t)\ \cos(nwt)\ dt$$

$$a_n = 1/2\pi \int_{-T/2}^{T/2} f(t)\ \sin(nwt)\ dt$$

Estas ecuaciones, las podemos reescribir en términos de números complejos, de la forma:

$$f(t) = \sum_{n=-\infty}^{\infty} C_n e^{inwt}$$

$$\text{donde: } C_n = \int_{-T/2}^{T/2} f(t)\ e^{-inwt}\ dt$$

Sí el período de la señal comprende un rango infinito, esto esta dado por:

$$f(t) = \int_{-\infty}^{\infty} F(w)\ e^{iwt}\ dw$$

$$F(w) = \int_{-\infty}^{\infty} f(t)\ e^{-iwt}\ dt$$

donde F(w) es la transformada de Fourier de f(t). Y si, si f(t) es una señal, F(w) es su espectro.

2.4.2.2 Transformada discreta de Fourier (DFT)

Sí la señal a sido digitalizada el análisis en frecuencia puede hacerse mediante una técnica conocida como la Transformada Discreta de Fourier o DFT. Asumiendo que la señal esta representada por x(n) donde n = 0,1, ..., N-1. Las relaciones son:

$$x(n) = 1/N \sum_{n=1}^{N-1} X(r)\, e^{2\pi i m/N}$$

donde

$$X(r) = \sum_{n=1}^{N-1} x(n)\, e^{-2\pi i m/N}$$

es la transformada discreta de Fourier.

Esta ecuación puede ser simplificada si sustituimos a $W = e^{-2\pi i/N}$, quedando de la siguiente forma:

$$x(n) = 1/N \sum_{n=1}^{N-1} X(r)\, W^{-m}$$

$$X(r) = \sum_{n=1}^{N-1} x(n)\, W^{m}$$

Una señal de voz no es periódica, pero no sufre muchos cambios de un período a otro. Sí el inicio de cada período puede ser determinado, es posible que N fuera igual al número de puntos en un período glotal y aplicar la DFT. Esto sería la más conveniente, pero en la práctica, es muy difícil determinar el inicio de cada período.

Normalmente se toma una secuencia arbitraria de N puntos. Esto equivale a multiplicar

la señal por un pulso rectangular el cual siempre es cero excepto durante el período a analizar. Esto introduce discontinuidades en los límites los cuales distorsionan el espectro agregándole componentes de muy altas frecuencias que no corresponden a dicho espectro.

Una técnica mejor, es el multiplicar la señal por una función suave. Las funciones suaves más usadas son las triangulares, gaussianas y el coseno, pero los efectos son los mismos. Una técnica común es el usar múltiples funciones coseno o ventana de Hamming. La ecuación es:

$$W(n) = 1/2\ (1-\cos(2\pi n/N-1)),\ n=0,1,...N-1$$

Las ecuaciones dadas expresan la DFT en términos de números complejos. Generalmente en el análisis de la voz es la energía de cada armónica (o frecuencia) la que se requiere. Esta es proporcionada por la potencia del espectro:

$$P_r = 10 \log_{10} (a_r^2 + b_r^2) \text{ decibeles}$$

donde a_r y b_r son los componentes reales e imaginarios del espectro complejo.

El oído humano es relativamente insensible a la fase de la señal percibida. Consecuentemente el espectro de fase es:

$$\phi_r = \tan^{-1} (b_r/a_r)$$

la cual no es usualmente requerida.

Para calcular la DFT para N-puntos se requieren de aproximadamente N^2 operaciones, las cuales son multiplicaciones y sumas. Se ha determinado que para el análisis de la señales de voz se requiere un período de 20 a 30 ms. Por ejemplo para una frecuencia de 10 kHz se necesitan aproximadamente 250 puntos. Para obtener un solo espectro es necesario ejecutar más de 60,000 operaciones. Como la voz es una señal variante en el tiempo es necesario calcular la DFT par 10 ms. Por lo que para analizar un segundo de la señal e voz es necesario calcular más de seis millones de operaciones complejas.(Refiérase a Burrus, 1985 para más información).

2.4.2.3 Transformada rápida de Fourier (FFT)

Un algoritmo más eficiente para calcular la DFT, fué el descubierto por Cooley y Tukey (1965). En este algoritmo las operaciones se dividen en dos conjuntos, donde cada conjunto es asimismo subdividido. Este proceso se repite hasta que cada conjunto contiene únicamente un término. Esta técnica requiere de solo N log N operaciones, lo cual disminuye considerablemente el tiempo de procesamiento de la señal. Pero para utilizar esta técnica es necesario utilizar un número de puntos que seas potencia de dos esto es N debe ser potencia de 2.(Refiérase a Brigham, 1988 para más detalle).

2.4.2.4 Tranformada de Hartley (TH)

Otra transformada utilizada es la de Hartley (*TH*). Esta herramienta matemática apareció por primera vez en "Proceedings of Radio Engineers" en el año de 1942 y fue desarrollada por Ralph V.L. Hartley. La TH no utiliza números complejos y se puede obtener la misma amplitud y fase que se obtiene del espectro de Fourier.

La Transformada de Hartley se define como:

$$H(f) = \int_{-\infty}^{\infty} f(t)\ \cos(2\pi ft)\ dt$$

donde:

$\cos(2\pi ft) = \cos(2\pi ft) + \text{sen}(2\pi ft)$

f(t) es una función en el dominio del tiempo

H(f) es una función en el dominio de la frecuencia.

Como la única diferencia en cuanto a las definiciones de la Transformada de Fourier y la Transformada de Hartley es el valor de j ($\sqrt{-1}$), las propiedades de simetría son las mismas, por lo que la ecuación anterior puede reescribirse en función de sus componentes de simetría par y non como:

$$H(f) = E(f) + O(f)$$

Considerando tales propiedades de simetría, la Transformada de Fourier puede ser obtenida por medio de la Transformada de Hartley de la siguiente manera:

$$E(f) = H(f) + H(-f)$$

$$O(f) = H(f) - H(-f)$$

donde:

E(f) y O(f) corresponden a la parte real e imaginaria de la Transformada de Fourier.

como $|F(f)| = (E(f)^2 + O(f)^2)^{1/2}$

$$P(f) = E(f)^2 + O(f)^2$$

$$= |H(f) + H(-f)|^2 + |H(f) - H(-f)|^2$$

$$= [|H(f)|^2 + |H(-f)|^2]/2$$

Haciendo un análisis semejante para la fase:

$$\text{Fase}(f) = \text{atan}\,[O(f)/E(f)]$$

$$= \text{atan}\left(\frac{H(f) - H(-f)}{H(f) + H(-f)}\right)$$

De lo anterior se puede observar que se obtiene la misma información que de la Transformada de Fourier, tanto de magnitud como de fase (Bracewell, 1986).

Semejante a la de Fourier, la Transformada Discreta de Hartley se obtiene a través de la siguiente ecuación :

$$H(r) = 1/N \sum_{n=0}^{N-1} h(n) \cos(2\pi rn/N)$$

2.4.2.5 Transformada de Haar (THr)

La Transformada de Haar forma parte de un grupo de transformadas que descomponen una señal en un conjunto de funciones rectangulares. Históricamente, las funciones de Haar fueron descritas por el matemático Húngaro Alfred Haar en 1910. Estas funciones también forman un conjunto completo de funciones ortogonales y ortonormales, cuya amplitud no tiene un valor único definido (como sucede en las funciones de Walsh -Gómez,E.,1992-) sino que depende de la función que se trate, y está en función de:

$$A = (\sqrt{2})^p, \text{ donde: } p = 1,2,....$$

Las funciones de Haar se pueden representar en el intervalo de tiempo $0 \leq t \leq 1$ como:

$$HAR(n,t)$$

donde: n identifica el número de la función

t es la base de tiempo

Si consideramos la base de tiempo definida como $0 \le t \le 1$, las funciones de Haar se pueden simplificar y obtenerse mediante la siguiente ecuación :

$$HAR(2^p+n,t) = \begin{cases} \sqrt{2^p} \text{ si } n/2^p \le t < (n+1/2)/2^p \\ -\sqrt{2^p} \text{ si } (n+1/2)/2^p \le t < (n+1)/2^p \\ 0 \quad \text{en cualquier otro caso} \end{cases}$$

donde: $p = 1,2...$ $\quad n = 0,1...2^p-1$

Ejemplos:

$$HAR(0,t) = 1 \text{ para } 0 \le t \le 1 \qquad HAR(1,t) = \begin{cases} 1 \text{ para } 0 \le t < \frac{1}{2} \\ -1 \text{ para } \frac{1}{2} \le t 1 \end{cases}$$

$$HAR(2,t) = \begin{cases} \sqrt{2} \text{ para } 0 \le t < \frac{1}{4} \\ -\sqrt{2} \text{ para } \frac{1}{4} \le t < \frac{1}{2} \\ 0 \text{ para } \frac{1}{2} \le t \le 1 \end{cases} \qquad HAR(3,t) = \begin{cases} 0 \text{ para } 0 \le t < \frac{1}{2} \\ \sqrt{2} \text{ para } \frac{1}{2} \le t < \frac{3}{4} \\ -\sqrt{2} \text{ para } \frac{3}{4} \le t \le 1 \end{cases}$$

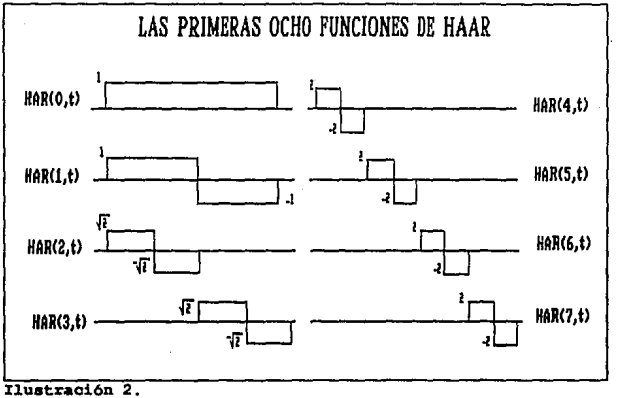


Ilustración 2.

Una característica que tienen las funciones de Haar, es la de convergencia, que permite representar discontinuidades en un segmento de la función con un número pequeño de términos, debido a que uns función de Haar puede representar un pequeño cambio en un segmento de la función.

Como se trata de un conjunto de funciones ortogonales, deben cumplir con:

$$\int_0^1 HAR(m,t)\ HAR(n,t) = \begin{cases} 1 \text{ si } n = m \\ 0 \text{ si } n \neq m \end{cases}$$

Y por tanto podemos llegar a que la transformada de Haar se define como :

$$HA(f) = \int_0^1 f(t)\ HAR(n,t)\ dt$$

donde los límites de la integral corresponden a la base de tiempo $0 \leq t \leq 1$.

De la ecuación anterior, podemos deducir que la Transformada Discreta de Haar esta definida por:

$$HA(r) = 1/N \sum_{n=0}^{N-1} ha(n)HAR(r,n/N) \text{ ; donde } i,n = 0,1 \ldots N-1$$

Como podemos observar, la ecuación anterior implica el hacer N^2 sumas. Sin embargo, éstas pueden ser reducidas a p*N donde $N= 2^p$ si únicamente los valores diferentes de sero son considerados.

En realidad se trata de algoritmos del tipo que Cooley y Tukey emplearon para la Transformada Rápida de Fourier, que tal parece ser, son algoritmos semejentes para diferentes tipos de transformadas , (Fino,1972).

2.4.3 Banco de filtros

En el pasado la cantidad de operaciones computacionales involucradas tanto para calcular la DFT era tan grande como para el procesamiento adecuado de la voz en un tiempo razonable, por lo que surgió la necesidad de desarrollar un método alternativo para estimar la potencia del espectro. En el cual la señal de voz se pasaba a través de un conjunto de filtros paso-banda y la se media la cantidad de energía resultante en cada canal. Las formas de onda representativas de la energía en cada uno de los filtros, como en una función del tiempo se digitalizaba y almacenaba en la computadora para otro tipo de procesamiento.

Este método de análisis se deriva de una técnica desarrollada para transmitir voz en un reducido ancho de banda. El equipo para implementar esta tarea se conoce como canal de voz.

Un canal de voz consiste de dos partes: Un analizador y un sintetizador. El analizador contiene un conjunto traslapado de filtros paso-bajas. La señal de voz se introduce a través de estos filtros y se mide la energía presente en cada uno. Una estimación de la frecuencia fundamental es también obtenida junto con una señal, indicando en cada instante si la señal de voz de entrada es sonora o sorda.

El sintetizador tiene un banco de filtros idéntico. Estos son excitados por pulsos periódicos de la frecuencia fundamental obtenida por el analizador. Sí la señal de voz original tiene voces

muy bajas o sordas, los pulsos se reemplazan por señales de ruido aleatorias. Un conjunto de amplitudes moduladoras controladas por señales de los filtros analizadores mezclan el espectro de la salida del sintetizador.

Se ha determinado que en la práctica se requieren de 10 y 20 filtros paso-banda para producir una señal de voz inteligible.(Ainswoth,1988).

El analizador del canal de voz es usado como un rápido y conveniente método de obtención del espectro de potencia de la señal de voz para el subsecuente análisis en el reconocimiento de la voz.

2.4.4 Análisis de autocorrelación.

La formas de onda de una señal de voz, se repiten aproximadamente, pero no exactamente, durante cada ciclo glotal. Una forma de estimar el intervalo de repetición, o frecuencia fundamental, es mediante la examinación de la función de autocorrelación. Esta función se obtiene multiplicando una señal x(n) por una versión de ella misma pero retardada en el tiempo.

$$\phi(k) = \sum_{n=1}^{N} x(n)\, x(n+k)$$

Prácticamente con los ejemplos de las formas de onda es necesario restringir el rango de la sumatoria a N puntos. Esto es especialmente necesario para las señales de voz ya que estas son no estacionarias por naturaleza (variantes en el tiempo).

La función de autocorrelación puede usarse para calcular la frecuencia fundamental de una señal de voz y también para determinar que voces son las más fuertes y cuales las más débiles.

2.4.5 Análisis de predicción lineal.

Todas las técnicas anteriormente mencionadas para el análisis de las señales de voz no hacen ninguna suposición de como se produce la voz. El análisis de predicción lineal asume que la señal a ser analizada se produce pasando por una señal de excitación a través de un filtro apropiado. Este es un buen modelo de la producción de muchos sonidos de la voz, el análisis de predicción lineal es una técnica muy apropiada para el análisis de la señal de voz.

Suponiendo que la forma de onda x(t) ha sido digitalizada. Si x(t) era continua, entonces una muestra x(n) puede conocerse a partir de una muestra previa:

$$x(n) = a_1 x(n-1) + e(n)$$

donde el coeficiente a_1 se escoge cuando la señal de error e(n) es pequeña. Solo si la forma de onda cambia lentamente a_1 será constante para varias muestras.

Esta idea puede ser extendida para reducir x(n) de las p muestras anteriores:

$$x(n) = a_1x(n-1) + a_2x(n-2) + \dots + a_px(n-p0) + e(n)$$

lo que es igual a

$$x(n) = e(n) + \sum_{k=1}^{p} a_kx(n-k)$$

La forma de onda puede se predecir en base a los coeficientes a_k y de la señal e(n).

El problema es como estimar los valores de los coeficientes. Estos se pueden obtener minimizando la el error cuadrático medio.

$$M = e(n)^2 = [x(n) - \sum_{k=1}^{p} a_kx(n-k)]^2$$

El orden para minimizar diferencias con respecto a cada uno de los coeficientes e igualar el resultado a cero.

$$dM/da_j = -2 \sum_{n} x(n-j)[x(n) - \sum_{k=1}^{p} a_kx(n-k)] = 0$$

o también

$$\sum_{k=1}^{p} a_k \sum_{n} x(n-j)\, x(n-k) = \sum_{n} x(n)\, x(n-j)$$

donde j = 1, 2, ... , p.

Este es un conjunto de p ecuaciones simultáneas lineales para p donde las incógnitas son $a_1 \ldots a_p$. Este conjunto de ecuaciones se puede resolver mediante métodos conocidos.

Con esta suposición es posible simplificar la ecuación en términos de la función de autocorrelación:

$$R(m) = \sum_{n} x(n)\, x(n+m)$$

Con el método de covarianza el rango de la sumatoria se maneja como finito, de n = h hasta n = h + n. Por lo que la ecuación anterior esta dada por:

$$\sum_{k=1}^{p} a_k \sum_{n=h}^{h+N-1} x(n-j)\, x(n-k) = \sum_{n=h}^{h+N-1} x(n)\, x(n-j)$$

donde j = 1, ... , p.

Como se puede observar el análisis de predicción lineal equivale a descomponer la señal de voz en una fuente y componentes de filtrado los cuales son equivalentes; además la señal de

error puede relacionarse con la fuente del sonido y los coeficientes pueden ser usados para determinar los polos del filtro complejo.

Las ecuaciones de la predicción lineal pueden ser utilizadas para la síntesis de la señal de los coeficientes de la predicción lineal. Se ha comprobado que diez coeficientes son suficientes para reproducir una voz con calidad aceptable.

2.5 Clasificadores de patrones

Un clasificador de patrones es un sistema para estimar la clase a la cual pertenece un patrón. Si los patrones se representan como puntos en un espacio n-dimensional, entonces existen tres métodos por los cuales se pueden clasificar los patrones. Estos han sido descritos por Levison (1985) como geométricos, topológicos y probabilísticos. Estos métodos se ilustran en la figura para un espacio bidimensional.

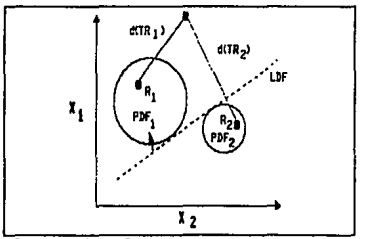


Ilustración 3.

R1 y R2 son los patrones de referencia y T es el patrón de prueba. En una clasificación geométrica T pertenecen la misma clase que R1 porque la línea se encuentra del mismo lado que la función de discriminación lineal LDF. En una clasificación topológica T pertenece a la misma clase que R1 si la distancia d(TR1) es menor que la distancia d(TR2). En un clasificador probabilístico T pertenece a la misma clase de R1 si la función de densidad probabilística PDF1 es mayor que PDF2.

Dentro de los métodos geométricos el espacio se divide por medio de fronteras en regiones.(Ainsworth,1988). Las posiciones de estos límites han sido seleccionadas de forma que los patrones de una clase caen a un lado del límite y aquellos de otras clases caen del otro lado. En un espacio n-dimensional los límites de la región pueden ser superficies. Frecuentemente, para facilitar su programación computacional, los hiperplanos son empleados como superficies de decisión.

En los métodos topológicos cada clase se representa por uno o más puntos en el hiperespacio. Un patrón desconocido se clasifica por la distancia considerada entre el punto que representa al patrón desconocido y los puntos que representan las clases. El punto que tienen la menor distancia es el que se toma para representar la clase del patrón desconocido.

Ambos métodos dependen solo de las coordenadas y de latopología local del espacio de los patrones. Los métodos en los cuales solo se depende de la distancia son frecuentemente

llamados clasificadores no paramétricos.

Dentro de los métodos probabilísticos una función probabilística de densidad (PDF) es definida para cada patrón en el espacio. La probabilidad de que un patrón desconocido pertenezca a una cierta clase estimada por las PDF's, seleccionando la clase en la cual el valor de PDF es mayor en un punto en el espacio ocupado por el patrón. Como las PDF's son usualmente caracterizados por parámetros tales como, posiciones, y las extensiones de los picos de esas funciones, los métodos probabilisticos son algunas veces llamados clasificadores paramétricos.

2.5.1 Correlación

Una de las técnicas más recientes usada para el reconocimiento de patrones es la correlación. Cada clase de patrones es representada por un vector:

$$Y_k = (Y_{1k}, Y_{2k}, \ldots Y_{nk})$$

Estos patrones también se conocen como modelos o plantillas.

Un patrón desconocido se representa por el vector característico:

$$X = (X_1, X_2 \ldots X_n)$$

La correlación entre un patrón desconocido y cada una de las plantillas se calcula por la evaluación:

$$C(X,Y_k) = \sum_{i=1}^{N} X_i Y_{ik}$$

La clase a la cual un patrón desconocido pertenece se estima calculando $C(X,Y_k)$ para cada clase K; y escogiendo K para cuando la función sea máxima.

La técnicas de correlación tienen ciertas desventajas, ya que para patrones muy parecidos, no son muy selectivas.

2.5.2 Análisis discriminante

Otra técnica para el reconocimiento de patrones, es el análisis discriminante, (Ainsworth,1988). Los patrones son presentados como puntos en un espacio n-dimensional. Las hipersuperficies están construidas en este hiperespacio de tal forma que los patrones están separadas dentro de clases. Estas hipersuperficies son conocidas como funciones discriminantes. En ciertos casos, especialmente cuando las clases están bien separadas la función discriminate puede ser representada por hiperplanos, las ecuaciones, las cuales toman la forma:

$$\sum_{i=1}^{N} W_{ik} X_i + Wo = 0$$

donde x_i son las variables y las W_{ik} son los coeficientes.

La determinación de los coeficientes W_{ik} se conoce como análisis discriminate y los hiperplanos representados por la ecuación anterior, son llamados función discriminante lineal. Nótese que la forma de esta ecuación es similar a aquellas del análisis correlacional.

La estructura de una máquina clasificadora de patrones basada sobre este análisis discriminante lineal es igual a la del perceptrón (y sus variantes) de Rosenblat, descritos en el capítulo III La Neurocomputación.

2.5.3 Clasificadores del vecino más cercano

En lugar de clasificar un patrón determinando la región del espacio de patrones dentro del cual caerá, el patrón puede ser clasificado por la distancia de los miembros conocidos de cada clase. La teoría de estos clasificadores se le atribuye tanto a Cover como a Hart (1967).

Para determinar la proximidad relativa de los patrones, se requieren algunas distancias. En general la distancia entre dos puntos, X y Y, en un espacio n dimensional esta dada por:

$$D_r(X,Y) = \left(\sum_{i=1}^{N} |x_i - y_i|^r \right)^{1/r}$$

En este caso r = 2, donde se define la familia de distancias euclidianas.

$$D_2(X,Y) = \left(\sum_{i=1}^{N} |x_i - y_i|^2 \right)^{1/2}$$

Dentro de los vecinos más cercanos, o distancias-mínimas, el clasificador del sistema se entrena por medio de un patrón representativo de cada clase. Estos se conocen como los patrones modelos. Un patrón desconocido se clasifica calculando la distancia de cada modelo y entonces se selecciona la clase del modelo más cercano.

Como la raíz cuadrada de un número se incrementa monotonamente con este valor la distancia quedaría simplificada a:

$$D(X,Y) = \sum_{i=1}^{N} (x_i - y_i)^2$$

Esto es conocido como la distancia cuadrada entre patrones. Moore (1986) ha mostrado que esta distancia puede ser derivada como consecuencia de asumir estadística Guassiana:

$$D(X,Y) = \sum_{i=1}^{N} x_i^2 + \sum_{i=1}^{N} y_i^2 - 2 \sum_{i=1}^{N} x_i y_i$$

El término de Σx^2_i es común a todos las plantilla y así puede ser no utilizado o evitarse en un clasifiador del vecino más cercano. La Σy^2_i es constante para cada plantilla, y puede reemplazarse por -2wk. De aquí que las nuevas distancias entre en patrón desconocido y el plantilla k queden definidas por:

$$D'(X,Y) = W_k + \sum_{i=1}^{N} x_i y_{ik}$$

Observese que la signo puede cambiar si el máximo valor de esta función así lo requiere.

Cuando las características de los patrones tienen una interpretación física, las distancias geométricas son probablemente las más adecuadas, pero cuando las características se son construcciones matemáticas, como los coeficientes de la predicción lineal, se necesita de distancias más sofisticadas para ser empleadas.

2.5.4 Clasificadores probabilísticos

La probabilidad de que un patrón pertenezca a la clase k dada la observación O_i que puede escribirse como $Pr(H_k/O_i)$. Esto no se puede medir directamente, pero el teorema de Bayes establece que:

$$Pr(H_k/O_i) = Pr(O_i/H_k)\ Pr(H_k)/Pr(O_i)$$

donde $Pr(O_j/H_k)$ es la probabilidad de la observación dado que el patrón pertenece a la clase k, $Pr(H_k)$ es la probabilidad a priori de la clase k y $Pr(O_j)$ es una probabilidad a priori de la observación.

Pr(Oi/Hk) puede estimarse de un gran número de patrones pertenecientes a la clase k. Esta es la función de densidad probabilística en el espacio n-dimensional para la clase k. Si la totalidad de los patrones tienen la misma posibilidad de
ocurrir Pr(Hk) será constante.

Sí varias observaciones sugieren que el patrón pertenece a la clase k, u estas observaciones son independientes, podemos decir que:

$$Pr(H_k/O_j) = \text{const. } (Pr(O_1/k)\ Pr(O_2/k).../(Pr(O_j)Pr(O_2)...)$$

Las observaciones consisten de los valores las características las cuales definen el espacio del patrón. De esta manera es posible definir la función como:

$$g(k) = \text{const. } (Pr(x_1/k)\ Pr(x_2/k)\ ...)$$

lo cual es proporcional a la probabilidad de que el patrón pertenezca a la clase k que consiste de un vector característico $(x_1, x_2, ..., x_n)$. Por lo que tomando logaritmos la función queda:

$$g'(k) = \sum_{i=1}^{N} \log (Pr(x_i/k)) + const.$$

En otras palabras la posibilidad de que un patrón perteneciente a la clase k dado el vector característico X puede interpretarse como la suma de los logaritmos de las probabilidades de las características.

2.6 Conclusiones

A lo largo de la historia se han desarrollado importantes métodos de análisis para todo tipo de señales, los cuales son una herramienta muy poderosa para poder obtener las características (dependiendo del enfoque o alplicación) propias y más significativas de una señal.

En lo que corresponde a los clasificadores de voz, podemos observar que hasta el momento se han implementado diferentes métodos para poder reconocer señales de voz, cada uno con ciertas ventajas respecto a otro, pero cabe señalar que casi todos basados en un estricta manipulación de la señal.

Como veremos más adelante, nosotros proponemos la combinacion de las características del análisis tradicional de una señal (utilizando la Transformada Rápida de Haar), aunado con las ventajas que nos ofrece un modelo neurocoputacional para complementar el análisis y proceso de una señal de voz.

CAPITULO III

LA NEUROCOMPUTACION

LA NEUROCOMPUTACION

3.1 Introducción

Las redes de neuronas artificiales han sido estudiadas por muchos años con la esperanza de alcanzar un proceso que asemeje al humano para campos como el reconocimiento de patrones. Éstos modelos están compuestos por muchos elementos computacionales no lineales **operando en paralelo** y arreglados en patrones que de alguna forma pueden simular las redes neuronales biológicas. Estos elementos de cómputo o nodos, están interconectados por valores llamados **pesos** que típicamente se van adaptando de tal manera de incrementar su potencial de cómputo colectivo. Ha habido un reciente re-surgimiento en el campo de las redes neuronales artificiales debido a nuevas topologías y algoritmos, que a su vez pueden ser implementados con tecnología VLSI (Very Large Scale Integration), y sobre todo, a la suposición de que el paralelismo masivo es esencial para tareas complejas como lo son el reconocimiento de voz y de imágenes.

En este capítulo, discutiremos algunas características de tales modelos, revisando algunos modelos importantes de redes neuronales que pueden ser usadas para la clasificación o reconocimiento de patrones. Estas redes son **altamente paralelas** construyendo bloques que ilustran los componentes neuronales y sus principios de diseño que pueden ser usados para sistemas más complejos.

3.2 Algunas características de las redes neuronales

Los modelos de redes neuronales artificiales o simplemente **redes neuronales (RN)** se conocen por varios nombres tales como modelos conexionistas, modelos de procesamiento distribuido paralelo y modelos neuromorfos. Sin importar cual sea su nombre, todos estos modelos intentan alcanzar un buen rendimiento a través de una densa interconexión de elementos simples y paralelos de cómputo. Tales modelos tienen gran potencial en áreas tales como reconocimiento de voz y de imágenes, donde muchas de sus hipótesis se basan en el procesamiento en paralelo, gran capacidad de cómputo y, donde los sistemas actuales están lejos, muy lejos de igualar el alcance humano.

En lugar de ejecutar un programa de instrucciones secuenciales como en una computadora von Newmann, los modelos neuronales exploran hipótesis de competencia simultáneamente usando redes de paralelismo masivo compuestas por muchos elementos de cómputo conectados por **ligas de pesos variables.**

Los elementos de cómputo o nodos usados por la red neuronal son **no lineales**, son típicamente analógicos, y poco se pueden comparar con los modernos circuitos digitales. El nodo más simple, suma N entradas con un peso asociado cada una y obtiene un resultado al pasar tales entradas por una función no lineal como se muestra en la figura.

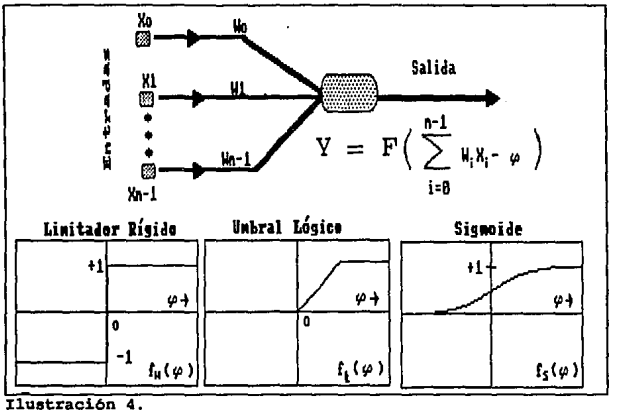


Ilustración 4.

El nodo está caracterizado por un umbral de valor Θ y por el tipo de no linealidad. La figura anterior muestra tres tipos comunes de no linealidades; un limitador rígido, umbral de elementos lógicos y funciones sigmoidales. Nodos más complejos pueden incluir integración en el tiempo u otros tipos de dependencias en tiempo y operaciones matemáticas más complejas que una simple suma.

Los modelos neuronales son especificados por la topología de la red, las características de los nodos y las reglas de aprendizaje o entrenamiento. Estas reglas establecen el conjunto de pesos iniciales e indican cómo estos pesos deberán cambiar para incrementar su funcionalidad.

Tanto los procesos de diseño como las reglas de entrenamiento, son los temas de más investigación actualmente.

Los beneficios potenciales de las redes neuronales van más allá de los grandes índices de cómputo debido al paralelismo masivo. Las redes neuronales proporcionan típicamente un más alto grado de **resistencia o tolerancia a la averías o daños** que las computadoras secuenciales de von Newmann debido a que existen más unidades de proceso (nodos), cada uno con conexiones primarias locales. El daño de unos pocos nodos o ligas no afectan significativamente el comportamiento general de la red. La mayoría de los algoritmos, por otra parte, adaptan los pesos de las conexiones para incrementar la funcionalidad de la red con base a resultados actuales; es precisamente esta adaptación o *aprendizaje* el foco de mayor atención dentro de la investigación en redes neuronales. La habilidad para adaptar y continuar aprendiendo es esencial en áreas tales como el reconocimiento de voz, donde los datos de entrenamiento se limitan a unos cuantos parlantes, y donde nuevas palabras, nuevos dialectos, nuevas frases y nuevos ambientes son continuamente encontrados.

Las técnicas estadísticas tradicionales no son adaptativas pero típicamente procesan todos los datos de entrenamiento simultáneamente antes de que empiecen a ser usados con nuevos datos. Los clasificadores neuronales son también no paramétricos y hacen poco caso de las tradicionales figuras de distribución de los clasificadores estadísticos. De esta manera, ellos pueden ser más resistentes cuando las distribuciones son generadas por procesos no lineales y son fuertemente no Gaussianas.

El diseño de redes neuronales artificiales para la resolución problemas, así como el estudio de redes biológicas reales pueden cambiar nuestra manera de pensar de como atacar los problemas y darnos, por otra parte, nuevas ideas y mejorar los sistemas computacionales.

Los trabajos sobre modelos neuronales tienen una larga historia. El desarrollo de modelos detallados matemáticamente comenzó hace más de 40 años con el trabajo de McCulloch y Pitss, Hebb, Rosenblatt, Widrow y otros. Trabajos más recientes hechos por Hopfield, Rumelhart y McClelland, Sejnowski, Feldman, Grossberg y otros han dado un resurgimiento al campo.

Las redes neuronales proveen una técnica para obtener la capacidad de procesamiento requerida usando una gran cantidad de elementos simples de proceso operando en paralelo.

3.3 Redes neuronales y clasificadores tradicionales

Un diagrama de bloques de clasificadores tradicionales y clasificadores utilizando redes neuronales es representado en el esquema, ambos tipos de clasificadores determinan cuál de las M clases es la más representativa de un patrón de entrada estático conocido que contiene a su vez N elementos de entrada. En un reconocedor de voz las entradas pueden ser los valores de salida de un analizador de espectros de un banco de filtros muestreados en un instante de tiempo y las clases pueden representar las diferentes vocales. En un clasificador de imágenes las entradas pudieran ser el nivel de gris de cada pixel de la imagen y las clases pueden representar diferentes

objetos.

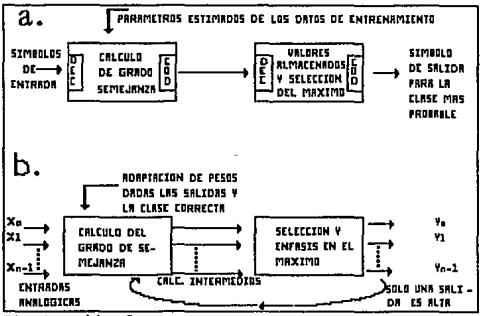


Ilustración 5.

a.Clasificador tradicional b.Red Neuronal

Los clasificadores tradicionales (Inteligencia Artificial Clásica, en la parte alta de la figura), comprenden dos etapas. La primera calcula el grado de semejanza o parecido para cada clase y la segunda selecciona la clase que tenga la máxima puntuación. Las entradas a la primera etapa son símbolos que representan valores de los N elementos de entrada. Estos símbolos son introducidos secuencialmente y decodificados desde la forma simbólica externa a una representación interna útil para ejecutar operaciones aritméticas y simbólicas. Un algoritmo calcula entonces el grado de semejanza para cada una de las M clases lo cual indica que tanto se parece el patrón de entrada con respecto al patrón representativo de cada clase (a éstos últimos los llamaremos "ejemplares"). En muchas situaciones un modelo probabilístico es usado para

modelar la generación de patrones de entrada dado un "ejemplar" determinado y el grado de semejanza representa la probabilidad de que el patrón de entrada haya sido generado desde cada uno de los M patrones "ejemplares" en la figura anterior.

Las Distribuciones Gaussianas son frecuentemente usadas en algoritmos relativamente simples para el cálculo del grado de semejanza. Posteriormente las puntuaciones para cada clase (de acuerdo a su grado de semejanza) son codificadas en una representación simbólica y pasan secuencialmente a la segunda etapa del clasificador. Aquí, son decodificadas y la clase con la máxima puntuación se selecciona. Un símbolo que representa a tal clase se envía a la salida y con ello termina la tarea de clasificación.

Por otro lado un clasificador basado en una red neuronal adaptativa se muestra en la parte inferior de la figura antes citada. Aquí los valores de entrada son alimentados a la primera etapa a través de N conexiones de entrada. Cada conexión lleva un valor analógico el cual puede tomar dos niveles para entradas binarias o puede variar sobre un gran rango de valores para entradas continuas. La primera etapa calcula el grado de semejanza y pasa estas puntuaciones en paralelo a la siguiente etapa sobre M líneas de salidas analógicas. Aquí el máximo de estos valores es seleccionado y resaltado o aumentado. La segunda etapa tiene una salida para cada una de las M clases. Después de que las clasificación se completa, únicamente esa salida correspondiente a la clase más parecida será puesta en "alto"; mientras, las otras salidas serán puestas en "bajo".

Note que en este diseño, existe una salida para cada clase y que esta multiplicidad de salidas

se debe conservar en etapas de proceso posteriores, así, habrá tantas salidas como clases se consideren distintas.

En el sistema de clasificación más simple estas líneas de salida pueden ir directamente a indicadores con etiquetas que especifiquen la identidad de las clases. En casos más complicados, irán a etapas de procesamiento posteriores donde entradas de otras modalidades o dependencias temporales se tomen en consideración. Si se proporciona la clase adecuada, entonces esta **información y las salidas del clasificador serán retroalimentadas a la primera etapa del clasificador para adaptar pesos utilizando un algoritmo de aprendizaje.** De esta manera, la adaptación hará más probablemente que se dé una respuesta correcta, para patrones de entrada que se asemejen al patrón actual.

Los clasificadores que se mostraron, pueden realizar tres diferentes tipos de tareas. Primero, como ya se ha descrito, pueden determinar cual clase representa mejor al patrón que se muestra a su entrada, el cual se asume ha sido deformado por "ruido" o por algún otro proceso. Este es un problema clásico de teoría de decisiones. Segundo, los clasificadores pueden ser utilizados como una memoria direccionable por contenido o **memoria asociativa**, donde una clase "ejemplar" es deseada y el patrón de entrada se utiliza para determinar cual "ejemplar" producir. Una memoria direccionable por contenido es útil, cuando únicamente está disponible parte de un patrón de entrada y el patrón completo se requiere. Esto requiere generalmente agregar una tercera etapa para generar el "ejemplar" para la clase más probable. No obstante, la adición de una tercera clase es innecesaria para algunas redes neuronales tales como la Memora Distribuida

Esparcida de Kanerva[1], las cuales están diseñadas específicamente como memorias direccionables por contenido.

3.4 Una taxonomía de las redes neuronales

Una taxonomía de seis importantes redes neuronales que pueden utilizarse en la clasificación de patrones estáticos se muestra a continuación.

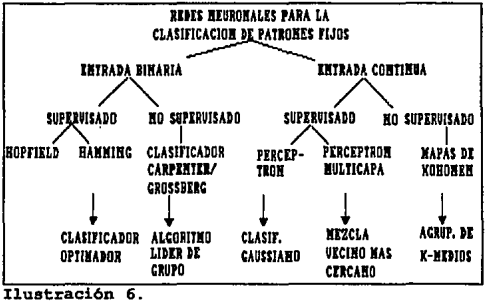


Ilustración 6.

Esta taxonomía primero se divide en redes con valores de entrada continuos y binarios. Estas a su vez se subdividen en entrenamiento supervisado y no supervisado. Las redes con

[1] Ver siguiente capítulo.

entrenamiento supervisado tales como las de Hopfield son usadas como memorias asociativas o clasificadores. Estas redes están provistas de información lateral o etiquetas que especifican la clase correcta de nuevos patrones de entrada durante el entrenamiento.

La mayoría de los clasificadores estadísticos, como los clasificadores Gaussianos, son entrenados de manera supervisada utilizando datos de entrenamiento "pre-etiquetados". Las redes con entrenamiento no supervisado, tales como los mapas característicos de Kohonen, se usan para formar grupos. Información no concerniente a la clase correcta es provista durante el entrenamiento para esta clase de red. Los clásicos algoritmos de los K-medios o líder de grupo son entrenados sin supervisión. Además, una diferencia entre las redes que no se muestra en la figura, es si el entrenamiento adaptativo puede ser soportado. Aunque todas las redes mostradas pueden ser entrenadas adaptativamente, las redes de Hopfield y Hamming son generalmente usadas con pesos fijos.

Los algoritmos listados en la parte inferior de la figura, son aquellos algoritmos clásicos que son los más semejantes para desarrollar la misma función que su correspondiente red neuronal. En algunos casos, una red implementa un algoritmo clásico exactamente. Por ejemplo, la red de Hamming es una implementación en red neuronal del algoritmo del clasificador óptimo para patrones binarios distorsionados por ruido aleatorio.

También se puede mostrar que la estructura de Perceptrón realiza aquellos cálculos necesarios por un clasificador Gaussiano, donde los pesos y umbrales son seleccionados apropiadamente.

En otros casos, los algoritmos de redes neuronales son totalmente diferentes de los algoritmos clásicos. Por ejemplo, el perceptrón entrenado mediante el procedimiento de convergencia se comporta de manera muy distinta a los clasificadores Gaussianos. Así como también, las redes de Kohonen no utilizan el entrenamiento iterativo de K-medios, en su lugar, cada patrón es presentado una sola vez y los pesos se modifican en cada nueva presentación (de manera muy semejante se comporta una red neuronal del tipo de una Memoria Distribuida Esparcida de P. Kanerva). No obstante, la red de Kohonen no forma un número predefinido de grupos, como lo hace el algoritmo de K-medios, donde K se refiere al número de grupos formados.

3.5 Algunos modelos de redes neuronales

En el presente capítulo no intentamos hacer un análisis profundo de los diversas redes neuronales que existen, sino mas bien dar un panorama general de la filosofía del proceso de información con este tipo de modelos. Si se desea un estudio más detallado veáse Lippmann 1987,1988,1989. En su lugar, analizaremos el modelo simplicado de la neurona biológca y algunas redes neuronales que han demostrado ser eficientes en el problema del reconocimiento y análisis de una señal de voz.

3.5.1 Redes Neuro-Lógicas de McCulloch-Pitts

En los años 40's surgió un simple pero poderoso concepto que mostraba que los componentes naturales de las *máquinas como la mente* podrían modelarse con base en el comportamiento de las células nerviosas biológicas, y tales máquinas podrían ser construidas interconectando tales elementos. En su manifiesto de 1943 "A Logical Calculus of de Ideas Immanent in Nervous Activity", Warren McCulloch y Walter Pitts presentaron la primera discusión formal de *redes neuro-lógicas* en la cual combinaban nuevas ideas acerca de máquinas de estado finito, elementos lineales de decisión (umbrales de decisión) y una representación lógica de varias formas de comportamiento y memoria. En 1947 publicaron otro trabajo,"How We Know Universals", el cual muestra, aunque a decir verdad de manera superficial, similitudes a la estructura del cerebro.

En primer término, para tratar de entender el modelo de McCulloch-Pitts será necesario introducir algunos conceptos matemáticos.

Un *grupo de transformaciones* es una colección G de transformaciones tales que si A y B están en G, entonces AB también están en G (donde AB es la transformación obtenida de realizar B y después la transformación A; nótese que en general, AB <> BA); y donde :

(a) I, la transformación identidad, esta en G. (I es la transformación trivial que deja todo fijo ,

por ejemplo, una traslación a través de una distancia de cero).

(b) Si A está en G, entonces tiene una inversa A^{-1} en G, tal que $AA^{-1} = A^{-1}A = I$. (A^{-1} es la transformación cuyo efecto es precisamente "deshacer" es efecto de A).

Nótese que para transformaciones tenemos asociatividad, esto es:

(AB)C = A(BC) = C seguida por B seguida por A

McCulloch y Pitts mencionan al respecto:

... numerosas redes, que se encuentran en estructuras nerviosas especiales, sirven para clasificar información utilizando caracteres comunes. En visión ellas detectan la apariciones equivalentes relacionadas por similitud y congruencia como aquellas de una cosa física vista desde varios lugares. En audición, ellas reconocen el timbre y acorde, sin tomar en cuenta el tono. Las apariciones equivalentes comparten una figura común y definen un grupo de transformaciones que pasan las equivalentes en otras, pero la conservan invariante la figura. De esta manera, por ejemplo, el grupo de traslaciones lleva un cuadrado apareciendo en un lugar a otros lugares, pero la figura de un cuadrado permanece invariante.

Las cosas que vemos y oímos son mapeadas al cerebro como patrones de neuronas disparando un grupo M ,de otras neuronas. La distribución de la excitación en M es descrita por una función $\phi(x,t)$, donde $\phi(x,t)=1$ si hay una neurona x disparándose en el tiempo t, y $\phi(x,t)=0$ en cualquier otro caso (Cuando no existe riesgo de confusión se omitirá la variable del tiempo y $\phi(x,t)$ se escribirá como $\phi(x)$ únicamente). (Arbib Michael,1987).

Ahora bien, sea G el grupo de transformaciones que realizan las funciones $\phi(x,t)$,

describiendo apariciones en sus equivalentes de la misma figura y supóngase que G tiene N miembros. Pitts y McCulloch consideran el caso donde las transformaciones T de G pueden ser generadas por transformaciones T' en el grupo de neuronas M, tal que

$$T\phi(x) = \phi(T'x),$$

es decir, el patrón es transformado alterando la actividad del conjunto x de acuerdo a la transformación T'.

Por ejemplo, si G es un grupo de traslaciones, entonces

$$T\phi(x) = \phi(x + a_T),$$

donde a_T es un vector constante que depende únicamente de T. Si G es un grupo de dilataciones

$$T\phi(x) = \phi(a_T x),$$

donde a_T es un número real positivo, que es el factor de magnificación, y que depende únicamente de T. Todas estas transformaciones son lineales, tal que

$$T(a\phi(x) + b\beta(x)) = a\phi(T'x) + b\beta(T'x) = aT\phi(x) + bT\beta(x)$$

La manera más simple para construir invariantes de una distribución de excitación $\phi(x,t)$ dada es obtener la media del grupo G. Así,sea f una funcional arbitraria (esto es una función de una función) que asigna un número a cada $\phi(x,t)$. Formamos cada una de las transformaciones $T\phi$ de $\phi(x,t)$, evaluando $f(T\phi)$, y promediamos los resultados sobre G para obtener

$$a = 1/N \Sigma f(T\phi), \quad \text{para toda } T \varepsilon G$$

Si empezamos con $S\phi$, donde S pertenece a G, en lugar de ϕ, tendremos

$$a = 1/N \Sigma f(TS\phi) \text{ , para toda } T \varepsilon G$$

$$= 1/N \Sigma f(R\phi) \text{ , para toda } R \varepsilon G \text{ tal que } RS^{-1} \varepsilon G$$

Para caracterizar completamente el "universal"[2] correspondiente a la aparición que produce el patrón de neuronas disparadas $\phi(x,t)$, necesitamos una colección completa de tales números *a* para diferentes funcionales f (una funcional corresponde a redondés, otra a cuadratura, etc.). Podemos distinguir entre las diferentes funcionales indexándolas por el subíndice δ para producir f_δ y permitiendo a δ un rango sobre un conjunto Φ. De esta manera obtenemos varios valores promedio

$$f_\delta(\phi) = 1/N \Sigma f_\delta(T\phi), \text{ para toda } T \varepsilon G$$

[2] Entiéndase universal como un término general o noción; un concepto abstracto o general respecto al cual se le da una absoluta existencia mental o nominal. Válido de una manera total e imperativa.

Introducimos un nuevo conjunto de neuronas, Φ, con una neurona para cada δ. Φ puede dividirse en varias dimensiones, en cuyo caso podemos especificar δ por sus coordenadas (δ_1, ..., δ_m). Si el sistema de neuronas necesita menos que la información completa para reconocer figuras, el grupo Φ puede ser más pequeño que M, tener menos dimensiones y en realidad reducirse a puntos aislados. El tiempo t y algunas de las x_j que representan la posición en M pueden servir como coordenadas en Φ, (Arbib Michael, 1987).

Como podemos observar, éste modelo de *Red de Neuronas Lógicas* tiene en principio, la capacidad del reconocimiento espacial de patrones de un manera invariante utilizando grupos de transformaciones geométricas. No obstante, hasta donde se sabe, no ha sido posible probar neurofisiológicamente que los mecanismos mostrados son empleados realmente en el funcionamiento del cerebro humano.

3.5.2 Adalinas y Madalinas

En la actualidad el análisis y procesamiento de señales involucra varias ramas de la ciencia y de la tecnología, con énfasis en el desarrollo de circuitos para el procesamiento digital de señales (DSP's) que pueden desarrollar las tareas de filtrado realizando filtros implementados en software. Esta implementación en computadora debe satisfacer que sea un sistema lineal, discreto e invariante en el tiempo. Si el sistema satisface estas restricciones, puede transformar una señal de entrada para producir una salida que corresponda a la entrada siendo pasada a través

de un filtro analógico. De esta manera, una computadora, ejecutando un programa que aplique una operación de transformación dada,*R*, a aproximaciones digitales (y por ende, discretas) de una señal de entrada continua, *x(n)*, puede producir una señal de salida *y(n)* por cada muestra de la entrada, donde *n* es el paso de tiempo discreto. Esta tarea para desarrollar tal transformación puede pensarse como un filtro digital. Además, cualquier filtro puede ser completamente caracterizado por su respuesta, *h(n)*, a la función impulso, representada como δ*(n)*. De manera más precisa,

$$h(n) = R[\delta(n)]$$

Lo importante de esta ecuación es que, una vez que se conoce la respuesta del sistema a la función impulso, la salida del sistema para cualquier entrada esta dada por

$$y(n) = R[\delta(n)] = \sum_{-\infty}^{\infty} h(i)x(n-i)$$

donde x(n) es la entrada del sistema.

Esta suma de productos es una operación similar al tipo de operación que realiza un *nodo o elemento de procesamiento* (EP)[3] dentro de un sistema neuronal artificial cuando calcula su

[3] Los elementos individuales computacionales que forman la mayoría de la redes neuronales artificiales son raramente llamados *neuronas artificiales;* es más frecuente que sean referidos como nodos, unidades o elementos de procesamiento (EPs). Como una neurona, el EP tiene muchas entradas, pero únicamente una salida, la cual puede llegar a muchas otros EPs en la red. Así, la entrada i-ésima recibida del EP j-ésimo la podemos detonar como x_j (la cual es la salida del j-ésimo nodo, tal como la salida generada por el nodo i-ésimo es x_i). Cada conexión al EP i-ésimo está asociada a una cantidad llamada *peso* o *fuerza de la conexión*. El peso de la conexión desde el nodo j-ésimo al nodo i-ésimo es denotado por w_{ij}. La salida del EP corresponde a la frecuencia de disparo de la neurona y los pesos corresponden a la

señal de activación. Específicamente, la Adalina, usa esta suma de productos para determinar cuanta estimulación de entrada recibe desde una señal de entrada en un instante. Como veremos, la Adalina extiende esta operación básica de filtrado y va más allá, ya que va adaptando los coeficientes de los "pesos" que le permiten incrementar o decrementar la estimulación que ella recibe en el siguiente tiempo con la misma señal. En otra palabras, la Adalina toma la entrada y la señal de salida deseada, y se ajusta ella misma de tal manera que pueda realizar la transformación deseada.

3.5.2.1 Adalina y el combinador lineal adaptivo

La Adalina, desarrollada por Bernard Widrow de la Universidad de Stanford, es un dispositivo consistente de un solo elemento de procesamiento, tal cual, no es técnicamente una red neuronal. Sin embargo, es una estructura importante que merece un estudio minucioso.

El término Adalina viene del inglés ADAptive LInear NEuron (Neurona Lineal Adaptiva); sin embargo su significado ha cambiado algo a través del tiempo. En los años 60's cambió a ADAptive LINear Element (Elemento Lineal Adaptivo). La estructura de la Adalina es muy parecida a la de un EP, pero existen dos modificaciones básicas para convertir la

---

conexión sináptica entre neuronas. Cada EP determina un valor de entrada a la red basado en todas sus conexiones de entradas. En la ausencia de conexiones especiales, la entrada a la red generalmente se calcula por la suma de los valores de entrada, multiplicados por sus correspondientes pesos.

$$red_i = \sum_j x_j w_{ij}$$

estructura general del EP en una Adalina. La primer modificación es la adición de una conexión con peso w_0, la cual referiremos como *término de tendencia*. Este término es un peso sobre una conexión que tiene su valor de salida siempre igual a 1. La segunda modificación es la adición de una condición bipolar sobre la salida.

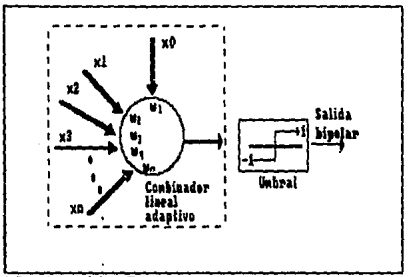


Ilustración 7.

El recuadro encierra la parte de la Adalina llamada *combinador lineal adaptivo* (CLA). Si la salida del CLA es positiva, la salida de la Adalina será de +1. Si la salida del CLA es negativa, la salida de la Adalina será de -1. El CLA realiza sumas de productos usando vectores de entrada y de pesos, y aplica una función de salida para obtener un solo valor de salida. Utilizando la notación de la figura anterior,

$$y = w_0 + \sum_{j=1}^{n} w_j x_j$$

donde w_0 es el peso de tendencia. Si hacemos $x_0 = 1$, podemos reescribir la ecuación anterior como

$$y = \sum_{j=0}^{n} w_j x_j$$

o en notación matricial $y = w^t x$

La Adalina (o el CLA) es ADAptivo en el sentido de que existe un procedimiento bien definido para modificar los pesos para obtener el valor de salida *correcto* para una entrada dada. Que valor de salida es el correcto depende de la función particular realizada por el dispositivo. La Adalina o el (CLA) es LIneal debido a que la salida es una simple función lineal de los valores de entrada. Finalmente es NEuronal únicamente en el estricto sentido de tener la estructura de EP. A continuación veremos el método de entrenamiento que utiliza la Adalina para realizar una función de proceso dado.

3.5.2.2 La regla LMS de aprendizaje

Dado un vector, *x*, es posible determinar un conjunto de pesos,*w*, tales que se obtendrá un particular valor de salida *y*. Supóngase que se tiene un conjunto de vectores de entrada,{x1, x2, .., x_L}, cada uno teniendo su valor de salida *correcto* o *deseado*, d_k,k = 1,L. El problema de encontrar un vector de pesos que pueda asociar satisfactoriamente cada vector de entrada con su valor de salida deseado no es una tarea simple. No obstante, el método llamado regla de

aprendizaje LMS (del inglés Least Mean Square - Menor [error] Cuadrático Medio), es uno de los métodos para encontrar el vector de pesos deseado. Este proceso de encontrar el vector de pesos lo llamaremos el *entrenamiento* del CLA. En este proceso, los pesos son levemente ajustados en cada combinación entrada-salida que es procesada hasta que CLA obtenga las salidas correctas. En este sentido, este procedimiento es un verdadero proceso de entrenamiento, ya que no necesitamos calcular el valor del vector de pesos explícitamente.

Antes de dar la descripción del proceso, comencemos con el problema un poco diferente: Dados los ejemplos (x_1,d_1), (x_2,d_2), ..., (x_L,d_L) de alguna función que asocia vectores de entrada, x_k, con (o mapea a) los valores deseados de salida, d_k, ¿Cual es el mejor vector de pesos, w^*, para un CLA que realiza este mapeo?

Para responder a tal pregunta, debemos definir primero que es lo constituye el *mejor* vector. La idea es eliminar, o al menos minimizar, la diferencia entre la salida deseada y la salida actual para cada vector de entrada. La premisa que se sigue es minimizar el error cuadrático medio para el conjunto de vectores de entrada.

Si el valor actual de salida es y_k para el k-ésimo vector de entrada, entonces el correspondiente término de error esta dado por $E_k = d_k - y_k$. Entonces el error cuadrático medio, valor de error esperado, esta definido por

$$\langle E_k^2 \rangle = 1/L \sum_{k=1}^{L} E_k^2$$

donde L es el número de vectores de entrada en el conjunto de entrenamiento. Utilizando la notación matricial, podemos expandir el error cuadrático medio como sigue:

$$\langle E^2_k \rangle = \langle (d_k - w^t x_k)^2 \rangle$$

Asumiendo que el conjunto de entrenamiento es estacionario, esto es, que cualquier valor esperado varía lentamente con respecto al tiempo, podemos factorizar el vector de pesos de los términos del valor esperado en la ecuación anterior y reescribirla de la siguiente manera (Freeman-Skapura,1991):

$$\langle E^2_k \rangle = \langle d^2_k \rangle + w^t \langle x_k x^t_k \rangle w - 2 \langle d_k x^t_k \rangle w$$

Ahora bien, si definimos $R = \langle x_k x^t_k \rangle$, llamada la matriz de correlación de entrada, y un vector $p = \langle d_k x_k \rangle$, además de hacer $\varepsilon = \langle E^2_k \rangle$, podemos reescribir la ecuación anterior como

$$\varepsilon = \langle d^2_k \rangle + w^t R w - 2 p^t w$$

Esta ecuación nos muestra que ε es una función explícita del vector de pesos w. en otras palabras $\varepsilon = \varepsilon(w)$.

Para encontrar el vector de pesos correspondiente al error cuadrático medio **mínimo**, derivamos ε con respecto a w, evaluamos el resultado en w^* e igualamos a cero:

$$\frac{\partial \varepsilon(w)}{\partial w} = 2Rw - 2p$$

$$2Rw^* - 2p = 0$$

$$Rw^* = p$$

$$w^* = R^{-1}p$$

Obsérvese que, ε es un escalar, mientras que la derivada parcial de $\varepsilon(w)$ con respecto a w es un vector, ya que es una expresión del gradiente de ε,$\nabla\varepsilon$, el cual es un vector.

$$\nabla\varepsilon = [\frac{\partial \varepsilon}{\partial w_1}, \frac{\partial \varepsilon}{\partial w_2}, \ldots, \frac{\partial \varepsilon}{\partial w_n}]^t$$

Todo lo que hemos hecho es mostrar que se puede encontrar un punto donde la pendiente de la función $\varepsilon(w)$ es cero. En general, ese punto puede ser un máximo o un mínimo. En el caso simple donde el CLA tiene únicamente dos pesos, la gráfica de $\varepsilon(w)$ es un paraboloide y en el caso de tener más dimensiones se conoce como hiperparaboloide.

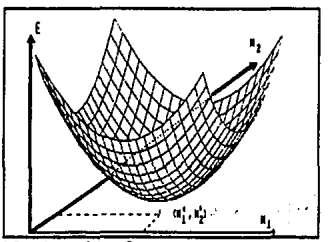


Ilustración 8.

3.5.2.3 Encontrando w^* por el Método del Descenso Rápido

El cálculo para determinar los pesos óptimos para un problema es más que difícil en términos generales. No únicamente se hace difícil la manipulación de la matriz para grandes dimensiones, sino que también cada componente de R y p son en sí mismos un valor esperado. De esta manera, los cálculos de R y p requieren del conocimiento de la estadísticas de las señales de entrada. Una mejor aproximación sería dejar al CLA encontrar los pesos óptimos por él mismo, teniendo que buscar sobre la superficie de los pesos hasta encontrar el *mínimo*. Si utilizásemos una búsqueda aleatoria no sería productiva ni eficiente, así, debemos agregar cierta *inteligencia* al procedimiento.

Empecemos por asignar valores arbitrarios a los pesos. Desde ese punto sobre la superficie de pesos, determinar la dirección sobre la cual la pendiente desciende más rápidamente. Entonces se cambian los pesos ligeramente de tal manera que el nuevo vector de pesos esté más abajo sobre la superficie. Se repite este procedimiento hasta que el mínimo sea alcanzado. Este método de ilustra en la siguiente figura.

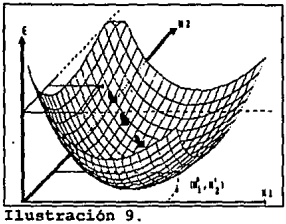


Ilustración 9.

Generalmente, el vector de pesos no se mueve inicialmente directamente hacia el punto mínimo. La sección transversal de una superficie paraboloidal de pesos es usualmente elíptica, así el gradiente negativo puede no apuntar directamente al punto mínimo, al menos inicialmente. La situación se ilustra más claramente en las curvas de nivel del paraboloide de pesos.

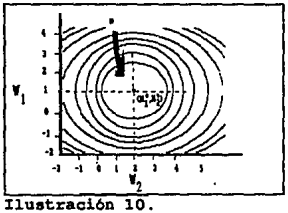


Ilustración 10.

Ya que el vector de pesos es variable en este procedimiento, lo escribimos como una función del tiempo. El vector inicial de pesos está denotado por w(t). En cada paso, el siguiente

vector de pesos se calcula de acuerdo a

$$w(t+1) = w(t) + \Delta w(t)$$

donde $\Delta w(t)$ es el cambio en w en el paso t-ésimo.

Para encontrar la dirección donde el descenso sea más abrupto en cada punto de la superficie, necesitamos calcular el gradiente de la superficie (el cual da la dirección de hacia donde la pendiente asciende más rápidamente). El negativo del gradiente está en la dirección donde se desciende más abrúptamente. Para obtener la magnitud del cambio, multiplicamos el gradiente por una contante μ. Este procedimiento resulta en la siguiente expresión:

$$w(t+1) = w(t) - \mu \nabla \varepsilon(w(t))$$

Todo lo que se necesita es determinar el valor de $\nabla \varepsilon(w(t))$ en cada paso sucesivo de la iteración.

El valor de $\nabla \varepsilon(w(t))$ fue determinado analíticamente previamente. Tales ecuaciones podrían ser usadas aquí para determinar $\nabla \varepsilon(wt))$, pero tendríamos el mismo problema que tuvimos en la determinación analítica de w^*: Necesitaríamos saber tanto R como p de antemano. Para evitar esta dificultad, utilizamos una aproximación del gradiente que puede ser determinado de la información que es explícitamente conocida en cada iteración.

Para cada paso de iteración, realizamos lo siguiente:

1. Aplicar un vector de entrada, x_k, a las entradas de la Adalina.

2. Determinar el valor del error cuadrático, $\varepsilon^2_k(t)$, utilizando el valor actual del vector de pesos.

$$\varepsilon^2_k(t) = (d_k - w^t(t)x_k)^2$$

3. Calcular una aproximación de $\nabla\varepsilon(t)$, utilizando $\varepsilon^2_k(t)$ como una aproximación de $<\varepsilon^2_k>$:

$$\nabla\varepsilon^2_k(t) \approx \nabla<\varepsilon^2_k>$$

$$\nabla\varepsilon^2_k(t) = -2\varepsilon_k(t)x_k$$

donde se ha usado la ecuación del paso (2) para calcular el gradiente explícitamente.

4. Actualizar el vector de pesos de acuerdo a la ecuación $w(t+1) = w(t) - \mu\nabla\varepsilon(w(t))$ y utilizar la ecuación del paso (3) como aproximación del gradiente:

$$w(t+1) = w(t) + 2\mu\varepsilon_k x_k \quad \ldots (a)$$

5. Repetir los pasos del 1 al 4 con el siguiente vector de entrada, hasta que el error haya sido a un valor aceptable.

La ecuación (a) es una expresión del algoritmo LMS. El parámetro μ determina la estabilidad y rapidez de convergencia del vector de pesos hacia un valor de error mínimo.

Debido a la aproximación del gradiente utilizada, el camino que el vector de pesos toma para moverse hacia mínimo de la superficie de pesos no será descrito por un movimiento suave sino por cambios bruscos como se ilustra en la figura siguiente:

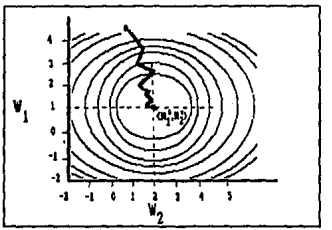


Ilustración 11.

Cambios pequeños en el vector de pesos deben ser tomados en cada iteración. Si son muy grandes, el vector de pesos podría vagar por la superficie, y nunca encontrar el mínimo, o encontrarlo únicamente por accidente más que como un resultado de una convergencia. La función del parámetro μ es evitar una búsqueda sin sentido. (Freeman-Skapura,1991).

3.5.2.4 Aplicaciones del Procesamiento Adaptivo de Señales

En repetidas ocasiones hemos experimentado el fenómeno del eco en las conversaciones telefónicas: uno puede oír las palabras que uno habla en el micrófono una fracción de segundo después en el auricular del teléfono. El eco tiende a ser más notable cuanto más largas son las llamadas, especialmente aquellas que son vía satélite donde el retardo de la transmisión puede ser una significante fracción de segundo.

Los circuitos telefónicos contienen varios dispositivos llamados *híbridos* que tienen el propósito de aislar las señales que llegan de las señales que salen, esto para evitar el efecto de eco. Desafortunadamente,estos circuitos no siempre realizan su tarea perfectamente, debido a incompatibilidad de impedancias, que resultan en un eco de regreso al parlante. Aún cuando la señal de eco haya sido atenuada en una cantidad substancial, sigue siendo audible y por lo tanto molesto para el parlante.

Ciertos dispositivos supresores de eco dependientes de relevadores, abren y cierran circuitos a través de la líneas, de tal manera que las señales de voz no son enviadas de regreso a la persona que habla. Cuando los retardos de las señales son grandes, como en las comunicaciones vía satélite, estos supresores de eco pueden truncar las palabras (efecto que es aun más común que el del eco). Un filtro adaptivo puede ser utilizado para quitar el eco sin el efecto de truncar las palabras.

La figura muestra un diagrama de bloques de un circuito telefónico con un filtro adaptivo usado como un dispositivo para la supresión del eco.

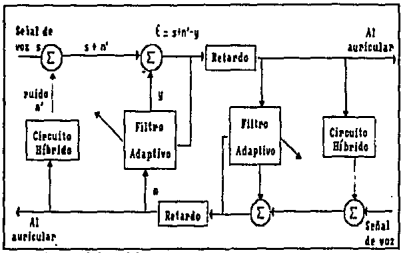


Ilustración 12.

El eco es causado por una fuga de las señal de voz que llega a la línea de salida a través del circuito híbrido. Esta fuga se agrega a la señal de salida que viene desde el micrófono. La salida del filtro adaptivo, *y*, es substraída de la señal de salida, $s+n'$,donde s es la señal de voz pura y n' es el *ruido* o eco causado por la fuga de la señal de voz de entrada a través del circuito híbrido. El éxito de la cancelación del eco depende de que tan bien pueda el filtro adaptivo imitar la fuga a través del circuito híbrido.

Nótese que la entrada al filtro es una copia de la señal que llega , n, y que el error es una copia de la señal que sale,

$$\varepsilon = s + n' - y$$

Asumimos que *y* está correlacionada con el ruido, *n'*, pero no con la señal de voz pura, *s*. Si la cantidad, n'-y, es diferente de cero, algún eco permanece en la señal que sale. Entonces, obteniendo el error cuadrático , tenemos

$$\langle \varepsilon^2 \rangle = \langle s^2 \rangle + \langle (n' + y)^2 \rangle + 2\langle s(n' - y) \rangle$$

Como *s* no está correlacionada ni con *y*, ni con *n'*, el último término de la ecuación es cero, por tanto puede escribirse

$$\langle \varepsilon^2 \rangle = \langle s^2 \rangle + \langle (n' + y)^2 \rangle$$

La señal de potencia, $\langle s^2 \rangle$, se determina por la fuente de la señal de voz. Así, $\langle s^2 \rangle$ se afecta directamente por cambios en $\langle \varepsilon^2 \rangle$. El filtro adaptivo intenta minimizar $\langle \varepsilon^2 \rangle$, y de esta manera, minimiza $\langle (n' - y)^2 \rangle$, la potencia del ruido no cancelado en la línea telefónica de salida.

El ejemplo detallado es un caso muy particular donde la Adalina ha sido aplicada satisfactoriamente; no obstante, a diferencia de muchas otras arquitecturas de redes neuronales, la Adalina es un dispositivo relativamente maduro con una larga historia de éxitos en variadas aplicaciones, (Freeman, 1991).

3.5.2.5 La Madalina

Como se obervará más adelante la adalina se asemeja mucho al perceptrón, pero también hereda sus limitaciones. Por ejemplo una Adalina de dos entradas no puede calcular la función XOR. Combinando adalinas en una estructura de capas, se puede evitar esta dificultad (como con el perceptrón multicapa). Tal estructura se muestra en el esquema.

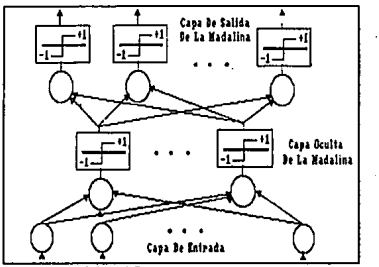


Ilustración 13.

Madalina es una abreviación de Muchas Adalinas (Many Adalines). Arregladas en una arquitectura multicapa, la madalina se parece a la estructura general de una red neuronal. En esta configuración, la madalina podría ser representada como un vector de grandes dimensiones. Con un apropiado entrenamiento, la red podría ser enseñada para responder con un +1 binario sobre alguno de los nodos de salida, cada uno de los cuales corresponda a una diferente categoría de

una imagen de entrada. En tal red, cada uno de los n nodos de salida corresponden a una sola clase. Para un patrón de entrada dado, un nodo tendría un +1 de salida si el patrón de entrada correspondiese a la clase representada por ese nodo en particular. Los otros n-1 nodos tendrían un valor de salida de -1. Si el patrón de entrada no fuera miembro de ninguna clase conocida, el resultado de la red podría ser ambiguo.

Para entrenar tal red, podríamos estar tentados a empezar con el algoritmo LMS en la capa de salida. Ya que la red está presumiblemente entrenada con patrones de entrada previamente identificados, el vector de salida *deseado* es conocido. Lo que no sabemos es la salida *deseada* por un nodo dado en las capas ocultas. Además, el algoritmo LMS operaría sobre las salida analógicas del CLA, no sobre los valores bipolares de salida de la Adalina. Por estas razones, es necesario una estrategia diferente de entrenamiento.

3.5.2.6 El algoritmo de entrenamiento MRII

Es posible concebir un método de entrenamiento para una estructura tipo Madalina basado en el algoritmo LMS; sin embargo, el método necesita reemplazar la función de salida (limitador lineal) con una función continuamente diferenciable (la función bipolar de salida o de umbral, es discontinua en 0; de esta manera, no es derivable en ese punto).

El MRII (Madaline Rule II) se parece a un procedimiento de prueba y error con una

inteligencia agregada en la forma de un *principio mínimo disturbio*. Este principio propone que aquellos nodos que puedan afectar el error de salida incurriendo en el menor cambio en sus pesos deben tener precedencia en el procedimiento de aprendizaje. Este principio está implícito en el siguiente algoritmo:

1. Aplicar un vector de entrenamiento a las entradas de la Madalina y propagarlo hasta la unidades de salida.

2. Contar el número de valores incorrectos en la capa de salida; a tal número le llamaremos el error.

3. Para todas las unidades en la capa de salida,
   a. Seleccionar el primer nodo previamente no seleccionado cuya salida *analógica* es la más cercana acero. (Este nodo es el nodo que puede revertir su salida bipolar con el mínimo cambio en sus pesos, de aquí el término de *mínimo disturbio*.)
   b. Cambiar los pesos sobre la unidad seleccionada tal que la salida bipolar de la unidad cambie.
   c. Propagar el vector de entrada desde las entradas a las salidas otra vez.
   d. Si el cambio de peso resulta en una reducción del número de errores, aceptar el cambio; de otra manera regresar los pesos originales.

4. Repetir paso 3 para todas las capas excepto para la capa de entrada.

5. Para todas las unidades de la capa de salida,

   a. Seleccionar el par de unidades previamente no seleccionadas cuyas salidas analógicas sean lo más cercanas a cero.

   b. Aplicar una corrección a los pesos de ambas unidades, de manera que se cambie la salida de cada uno.

   c. Propagar el vector de entrada desde las entradas a las salidas.

   d. Si el cambio de los pesos resulta en una reducción del número de errores, aceptar los cambios; de otra manera regresar los pesos originales.

6. Repetir paso 5 en todas las capas excepto en la capa de entrada.

Si es necesario, la secuencia de pasos 5 y 6 puede repetirse con tercias de unidades, o con combinaciones de cuatro o más unidades, hasta obtener resultados satisfactorios.

En 1991 se realizaron experimentos para determinar la convergencia y otras propiedades de este método. Además un nuevo algoritmo, MRIII, se desarrolló, el cual es similar a MRII, pero las unidades individuales tienen una función de salida continua, más que la función de umbral bipolar, (Freeman-Skapura,1991).

3.5.2.7 Madalinas como reconocedores de patrones invariantes a la traslación.

Varias Madalinas han sido usadas recientemente para demostrar la capacidad de su arquitectura para el reconocimiento adaptivo de patrones teniendo las propiedades de ser invariantes a la rotación, traslación y a la escala. Estas tres propiedades son esenciales para el reconocimiento de objetos por medios ópticos. La siguiente figura muestra una porción de una red que es usada para implementar un reconocedor de patrones invariante a la traslación. La **"retina"** es un arreglo de 5 por 5 pixeles sobre el cual, patrones mapeados en bits (bit-mapped), tales como las letras del alfabeto, pueden ser representados. La parte de la red mostrada es conocida como **slab** (tableta, plancha) . A diferencia de una capa, un slab no se comunica con otros slabs en la red. Cada Adalina en el slab recibe las mismas 25 entradas desde la retina, y calcula una salida bipolar en la forma usual; sin embargo, los pesos de las 25 Adalinas comparten una única relación entre ellas.

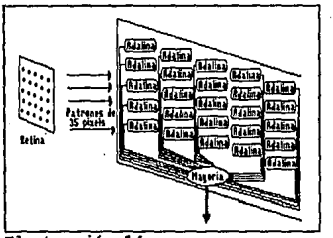
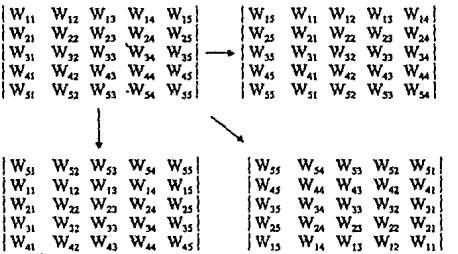


Ilustración 14.

Considérense los pesos de la Adalina de la esquina superior izquierda, como un arreglo matricial duplicando el arreglo de pixeles de la retina. La adalina inmediata a la derecha del pixel superior izquierdo tiene el mismo conjunto de valores de pesos, pero trasladados un pixel a la derecha: La columna de pesos más a la derecha sobre la primera unidad cubre la segunda unidad y llega a ser la columna de más hacia la izquierda de ésta. De manera semejante, la unidad bajo la unidad superior izquierda también tiene pesos idénticos, pero trasladados un pixel hacia abajo. El renglón inferior de pesos de la primera unidad se convierte en el renglón superior de la unidad debajo de ella. Esta traslación continúa a través de cada renglón y columna de manera similar.



Debido a la relación entre las matrices de pesos, un único patrón de la retina sacará idénticas respuestas del slab, independientemente de la traslación de la posición del patrón en la retina.

El nodo denominado *mayoría*, es una simple Adalina que calcula una salida binaria basada

en la salida de la mayoría de la Adalinas conectadas a ella. Gracias a la relación traslacional entre los vectores de pesos, la colocación de un patrón particular en cualquier lugar de la retina resultará en la idéntica salida para el elemento que llamamos mayoría. Debido a que únicamente dos respuestas son posibles, el slab puede diferenciar dos clases de patrones de entrada. En otros términos, un slab es capaz de dividir un hiperplano en dos regiones.

Para evitar la limitante de tener únicamente dos clases, la retina puede ser conectada a múltiples slabs, cada uno con diferentes matrices de pesos. Dada la naturaleza binaria de la salida de cada slab, un sistema de n slabs puede diferenciar 2^n diferentes clases de patrones.

3.5.3 El Perceptrón

La máquina que analizaremos a continuación es una versión abstracta de una clase de dispositivos conocidas con varios nombres, pero nosotros usaremos el de "perceptrón" en reconocimiento al trabajo pionero de Frank Rosenblatt.[4] Los perceptrones tienen la capacidad de hacer decisiones -esto es, determinan si un evento cae o no dentro de cierto patrón - gracias al hecho de ir agregando evidencia desde muchos pequeños experimentos. Este claro y simple concepto es importante debido a que la mayoría y más complejas máquinas para hacer decisiones comparten un poco de esta característica. Hasta que ésto no se entienda lo mejor posible,

[4] Los perceptrones han sido ampliamente publicados como máquinas de "reconocimiento de patrones" o de "aprendizaje" y como tales han sido discutidos en numerosos libros y artículos.

podemos tener problemas con más ideas avanzadas. De hecho, los avances críticos en muchas ramas de la ciencia y matemáticas comienzan con una buena formulación de sistemas "lineales" , y estas máquinas son un buen candidato para comenzar el estudio de *máquinas paralelas* en general, (Minsky M, Papert S., 1988).

3.5.3.1 Computación Paralela

El concepto más simple de computación paralela se presenta en la siguiente figura. En ella se muestran como uno podría calcular una función $\sigma(x)$ en dos etapas. Primero se calcula *independientemente* un conjunto de funciones $\phi_1(x)$, $\phi_2(x)$, ..., $\phi_n(x)$
y entonces combinar los resultados por medio de una función Ω de n argumentos para obtener el valor de σ.

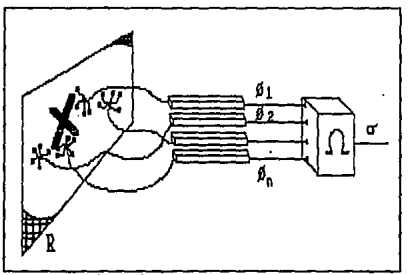


Ilustración 15

Para hacer la definición más productiva uno necesita poner algunas restricciones sobre la función Ω y el conjunto Φ de funciones ϕ_1, ϕ_2, ... Si no se tienen restricciones, no se puede obtener una teoría: cualquier cálculo de σ podría ser representado como un cálculo paralelo en varias formas triviales, por ejemplo, haciendo que uno de los ϕ's sea σ y dejando que Ω haga absolutamente nada mas que trasmitir su resultado.

Veamos algunos ejemplos concretos de las clases de funciones que quisiéramos que fuera σ.

Sea R un plano ordinario de dos dimensiones y sea X una figura geométrica sobre R. X podría ser un círculo o un par de círculos,etc. En general X será un subconjunto de puntos de R. Sea $\sigma(X)$ una función (de figuras X en R) que puede tener solo dos valores. Si utilizamos falso y verdadero en lugar de 0 y 1, podemos entonces pensar a $\sigma(X)$ como un predicado, esto es, una sentencia variable cuya certeza o falsedad depende de la selección de X. Los siguientes ejemplos muestran algunos predicados.

$$\sigma_{\text{círculo}}(X) = \begin{cases} 1 \text{ si la figura X es un círculo} \\ 0 \text{ si la figura X no es un círculo} \end{cases}$$

Ilustración 16.

$$\sigma_{convexa}(X) = \begin{cases} 1 \text{ si la figura X es convexa} \\ 0 \text{ si la figura X no es convexa} \end{cases}$$

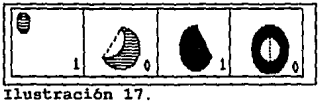


Ilustración 17.

$$\sigma_{conectada}(X) = \begin{cases} 1 \text{ si X es una figura conectada} \\ 0 \text{ de otra forma} \end{cases}$$

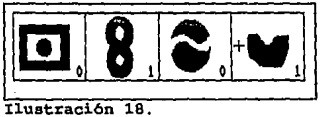


Ilustración 18.

También usaremos algunos predicados más simples[5]. Los predicados más simples que reconocen cuando un punto está en X: sea p un punto en el plano y definido por

$$\phi_p(X) = \begin{cases} 1 \text{ si p esta en X} \\ 0 \text{ de otra forma} \end{cases}$$

[5] Se usará ϕ en lugar de σ para aquellos predicados simples que serán combinados más tarde para formar predicados complejos.

Finalmente necesitamos un predicado cuando un conjunto particular A es un subconjunto de X:

$$\sigma_A(X) = \begin{cases} 1 \text{ si A pertenece a X} \\ 0 \text{ de otra forma} \end{cases}$$

Obviamente debe existir una diferencia entre $\sigma_{conectada}$ y $\sigma_{convexa}$. Para resaltarlo declaremos un hecho acerca de una figura convexa:

Un conjunto X no es convexo si y solo si existen tres puntos p, q y r una línea tales que :

p esta en X,
q no esta en X,
r esta en X.

De esta manera nosotros podemos probar si una figura es convexa examinando ternas de puntos. Si todas las ternas pasan las pruebas entonces X es convexa; si alguna terna falla (esto es, que no cumpla con todas las condiciones mencionadas arriba) entonces X no es convexa. Debido a que todas las pruebas pueden hacerse independientemente, y la decisión final hacerse por medio de un simple procedimiento lógico, ésto representa - aunque de groso modo - un definición de "local", (Minsky & Papert, 1988).

Definición: Un predicado σ es *conjuntamente local* e orden K si puede ser calculado como en §3.5.3.1 por un conjunto Φ de predicados ϕ tales que:

1. Cada ϕ depende de no mas de k puntos e R;

2. $\sigma(X) = \begin{cases} 1 \text{ si A pertenece a X} \\ 0 \text{ de otra forma} \end{cases}$

Por ejemplo $\sigma_{convexa}$ es conjuntamente local de orden 3. No obstante $\sigma_{conectada}$ no es conjuntamente local de ningún orden. Para probarlo, supongamos que $\sigma_{conectada}$ tiene orden k. Entonces para distinguir las dos figuras

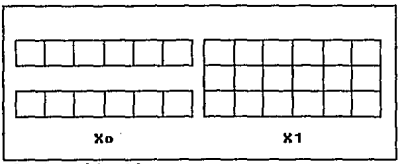


Ilustración 19.

debe haber alguna ϕ_0 tal que $\phi_0(X_0)=0$ debido a que X_0 es no conectada. Todas las ϕ's tendrán un valor de 1 en X_1, la cual es conectada. Ahora ϕ_0 puede depender a lo más de k puntos, entonces debe haber al menos un cuadro, llamémosle S_j, que no contiene uno de estos puntos. Pero entonces, en la figura X_2,

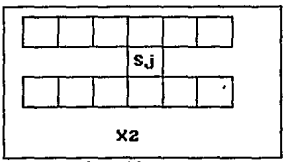


Ilustración 20.

la cual es conectada, ϕ_0 debe tener el mismo valor, 0, que tiene en X_0. Pero no puede ser, ya que todas las ϕ's deben tener un valor de 1 en X_2.Esto es claro, ya que si alguna ϕ pudiese ver *todos* los puntos de R entonces $\sigma_{conectada}$ puede ser calculada, pero ésto iría en contra de cualquier concepto de ϕ's como funciones "locales".

La intención de la definición fue dividir el cálculo de un predicado σ en dos etapas.

Etapa I:

El cálculo de muchas propiedades o características ϕ_α las cuales son fáciles de calcular, ya sea por que cada una depende únicamente de una pequeña parte de la entrada total R, o porque son muy simples de cualquier otra forma.

Etapa II:

Un algoritmo de decisión Ω que define σ combinando los resultados de los cálculos de la etapa I. Para que esta separación en etapas tenga más significado, esta función de decisión debe ser distintivamente homogénea, o fácil de programar o fácil de calcular (por ejemplo, en $\sigma_{convexa}$, la regla de decisión fue que si la ϕ's eran *unánimes* se aceptaba la figura; si una ϕ fallaba entonces la figura se rechaza).

Aún cuando encontramos un esquema intuitivamente satisfactorio, los requerimientos de la Etapa I y II contienen un carácter heurístico que dificulta una definición formal.

3.5.3.2 Perceptrones

Un esquema *perceptrón*, es una "combinación lineal" de predicados de la Etapa I. Esta "linealidad" la definimos :

Sea $\Phi=\{\phi_1, \phi_2, ..., \phi_n\}$ una familia de predicados, Entonces σ *es lineal con respecto a* Φ si y solo si existe un número Θ y un conjunto de números $\{\alpha_{\phi 1}, \alpha_{\phi 2}, ..., \alpha_{\phi n}\}$ tal que $\sigma(X)=1$ si y solo si $\alpha_{\phi 1}\phi_1(X), \alpha_{\phi 2}\phi_2(X), ..., \alpha_{\phi n}\phi_n(X) > \Theta$. El número Θ es llamado umbral y las α's son llamadas coeficientes o pesos. Esto puede escribirse de manera más compacta

$$\sigma(X) = 1 \text{ si y solo si } \sum_{\alpha\varepsilon\Phi} \alpha_\phi\phi(X) > \Theta$$

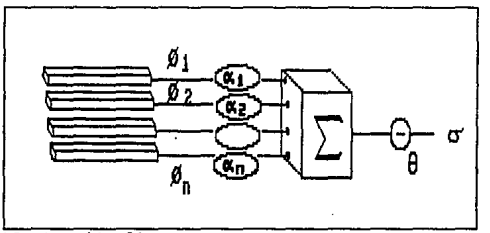


Ilustración 21.

De esta manera, cualquier predicado conjuntamente local puede ser expresado en esta forma si se elige $\Theta = -1$ y $\alpha_\phi = -1$ para cada ϕ. Para entonces

$$\Sigma(-1)\phi(X) > -1$$

exactamente cuando $\phi(X) = 0$ para cada ϕ en Φ.

Definición: Un *perceptrón* es un dispositivo capaz de calcular todos los predicados los cuales son lineales en algún conjunto dado Φ de predicados parciales.

Esto es, se da un conjunto de ϕ's, pero podemos seleccionar libremente su "pesos" (α_ϕ's) y también el umbral. Hay poco que decir acerca de todos los perceptrones en general, pero imponiendo ciertas condiciones y restricciones, se encuentra mucho que decir acerca de ciertas *familias* particulares de perceptrones.

1. *Perceptrones de diámetro limitado:* Para cada ϕ en Φ, el conjunto de puntos sobre el cual ϕ depende está restringido a no exceder un cierto diámetro fijo en el plano.

2. *Perceptrones de orden restringido:* Un perceptrón tiene orden $\leq n$ si ningún miembro de Φ depende de más de n puntos.

3. *Perceptrones aleatorios:* Estos son la forma más extensamente estudiadas por el grupo de Rosemblatt: los ϕ's son funciones booleanas aleatorias. Es decir, son de orden restringido y Φ es generada por un proceso estocástico de acuerdo a una función de distribución asignada.

4. *Perceptrones Limitados:* Φ contiene un número infinito de ϕ's pero todos los α_ϕ poseen un conjunto finito de números.

La más pura visión del perceptrón como un dispositivo de reconocimiento de patrones es el siguiente:

La máquina se construye con un número fijo de elementos de cómputo para la funciones parciales ϕ_i generalmente obtenidas por un proceso aleatorio. Para hacerlo reconocer un patrón particular (conjunto de figuras de entrada) uno únicamente tiene un conjunto de coeficientes α_i con valores adecuados. Así, el "programar" toma una agradable forma homogénea. Además como los "programas" son representados como puntos $(\alpha_1, \alpha_2, ..., \alpha_n)$ en un espacio n-dimensional, heredan una métrica lo cuál hace fácil imaginar una clase de programación automática la cuál la gente ha llamado *aprendizaje:* esto se hace poniendo dispositivos de retroalimentación a los parámetros α_i control y se "programa" la máquina alimentándola con una secuencia de patrones de entrada y una "señal de error" la cual causará que los coeficientes cambien en la dirección correcta cuando la máquina haga una decisión equivocada (Minsky M, Papert S., 1988).

El perceptrón fue concebido como una dispositivo de operación paralela en el sentido físico donde los predicados parciales son procesados simultáneamente. El precio pagado por ésto es que todas las ϕ_i deben ser procesadas, aunque únicamente una fracción pequeña de ellos puede de hecho ser relevante para cualquier decisión final particular. La cantidad total de cálculo puede llegar a ser bastante mayor al que llevaría en un proceso secuencial bien planeado (utilizando las mismas ϕ's). De esta manera la elección entre métodos paralelos o secuenciales en una situación particular debe basarse en un balance de reducir el tiempo de proceso con el costo de cómputo adicional involucrado.

De hecho, para ciertos predicados y clases de funciones parciales, el número de funciones parciales que tienen que ser usadas excedería los límites para la realización física. Sin embargo

la carencia de una teoría general no es excusa para evitar el problema de casos particulares y desarrollar una teoría para un limitado pero importante clase de problemas.

3.5.3.3 El perceptrón de una capa

El perceptrón de una capa es la primera de las tres redes que se mostraban en la taxonomía que es capaz de recibir valores continuos o binarios. Esta red simple fue muy interesante cuando fue desarrollada por su habilidad para aprender a reconocer patrones. Un perceptrón que decide si una entrada pertenece a una de dos clases (denotadas como cuadros y círculos) se muestra se muestra en el esquema siguiente:

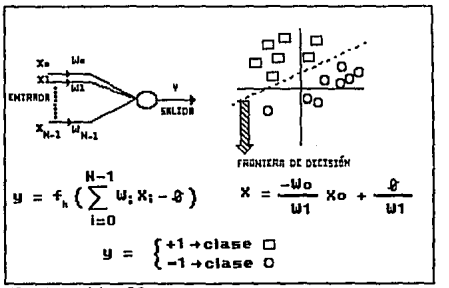


Ilustración 22.

El nodo sencillo calcula la suma de los pesos de los nodos de entrada restando el umbral (Θ) y pasa el resultado por un limitador rígido no lineal de tal manera que la salida solo puede ser +1 o -1. La regla de decisión sirve para saber a que tipo de clase pertenece la entrada. Si la salida es +1 será clase cuadro y si es -1 será clase círculo.

Una técnica útil para analizar el comportamiento de redes como el perceptrón es mapear las elecciones de decisión creadas en un espacio multidimensional por las variables de entrada. Estas regiones especifican cuáles valores de entrada caen dentro de la clase A (o en este caso cuadro) y cuáles dentro de la clase B (círculos, para nuestro ejemplo). El perceptrón forma dos regiones de decisión separadas por un hiperplano. Estas regiones de decisión se muestran en parte derecha del esquema anterior, cuando hay únicamente dos entradas y el hiperplano es una línea. En este caso las entradas que estén por encima de la línea pertenecerán a la clase A y las que estén por debajo pertenecerán a la clase B. Como se puede observar, la ecuación de la línea de frontera depende de los pesos de conexión y del umbral.

Los pesos de conexión y el umbral en un perceptrón pueden fijarse utilizando diferentes algoritmos. El procedimiento de convergencia original para ajustar los pesos fue desarrollado por Rosenblatt y se describe en el recuadro posterior. Primero los pesos y el umbral son inicializados a un valor aleatorio pequeño diferente de cero. Entonces una nueva entrada con N elementos continuos es aplicada y la salida se calcula como se mostró en la figura anterior. Los pesos son adaptados únicamente cuando un error ocurre usando la formula en el paso 4. Esta ecuación

incluye un término de ganancia (η) que tiene un rango de 0.0 a 1.0 y controla el grado de adaptación.

Un problema con el proceso de convergencia del perceptrón es que la frontera de decisión puede oscilar continuamente cuando las entradas no son separable o las distribuciones se traslapan. Una solución sería el minimizar el error cuadrático medio entre la salida deseada y la salida actual de la red. Este algoritmo, llamado de Widrow-Hoff o LMS, el cual ya ha sido deducido y analizado en §3.5.2.2 es idéntico al procedimiento descrito en el recuadro excepto que el limitador rígido se hace cuasi-lineal al ser reemplazado por un umbral tipo lógico.

Los pesos son entonces corregidos en cada proceso por una cantidad que depende de la diferencia entre la salida deseada y la salida actual. Un clasificador que usa este procedimiento de entrenamiento podrá utilizar valores de 1 para una clase A y de 0 para una clase B. Durante el proceso, la entrada será asignada a clase A si y solo si la salida tiene un valor mayor a 0.5.

Paso 1. Inicializar pesos y umbral

Asignar $W_i(0)$ $(0 \leq i \leq N-1)$ y Θ valores aleatorios pequeños. Aquí, W_i es el peso de la entrada i en el tiempo t y Θ es el umbral en el nodo de salida.

Paso 2. Presentar Nueva entrada y salida deseada

Presentar nuevos valores de entrada continuos X_0, X_1 ... X_{n-1} con la salida deseada d(t)

Paso 3. Calcular salida actual

$$y(t) = f_h \left(\sum_{i=0}^{N-1} W_i(t) X_i(t) - \Theta \right)$$

Paso 4. Adaptar pesos

$$W_i(t+1) = W_i(t) + \cap [d(t)-y(t)]X_i(t), \quad 0 \leq i \leq N-1$$

$$d(t) = \begin{cases} +1 \text{ si la entrada pertenece a la clase A} \\ -1 \text{ si la entrada pertenece a la clase B} \end{cases}$$

Paso 5. Repetir a partir del paso 2.

Algoritmo de entrenameinto para el Perceptón.

3.5.3.4 El Perceptrón Multicapa

Los perceptrones multicapa son redes con alimentación hacia adelante con una o más capas de nodos entre la entrada y la salida. Estas capas adicionales contienen unidades ocultas o nodos que no están directamente conectados a los nodos de entrada y salida. Un perceptrón tres capas con dos capas de ocultas se muestra en la figura.

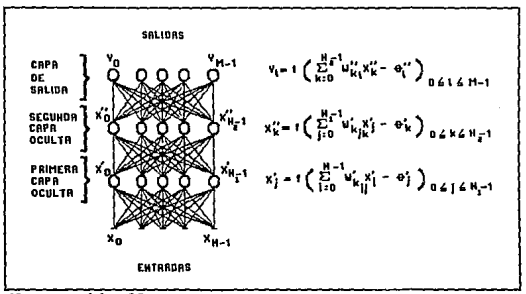


Ilustración 23.

El perceptrón multicapa supera muchas de las limitaciones del perceptrón de una capa, pero en el pasado no fue utilizado debido a que no existían aún los algoritmos para su entrenamiento. Esto ha cambiado recientemente con el desarrollo de nuevos algoritmos, aunque éstos no pueden proporcionar una convergencia como en el perceptrón uni-capa, si han demostrado ser eficientes y exitosos en interesantes problemas, (Lippmann, 1987).

Las características de un perceptrón multicapa se basa en las no linealidades utilizadas entre sus nodos. Si los nodos fueran elementos lineales, entonces una red uni-capa con los pesos escogidos apropiadamente podrá funcionar de igual manera que cualquier red multicapa. Las características de perceptrones con una, dos y tres capas que utilizan un limitador rígido se ilustran en la figura siguiente. La segunda columna en la figura indica los tipos de regiones de decisión que pueden formarse utilizando diferentes redes. Las siguientes dos columnas presentan ejemplos de regiones de decisión para el problema del OR exclusivo y con regiones que se "enredan o embonan". La última columna muestra las regiones más generales que pueden formarse.

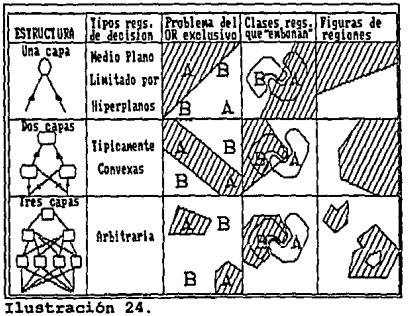


Ilustración 24.

Como puede notarse, un perceptrón de una capa puede formar regiones de medio plano. Uno de dos capas, puede formar cualquier región convexa (la cual está formada por intersecciones de regiones de medio plano formadas por cada nodo en la primera capa del perceptrón multicapa) en la hiperespacio de las entradas.

Finalmente, no se necesitan más de tres capas en redes tipo perceptrón, ya que con tres capas se pueden generar regiones complejas arbitrarias.

Toda la discusión hecha hasta el momento, se centra en perceptrones multicapa cuando se utilizan limitadores rígidos a la salida. Un comportamiento semejante se observa cuando se utilizan funciones sigmoidales y la regla de decisión es hecha para seleccionar la clase correspondiente al nodo de salida con el mayor valor. Sin embargo, el comportamiento de estas redes es más complejo debido a que las regiones de decisión son típicamente limitadas por curvas "suaves" en lugar de segmentos de rectas y el análisis es entonces más complejo. Estas redes pueden ser entrenadas, sin embargo, con el algoritmo de **back-propagation o retro-propagación.**

El algoritmo de retro-propagación es una generalización del algoritmo LMS. Usa una técnica de búsqueda del gradiente para minimizar un "costo" de función que es igual a la diferencia cuadrática media entre la salida deseada y la salida actual. Requiere que las funciones no lineales para los umbrales (generalmente sigmoides) sean diferenciales. La salida deseada de todos los nodos es típicamente "bajo" (0 o <0.1) menos el nodo correspondiente a la clase de la entrada actual, cuyo valor será "alto" (1.0 o >0.9). La red es entrenada inicialmente con valores aleatorios pequeños para los pesos y umbrales internos,y entonces se presentan todos los datos de entrenamiento repetidamente. Los pesos son ajustados después de cada ciclo hasta que el "costo" del la función se reduzca a un valor aceptable. Un punto esencial del algoritmo es el método iterativo que propaga los términos del error necesarios para adaptar los pesos hacia atrás, esto es, de los nodos de salida a los nodos de las capas anteriores.

El algoritmo ha sido probado en un gran número de problemas determinísticos tales como problemas del OR exclusivo, o problemas relacionados con la síntesis y reconocimiento de voz, y problemas de reconocimiento de patrones visuales.

Una demostración del poder de este algoritmo fué proporcionada por Sejnowski. El entrenó un perceptrón de dos capas con 120 nodos ocultos y más de 20,000 pesos con reglas para transformar de la forma escrita a fonemas. La entrada a la red fué un código binario que indicaba las letras que se encontraban dentro de un "ventana" de 7 caracteres y que se desliza a través del texto escrito. La salida fué un código binario que indicaba la transcripción fonética de la letra en el centro de la ventana. Después de 50 ciclos a través de un texto de 1024 palabras el porcentaje de error de la transcripción fué del 5% y de un 22% para un texto que no fué utilizado en el entrenamiento.

Paso 1. Inicializar pesos y Offsets

Asignar a todos los pesos y a los offsets de nodos, valores aleatorios pequeños.

Paso 2. Presentar la salida y entrada deseadas

Presentar un vector de valores continuos a la entrada X_0, X_1 .. X_{N-1} y especificar la salida deseada d_0, d_1 ...d_{M-1}. Todas las salidas son cero excepto la correspondiente a la clase de la entrada, la cual será puesta en 1. La entrada se presenta cíclicamente hasta estabilizar todos los pesos.

Paso 3. Calcular las salida actuales

Utilizar una función tipo sigmoide logística, $f(\alpha)=1/[1+e^{-(\alpha-\theta)}]$, y las ecuaciones de la figura "Perceptrón de tres capas" para calcular $y_0, y_1 .. y_{M-1}$.

Paso 4. Adaptar Pesos

Usar un algoritmo recursivo comenzando en los nodos de salida y trabajando hacia atrás hasta la primer capa oculta. Ajustar los pesos como se especifica:

$$W_{ij}(t+1) = W_{ij}(t) + \mu\delta_j X'_i$$

W_{ij} es el peso del nodo oculto i o de una entrada al nodo j en el tiempo t, X'_i es la salida del nodo i o una entrada, μ es un término de ganancia, y δ_j es un término de error para el nodo j. Si el nodo j es de salida, entonces

$$\delta_j = y_j(1-y_j)(d_j-y_j),$$

donde d_j es la salida deseada del nodo j y y_j es la actual. Si j es un nodo oculto, entonces

$$\mu_j = X'_i(1-X'_i)\sum_k \delta_k W_{jk},$$

donde k es para todos los nodos abajo del nodo j.Los umbrales de nodos internos se adaptan similarmente asumiendo que son pesos sobre ligas auxiliares de valores constantes de entrada. La convergencia es más rápida en ocasiones si un término de "momento" (momentum) se agrega y los pesos varían más suavemente de acuerdo a ...

$$W_{ij}(t+1) = W_{ij}(t) + \mu\delta_j X'_i + \alpha(W_{ij}(t)-Wij(t-1))$$

Paso 5. Repetir a partir del paso 2.

Algoritmo de entrenamiento por Retro-propagación.

3.6 Conclusiones

El reconocimiento de voz, requiere de diferentes redes para diferentes tareas. Hemos revisado diferentes redes y mostramos que un perceptrón de tres capas puede formar regiones de decisión cualquier forma. Estas redes neuronales clasificadoras funcionan aún mejor que los clásicos clasificadores Gaussianos para problemas de clasificación de dígitos, funcionando también adecuadamente para la clasificación de vocales.

El esfuerzo de la investigación actual en redes neuronales ha atraído a investigadores en ingeniería, física, matemáticas, neurociencias, biología, computación y psicología. La investigación actual se enfoca en el análisis de algoritmos aprendizaje y auto-organización usados en redes multicapa, en el diseño de principios y técnicas para resolver problemas de dinámica y sensitividad, en la construcción de sistemas completos para el reconocimiento de voz e imagen, en obtener experiencia de éstos para determinar cuáles de los algoritmos actuales pueden implementarse utilizando componente "pseudo-neuronales". Los avances en estas áreas, así como técnicas de implementación VLSI conducirán a sistemas neuronales prácticos y con capacidad de proceso en tiempo real.

CAPITULO IV

MEMORIA DISTRIBUIDA ESPARCIDA

MEMORIA DISTRIBUIDA ESPARCIDA

4.1 Introducción

El éxito futuro de las red las redes neuronales depende de su habilidad para "ascender" de pequeñas redes y problemas de "juguete" de pequeñas dimensiones a redes de cientos de millones de nodos y problemas de grandes dimensiones propios del mundo real[1]. A menos que las redes neuronales muestren que pueden ascender a problemas reales, quedarán muy probablemente restringidas a pocas aplicaciones especializadas.

Además de este ascenso, los sistemas de cómputo deben tomar en cuenta dos tipos de demandas que se hacen por demás necesarias. Primero, existe un incremento lineal en la demanda de cómputo proporcional al incremento del número de variables manejadas. Segundo, que es aún más grave, existe un incremento no lineal en la demanda del número de interacciones que pueden ocurrir entre las variables. Este efecto secundario es el responsable principal de los tropiezos para que muchos sistemas "crezcan al mundo de tareas **reales**".

[1] La "dimensionalidad" de un problema se refiere al número de variables necesarias para describir el "dominio" del problema.

Dos sistemas han demostrado un buen funcionamiento en problemas con dominios de grandes dimensiones, la **Memoria Distribuida Esparcida de Pentti Kanerva y el Algoritmo Genético de Holland.**

La Memoria Distribuida Esparcida o MDE (SDM para siglas en inglés) es un modelo conexionista formal que actúa como memoria asociativa y que está basado en propiedades de un espacio multi-dimensional de direcciones binarias. Puede representarse como una red neuronal de tres capas con un número extremadamente grande de nodos (más de un millón) en la capa intermedia. En su diseño original, las conexiones entra la capa de entrada y la capa oculta son fijas y el aprendizaje se hace cambiando los valores de los pesos entre la capa intermedia y la salida.

Este modelo neuronal resulta bastante eficiente para la implementación de un sistema capaz de realizar reconstrucción de patrones ruidosos (tal como una red de Hopfield o una Memoria Asociativa Bidireccional -BAM-) o para actuar como clasificador de patrones, (Rogers).

4.2 Modelo de la Memoria Distribuida Esparcida.

La Memoria Distribuida Esparcida o MDE es un modelo de memoria relacionado con las redes neuronales. Es un modelo simple pero elegante de una memoria direccionable por

contenido (Kanerva, 1988). La MDE tiene la capacidad de recordar y obtener patrones asociadas cuando se presentan patrones incompletos distorsionados por ruido cuando se entrena sin supervisión. La ventaja, es que también funciona adecuadamente cuando esta red "aprende" a través de un entrenamiento supervisado y se utiliza entonces como clasificador.

Una MDE puede ilustrase mejor como una variante de un algoritmo usado comúnmente para implementar una memoria de acceso aleatorio. La estructura de una memoria de este tipo conocida mejor por su siglas como RAM se muestra en el esquema inferior.

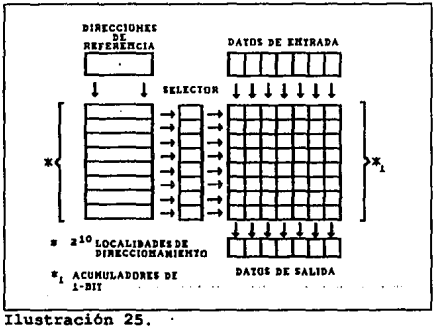


Ilustración 25.

La dirección en la cual se está leyendo o escribiendo puede ser llamada ***dirección de referencia***. La memoria compara esta dirección contra la dirección de cada localidad de memoria.

La localidad que sea igual a la dirección de referencia se selecciona, la cual se denota con un "1" en el vector de selección.

Si se está realizando una escritura en la memoria, el dato de entrada se substituye y entonces queda almacenado en el registro de n bits de la dirección seleccionada.

Si una lectura esta siendo realizada desde la memoria, el contenido del registro seleccionado es transmitido sobre el bus de datos y entonces la información se hace disponible como datos de salida.

Ahora bien, la Memoria Distribuida Esparcida puede considerarse como una extensión de la memoria de acceso aleatorio. La arquitectura de un simulador de un MDE standard se muestra en la figura.

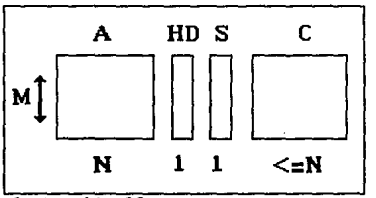


Ilustración 26.

En la figura, A es un arreglo de direcciones. Usualmente el ancho de A es igual a N, es decir, el ancho del patrón que se desea almacenar en la memoria. Debido a este ancho de N bits, el número total de patrones posibles es de 2^N. Sin embargo, en la práctica, los M renglones de A son muestras aleatorias del total de 2^N posibles direcciones. M es siempre mucho menor que 2^N. P es un vector de N elementos en el cual se introduce el patrón a ser procesado. HD es un vector de M elementos que almacena M distancias de Hamming calculadas entre el patrón del vector P y cada renglón del arreglo A de direcciones aleatorias. S es un vector de selección de longitud M en el cual las direcciones de los K patrones más parecidos son puestas en "1" o elegidas. C es un arreglo cuyo ancho puede ser menor o igual a N y debe tener M renglones. Además M puede ser menor igual a las N columnas creadas para almacenar la información del patrón de entrada.

En cada una de las tres operaciones que una MDE es capaz de realizar (direccionamiento, lectura y escritura) existen variaciones y ventajas sobre una RAM.

En lugar de "buscar" un parecido exacto (igualación) entre la dirección de referencia y las direcciones de las localidades, la memoria calcula la distancia de Hamming entre la dirección de referencia y cada una de las direcciones. Cada distancia se compara entonces con un "radio determinado" (el cual se escoge de tal forma que solo un pequeño porcentaje del total de las localidades de memoria sean seleccionadas para una dirección de referencia dada); si la distancia es menor o igual que el radio, la localidad se selecciona. **Más de una localidad se selecciona generalmente en este proceso.**

Los registros de datos se convierten ahora en "contadores" en lugar de elementos de almacenamiento de bits únicamente. Estos contadores serán elementos de n-bits, incluyendo un bit para signo. Cuando se está escribiendo en las direcciones seleccionadas en lugar de hacer una "sobreescritura" (con lo cual se borraría la información anterior), la memoria incrementa el contador si el bit correspondiente de los datos de entrada es un 1, y decrementa el contador si el bit correspondiente es un 0.

Cuando se está leyendo, la memoria generalmente selecciona más de una localidad. La memoria suma los contenidos por columna de cada localidad seleccionada y entonces pasa cada suma a través de un limitador rígido. De esta manera, si la suma correspondiente a cada columna es mayor o igual a cero la salida será una salida binaria con valor de 1, y si la suma es menor que cero la salida será un 0.

El siguiente ejemplo, muestra una Memoria Distribuida Esparcida cuya longitud de dirección de referencia es de 10 bits. Los datos se **distribuyen** sobre los contadores de las localidades seleccionadas cuando se está efectuado una escritura, y esa misma información es **reconstruida** durante la lectura promediando la suma de estos contadores. Sin embargo, dependiendo de que se haya escrito información adicional en algunas direcciones seleccionadas, es decir, que algunas de las direcciones seleccionadas para un patrón de entrada coincidan con direcciones seleccionadas para otro patrón, y dependiendo de la **correlación de estos datos con los datos originales**, la reconstrucción puede contener ruido.

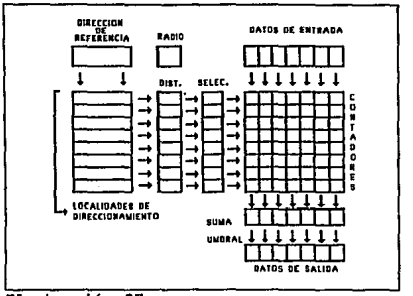


Ilustración 27.

A pesar de que en muchas ocasiones el patrón recuperado puede ser otro patrón almacenado y que no es el deseado, o de que en otras el patrón de salida sea una combinación de patrones con características semejantes, este problema se puede solucionar substancialmente. Si el patrón está muy deformado o "tiene mucho ruido", el patrón de salida P, puede usarse nuevamente como patrón de entrada y repetir este paso hasta que el P no cambie, esto es, la red ha llegado a un **estado estable de mínima energía.**

Como podemos observar al realizar esta retroalimentación, la red es potencialmente más poderosa y con porcentajes de error bastante más pequeños que sin retroalimentación.[2]

[2] La idea se nos ocurrió originalmente a raíz de trabajos anteriores con una Memoria Asociativa Bidireccional, la cual hace uso de la retroalimentación. No obstante, posteriormente nos dimos cuenta que en el artículo de Alvin Surkan y Liping Di, de la Universidad de Nebraska, proponían esta misma idea.

La MDE consiste de un conjunto de direcciones , X_μ, escogidas "esparcidamente" en un espacio de 2^n, donde n es del orden de 1000. En cada una de estas localidades, existen m contadores, $X(j,X_\mu)$, donde m es el número de bits de la palabra. Cuando escribimos la palabra Y en X se sigue la siguiente regla de aprendizaje :

Para toda $|X-X_\mu|_H < h$

incrementar $C(j,X_\mu)$ si $Y_j = 1$
decrementar $C(j,X_\mu)$ si $Y_j = 0$

donde $|X-X_\mu|_H$ se refiere a la distancia de Hamming entre las direcciones X y X_μ y Y_j es el j-ésimo bit de la palabra Y. El parámetro h es usualmente 450 bits para n = 1000.

La lectura en una dirección X se hace de acuerdo a la regla siguiente :

$$Y_j = 1 \text{ si } \sum_{|X-X_\mu|_H<h} C(j,X_\mu) >= 0$$

$$Y_j = 0 \text{ si } \sum_{|X-X_\mu|_H<h} C(j,X_\mu) < 0$$

Algoritmo de entrenamiento de la MDE.

A continuación mostramos diversos ejemplos del comportamiento de la Memoria Distribuida Esparcida para el reconocimiento de patrones. El simulador de la MDE que obtuvo los resultados presentados fué hecho por A. Enríquez y A. Tokun Haga para demostrar la funcionalidad y potencia que ofrece este modelo para este tipo de tareas.

Se entrenaron cuatro patrones de 64 bits, de esta manera el tamaño de palabra de memoria, de dirección de referencia y de direcciones de localidades fué de 64 bits. Un radio de 15 bits y un total de 250 direcciones.

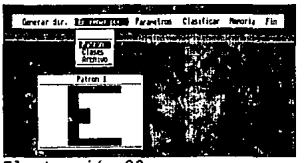


Ilustración 28.

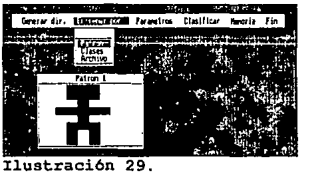


Ilustración 29.

Ilustración 30.

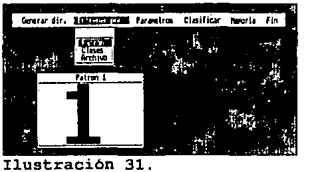


Ilustración 31.

Después de hacer una sola presentación de los cuatro patrones a la red, se intentó recuperar el patrón "S" deformado. El resultado se presenta en el esquema. El patrón izquierdo representa la entrada y el derecho la salida de la MDE.

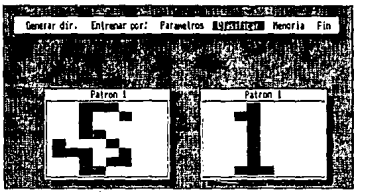


Ilustración 32.

Como se puede notar la red llega a una solución errónea. No obstante, volvimos a entrenar el patrón, obteniéndose el resultado siguiente.

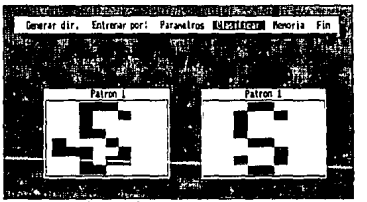


Ilustración 33.

A pesar de que el resultado anterior fué correcto, no tuvimos éxito al tratar de reconstruir el patrón "hombre" el cual había sido deformado invirtiendo algunos bits. Además de obtener nuevamente una solución espuria, el sistema entró en un "loop" (o ciclo) ya que no encontró algún estado estable y al retroalimentar la salida se obtenía nuevamente la entrada original; en otras palabras, la entrada X produce la salida Y, y la entrada Y (que salida en el estado anterior) produce la salida X. Desde el punto de vista de la dinámica del sistema, la red no encontró un estado estable de mínima energía y por tanto oscilaba de un posible estado mínimo a otro que parecía tener igual posibilidad.

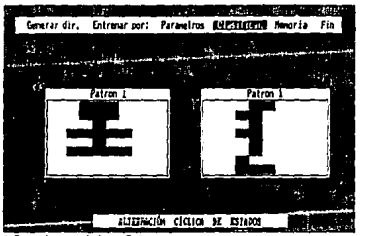


Ilustración 34.

La manera de resolver este problema fue entrenando más a la red. La siguiente tabla muestra el número de presentaciones totales de cada patrón para que la tuviera un comportamiento aceptable, el cual se ejemplifica con las ilustraciones posteriores.

PATRÓN	PRESENTACIONES
E	2
I	2
S	3
HOMBRE	3

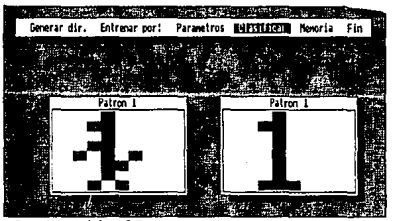


Ilustración 35.

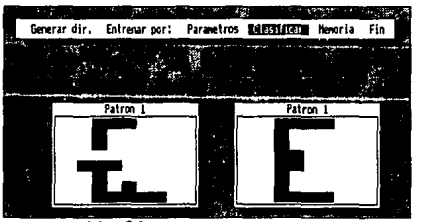


Ilustración 36.

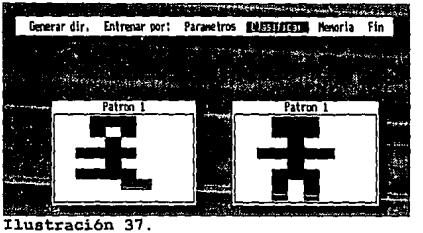


Ilustración 37.

4.2.1 Modelo Neuronal de un Memoria Distribuida Esparcida

A través de la analogía con una RAM, es tal vez la manera fácil explicar la estructura de una MDE, no obstante también se puede describir como una red neuronal de tres capas totalmente conectada y con alimentación hacia adelante, es decir, como un perceptrón de tres capas. Una modelo neuronal equivalente para la MDE se muestra en la figura.

La capa inferior es donde la dirección de referencia es dada; esto es, hay un nodo en esta capa para cada bit de la dirección de referencia. Estos nodos tienen un valor de 1 o -1 dependiendo de si el correspondiente bit de la dirección de referencia es 1 o es 0.

Las conexiones entre la capa inferior y los nodos de la llamada "capa intermedia" tienen un peso de 1 o -1. Estos pesos nunca cambian, así, ellos determinan la dirección de las localidades físicas de la memoria.

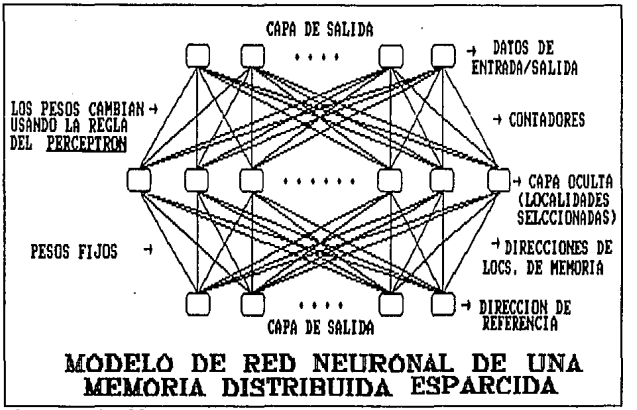


Ilustración 38.

Cada uno de los nodos en la capa intermedia corresponde a una localidad de memoria en el modelo MDE. Una localidad de memoria es seleccionada si la suma de la suma de sus entradas (esto es, el producto punto de la dirección de referencia y el vector de pesos de la localidad) es mayor o igual a su umbral. Este umbral corresponde al radio del modelo MDE, y la suma de las entradas es efectivamente tomando la distancia de Hamming entre la dirección de la localidad de memoria y la dirección de referencia.

La capa superior es donde aparecen los datos de salida. Cada nodo de la capa oculta está completamente conectado a los nodos de la capa superior. Los cantadores de una localidad de

memoria se representan como los pesos de las conexiones entre la capa oculta y los nodos de salida.

La lectura de la memoria implica colocar valores en la dirección de referencia y leer la salida de los nodos de salida. Escribir en la memoria, por otra parte, implica colocar los valores deseados tanto en la dirección de referencia como en los nodos de entrada;los nodos internos que están activos agregan entonces el valor de cada nodo de entrada (1 o -1) a sus pesos.

En esta forma, la MDE es muy similar a otras arquitecturas de redes neuronales. Sin embargo para una MDE el número de nodos en la capa oculta es mucho más grande que los usados comúnmente por las otras redes neuronales. Un tamaño razonable puede tener una logitud de dirección y de dato de 1,00 bits el cual corresponde a 1,000 nodos en la capa inferior y en la superior. Esto es muy grande, pero no mayor a las capacidades de los algoritmos neurocomputacionales actuales. Sin embargo, si la memoria tiene 1,000,000 de localidades de memoria, esto corresponderá a una red con 1,000,000 de nodos en la capa intermedia. Y es aquí donde la MDE tiene su mayor potencial, ya que todavía no queda claro, cómo algoritmos estándares, tales como el **back-propagation**, pueden funcionar cuando se tiene un número tan grande de unidades en la capa oculta.

4.2.2 La Memoria Distribuida Esparcida como clasificador

Como hemos mencionado con anterioridad, una MDE puede funcionar además de "reconstructor de patrones ruidosos", como clasificador. De hecho, al actuar como clasificador la memoria se hace más resistente al ruido al no tener que obtener como salida un patrón que sea exactamente el asociado a una entrada determinada; es decir, supongamos que tenemos un patrón sumamente deformado ("ruidoso") a la entrada de la memoria y que ésta ha sido entrenada previamente con un número determinado de patrones, entonces la salida puede ser en el mejor de los casos el patrón de salida asociado deseado, pero puede suceder también que la salida sea un patrón "espurio" resultado de la combinación de patrones almacenados y que en muchas ocasiones es muy parecido al patrón deseado excepto que difiere en unos pocos bits[3]. Si el resultado fuera este, podríamos decir que la red ha errado en su resultado. Sin embargo, cuando una MDE funciona como clasificador, la red definirá como salida aquella clase que más se asemeje a la entrada dada de acuerdo a un previo entrenamiento. De esta manera, o falla totalmente o tiene éxito totalmente, pero no dará nunca como resultado un clase que sea la combinación de otras dos clases. De esta manera, el grado de resistencia al ruido se hace mucho mayor.

Cuando la MDE tiene un entrenamiento supervisado para la clasificación de patrones, el

---

[3] Aunque hemos comprobado experimentalmente que ésto depende mucho del grado de entrenamiento que haya tenido la red, ya que con presentar a la red aproximadamente n veces cada patrón, donde n es el número total de patrones al almacenar, la respuesta es satisfactoria con porcentajes mayores al 85%.

algoritmo se modifica ligeramente. Supongamos que los patrones entrenados pertenecen a un total de N clases, entonces el tamaño de la palabra de memoria deberá ser de N también y cada **columna del arreglo corresponderá a una clase.** Durante el proceso de entrenamiento, K direcciones se obtienen calculando la distancia de Hamming entre las direcciones de las localidades y el patrón a almacenarse (dirección de referencia) perteneciente a la clase **i.** Entonces, K localidades de memoria tendrán agregado ya sea 1 o -1 en la **columna i.** Durante el proceso de clasificación, el patrón a ser clasificado se compara con cada uno de las direcciones y nuevamente K direcciones serán seleccionadas. Las K localidades seleccionadas se suman por columna para obtener un vector **V** de dimensión N (un elemento por cada suma-columna). El patrón pertenece a la clase i si el elemento i-ésimo del vector **V** tiene el máximo valor de todos los elementos. Si el máximo valor es cero, el patrón pertenece a una clase desconocida o la red no puede clasificarlo entre las clases entrenadas.

Las ilustraciones siguientes muestran el comportamiento de la Memoria Distribuida Esparcida como clasificador. Las imágenes fueron tomadas del sistema desarrollado descrito en el capítulo subsecuente.

Se entrenaron 3 señales de voz, correspondientes a las palabras "estella", "galaxia" y "universo" digitalizadas y tomando un total de 512 muestas por palabra. El entrenador fué un hombre, pero como veremos posteriormente, este sistema ha demostrado tener la capacidad de reconocer señales de voz con independencia del parlante.

El número de entrenamientos necesarios para lograr un porcentaje de reconocimiento aceptable fué de 3 entrenamientos para cada patrón de voz, lo cual es congruente con el modelo matemático presentado en el capítulo V sección §5.4. Según este modelo, se necesitan n^2 entrenamientos para n patrones de voz a reconocer.

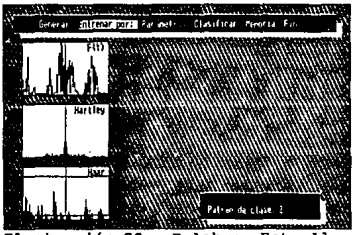


Ilustración 39. Palabra: Estrella.

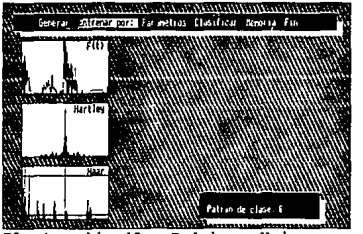


Ilustración 40. Palabra: Universo.

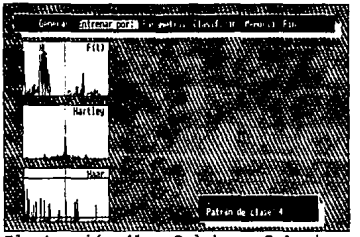


Ilustración 41. Palabra: Galaxia.

Patrón a reconocer:	Galaxia
Patrón reconocido:	Galaxia
Habla:	Hombre

Patrón a reconocer:	Estrella
Patrón reconocido:	Estrella
Habla:	Hombre

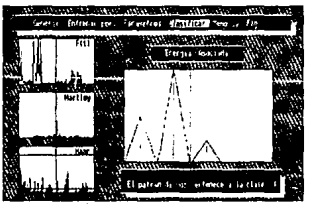


Ilustración 42.

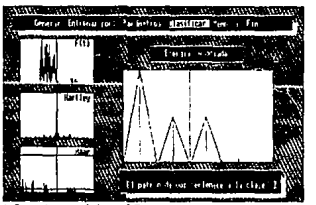


Ilustración 43.

Patrón a reconocer: Universo
Patrón reconocido: Universo
Habla: Mujer

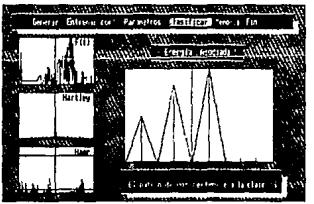


Ilustración 44.

Patrón a reconocer: Galaxia
Patrón reconocido: Estrella
Habla: Mujer

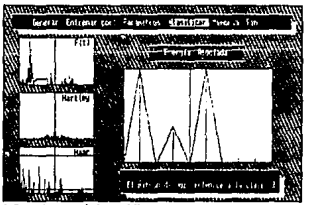


Ilustración 45.

Nótese que aún cuando el entrenador fué un hombre, al hablar la mujer también reconoce la señal, aunque no con el mismo porcentaje. Para aumentar el grado de reconocimiento es suficiente con hacer un entrenamiento mixto un poco más exhaustivo.

La idea más importante de la Memoria Distribuida Esparcida es que muchas localidades participan en una operación de escritura con lo cual la información almacenada queda **distribuida**

en la toda la memoria y no se encuentra localizada en un solo lugar[4]. De esta forma, si una palabra τ es almacenada en la dirección ß, leyendo desde la dirección ß se obtiene τ, y lo que es más importante, leyendo desde una dirección x la cual es suficientemente similar a ß se obtiene una palabra z la cual es más similar a τ de los que es x a ß.

4.3 Conclusiones

Podemos pronosticar que el éxito o fracaso de una MDE dependerá en primer término de la correlación que exista entre el patrón a almacenarse y el conjunto de direcciones de localidades de la memoria.

A pesar de su aparente sencillez, ha demostrado que es un modelo muy poderoso, y en muchas ocasiones funcionando aún mejor que los tradicionales modelos neuronales, en tareas relacionadas con el reconocimiento de patrones.

En el capítulo siguiente, veremos cómo utilizamos esta memoria para el diseño e implementación de un sistema "real" capaz de clasificar señales de voz para palabras aisladas.

---

[4] Se ha demostrado que en el cerebro aún cuando existen ciertas regiones asociadas a tareas especificas y totalmente determinadas, existe una redundancia de información tal, que al no encontrase la información centralizada en un solo lugar del cerebro, hace que el cerebro sea relativamente resistente a pequeños averías. Así si algunas cuantas neuronas se mueren o dañan, no afectarán significativamente al comportamiento global de la red de neuronas y la información que ellas representaban o contenían no se perderá tampoco.

CAPITULO V

DISEÑO DE UN SISTEMA PARA EL RECONOCIMIENTO DE VOZ

DISEÑO DE UN SISTEMA PARA EL RECONOCIMIENTO DE VOZ

5.1 Introducción

Tomando en cuenta la complejidad que involucra el reconocimiento del lenguaje natural, y la gran importancia que representa el realizarlo de forma automática por alguna máquina, se diseñó e implementó un sistema por medio del cual se esperaban obtener resultados satisfactorios en cuanto a capacidad de reconocimiento, y que además fuera práctico, fácil de manejar, sin utilizar tecnología muy compleja y sobre todo sin incurrir en grandes costos.

Es cierto que este problema ya ha sido tratado en muchas ocasiones pero con ideas y herramientas convencionales y generalmente costosas. De esta manera el sistema desarrollado tiene como característica el no utilizar tales conceptos, permitiéndonos asimismo obtener adecuados porcentajes de reconocimiento no muy lejanos de los obtenidos por los sistemas anteriores.[1]

El sistema está integrado por dos subsistemas principales, una etapa de hardware (circuito

[1] *Para mayor detalle, refiérase al capítulo I Historia y antecedentes del reconocimiento del lenguaje.*

electrónico que es una interfase entre el humano que habla y la computadora que recibe la señal de voz ya digitalizada) y otra de software (simulador de un memoria distribuida esparcida).

5.2 El circuito electrónico

Un circuito electrónico (hardware) obtiene y transforma la señal de voz para ser manipulada por una computadora mediante diferentes algoritmos y técnicas computacionales (software). La figura siguiente muestra, utilizando un diagrama de bloques, cada uno de los módulos que constituyen nuestro sistema para el reconocimiento de voz.

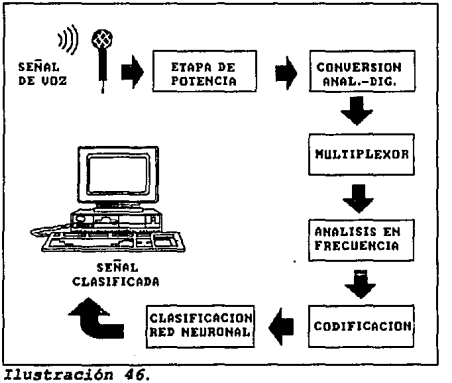


Ilustración 46.

5.2.1 Señal de entrada

La señal de entrada consiste en una señal eléctrica continua que se obtiene al pasar a través de un transductor (micrófono) una señal de voz. La señal eléctrica representará entonces la amplitud y frecuencias características de la señal de voz en cuestión. Cabe hacer notar que depende en mucho la calidad de la señal de entrada de la calidad e impedancia del micrófono, ya que para un micrófono de mala calidad las deformaciones de la señal de salida con respecto a la señal de voz de entrada al micrófono pueden ser muy grandes.

5.2.2 Etapa de amplificación de la señal

Debido a que la señal eléctrica obtenida tiene una amplitud muy pequeña, y por ende, no es posible manipularla, es necesario amplificarla y proporcionarle una *ganancia en corriente.*

La señal fue amplificada aproximadamente cien veces, de tal manera que el valor máximo de la señal de entrada fuera de cinco volts (5 V, corriente de directa). Este valor máximo fue determinado por las especificaciones eléctricas del circuito integrado utilizado para la conversión analógica digital (A-D) de la señal.

5.2.3 Conversión analógica-digital

Para esta etapa, el valor máximo de la señal recibida debería ser aproximadamente 5 volts, para poder ser transformada a una señal digital con una resolución de ocho bits. Esto se debe a que el valor de salida proporcionado por el convertidor es una *relación entre el valor de referencia (fijo e igual a 5 volts) y la señal de entrada*, de esta forma, el valor máximo teórico que pudiese ser obtenido sería de 1. No obstante, con ocho bits el máximo valor que se puede obtener es 0.9961, es decir, aún cuando la señal de entrada se mayor o igual a 5 volts, el valor resultante nunca podrá llegar a formar la unidad. Por todo lo anterior debe tomarse en cuenta que se está incurriendo en un error debido a la resolución misma del circuito integrado utilizado.

Por otra parte, fue necesario determinar el número de puntos al cual tendría que ser discretizada la señal de voz, de tal manera que fueran los puntos suficientes para tener una adecuada representación discreta (y posteriormente digital) de la señal analógica de entrada, pero asimismo no tantos como para que el tiempo y todos los demás recursos utilizados en su manipulación se incrementaran demasiado. Por tanto, se determinó a través de numerosas pruebas que 512 puntos eran adecuados.

Finalmente cabe mencionar, que el temporizador utilizado para la sincronización de la conversión oscilaba a una frecuencia de 10 KHz. Esta frecuencia se obtuvo aplicando el Teorema de Nyquist, ya que partiendo de la premisa de que el ancho de banda de una señal de voz es de

20 Hz - 4 KHz, entonces la frecuencia mínima para la digitalización viene dada por:

$$f_{min-dig} = 2 * f_{max-analog}$$
$$= 2 * 4KHz = 8KHz$$

De esta manera, una frecuencia de muestreo de 10 KHz resultó adecuada para nuestros propósitos.

5.2.4 Multiplexaje

La información de la señal de voz una vez digitalizada, se pasa a la computadora a través del puerto paralelo. Este puerto consta de tres puertos lógicos de 8 bits cada uno. El puerto de salida está constituido por los pines 2-10, el cual se ocupó para transmitir señales de control de la computadora al circuito electrónico, mientras que el puerto de entrada por medio del cual la información es transmitida del circuito a la computadora, está integrado por los pines 10-13 y el 15. Como se observa, solo 5 bits pueden ser manipulados para la entrada ya que los otros 3 (los menos significativos) son implícitos y tienen un estado de 1 lógico.

Es precisamente por esta causa que fue necesario hacer uso de un multiplexor para que se pudieran pasar a la computadora los 8 bits que proporcionaba el convertidor analógico digital.

De esta manera, se transmitían primero los 4 bits más significativos y posteriormente los 4 menos significativos.

Una vez leídos los 8 bits, se obtiene su representación en base 10 y a dicho número se le divide entre 256 para obtener el valor que representaría la relación del valor de señal de entrada entre el valor de referencia (5 volts), esta operación ofrece el mismo resultado que convertir los ocho bits (tomándolos como una cantidad fraccionaria) a su valor decimal (fraccionario).[2]

5.3 Análisis de los datos y simulación de MDE

5.3.1 Análisis en frecuencia

Los 512 puntos obtenidos de la digitalización de la señal fueron empleados por tres diferentes algoritmos que proporcionan la representación de la señal en el dominio de la frecuencia:

a) Transformada Rápida de Fourier (FFT)

b) Transformada Rápida de Hartley (HT)

c) Transformada Rápida de Haar (HrT)

Como ya se ha mencionado en capítulos anteriores, el análisis en frecuencia de una señal es

[2] *Ver apéndice I para su demostración.*

una herramienta muy poderosa para determinar cualidades y características particulares de una señal (propiedades que en el dominio del tiempo en muchas ocasiones no se pueden observar ni mucho menos definir). Es por ello que se decidió analizar la señal con estos diferentes métodos.

Tanto la transformada de Fourier como la de Hartley son adecuadas para señales de tipo analógicas, y aún cuando proporcionan el mismo espectro (siempre y cuando de la transformada directa de Hartley se obtenga una función de potencia[3]), la segunda resulta ser bastante más rápida cuando se tiene una gran cantidad de datos (mayor a 256 puntos).

La transformada de Haar en cambio, es adecuada para el análisis de señales digitales. Y como nuestra señal analógica de voz sufrió una transformación al ser digitalizada, en primera instancia resultaría mejor el utilizar tal tipo de transformada.

Por otra parte el método de codificación utilizado para trasladar la información a la primera capa de la MDE,como veremos en el siguiente punto, tiene características que hacen que sea también la transformada de Haar la más adecuada para este propósito. A continuación se muestran algunos ejemplos de la representación en el dominio de la frecuencia obtenida por los diferentes métodos para una misma señal de voz.

El siguiente esquema muestra una señal de voz (en el dominio del tiempo), el número de muestras tomado en consideración para su digitalización fue de 512, ya que esta cantidad de

---

[3] *Ver capitulo II Técnicas de análisis y procesamiento...*

puntos demostró ser suficiente para una buena representación de la señal y no involucraba, por otra parte, demasiado tiempo para su análisis. La palabra en consideración es *Marte*.

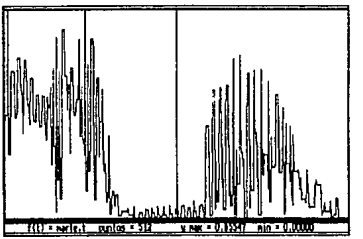


Ilustración 47. Palabra: Marte.

El espectro de la función anterior utilizando la Transformada Rápida de Fourier es el siguiente:

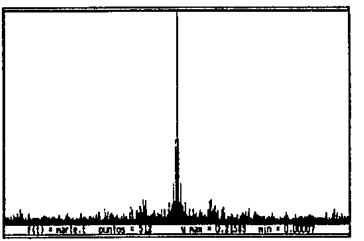


Ilustración 48. Espectro: FFT

Por otra parte la Transformada Directa de *Hartley* muestra el siguiente espectro:

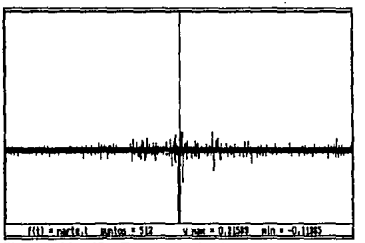


Ilustración 49. Espectro: Hartley

Al obtener el Espectro de Potencia de la transformada anterior se obtiene...

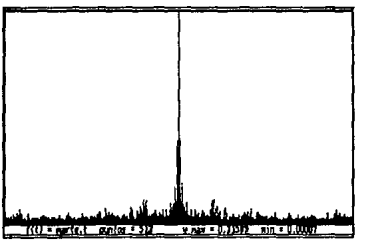


Ilustración 50.
Espectro: Potencia Hartley

... que es precisamente el mismo espectro que se obtiene por medio de la Transformada Rápida de Fourier, pero con un tiempo de cálculo bastante menor.

La ecuación para obtener el Espectro de Potencia a partir de la Transformada Directa de Hartley de una señal, se define por:

$$P = [\ (f^2_i + f^2_{n-i})/2]^{1/2}$$

donde:

$i = 1...n/2$

n = número de espigas en el espectro

f_x = componente x-ésima del espectro

Finalmente la Transformada de Haar de la misma señal tiene el este espectro:

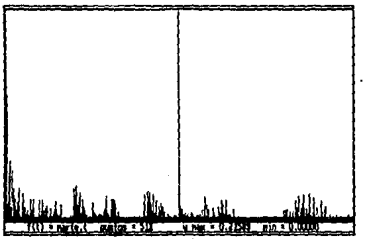


Ilustración 51. Espectro: Haar

Para ejemplificar aún más todo lo anterior se muestran las transformadas obtenidas para otra señal de voz, la palabra *Mercurio*.

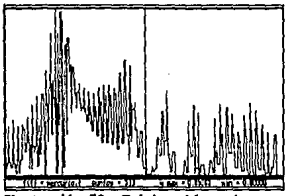


Ilustración 52. Palabra: Mercurio

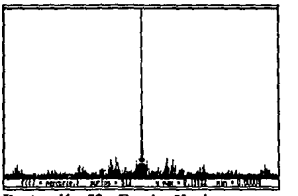


Ilustración 53. Fourier-Hartley

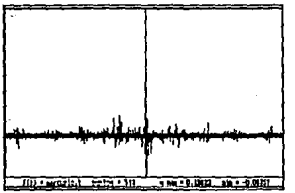


Ilustración 54. Hartley Directa

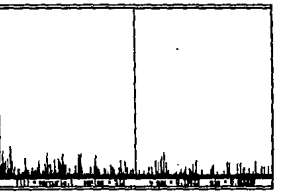


Ilustración 55. Haar

5.3.2 Codificación

El método para la codificación consistió en eliminar todas aquellas frecuencias que tuvieran

una fuerte ingerencia sobre la señal ya que éstas son generalmente características propias de la persona que habla y no propiamente de la palabra pronunciada.

Por otra parte también se eliminaron aquellas frecuencias cuya amplitud fuera muy baja ya que por lo general, están asociadas al ruido producido por el ambiente, el micrófono y sobre todo a las diferencias que existen entre la pronunciación de una persona a otra. Por consiguiente las frecuencias intermedias, son las que representan las características esenciales de la palabra pronunciada, lo cual es lo que se espera reconocer. Para obtener lo anterior fue necesario establecer dos umbrales que determinaran que espigas del espectro se consideraban y cuales se eliminaban.

A las espigas eliminadas se les asignaba un valor de cero dentro de un arreglo de bits, de tal forma que cada elemento del arreglo binario (que sería nuestro patrón) correspondiese a cada una de las espigas del espectro es decir, la espiga i-ésima corresponde al bit i-ésimo del arreglo. Las espigas no eliminadas, se les asignaba un valor de uno dentro del arreglo de bits; por lo tanto, toda la información necesaria para la clasificación de la señal por la red neuronal consiste únicamente de un arreglo de ceros y unos.

Como se puede observar el análisis de la señal se basa en transformar una señal de su representación temporal a una representación espacial, de esta forma, se están eliminado inconvenientes como el defasamiento en el tiempo y variaciones en amplitud que una misma señal de voz podría sufrir al ser pronunciada por diferentes personas y en diferentes

circunstancias.

Para obtener el valor de los umbrales, se tuvieron que realizar una gran cantidad de pruebas, para finalmente determinar que los valores más adecuados eran, para el umbral superior el noventa porciento del valor máximo de las espigas del espectro, y para el umbral inferior, el diez porciento de ésta.

Cabe mencionar que para obtener la espiga de mayor valor no se tomó en cuenta la componente de directa de la señal (es decir, cuando se tiene una frecuencia igual a cero), ya que esta componente al tener un valor mucho más grande que el de las demás, afectaba el número de espigas que pudieran estar dentro de los umbrales, y por ende, aquellas seleccionadas no representaban realmente las características propias de la palabra analizada.

Los valores de los umbrales fueron determinados por medio de las siguientes ecuaciones:

$$\text{Umbral}_{\text{superior}} = 0.9 * \max \{F\}.$$

$$\text{Umbral}_{\text{inferior}} = 0.1 * \max \{F\}$$

donde $F = \{f_1, f_2, \ldots, f_n\}$

Las siguientes figuras ilustran, de manera más clara, como los umbrales determinan las espigas que se eliminan y cuales son las que se incluyen dentro de los límites. La palabra ejemplificada es *Venus*.

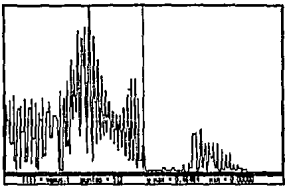


Ilustración 56. Palabra: Venus

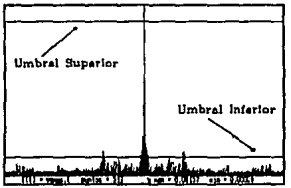


Ilustración 57. Fourier-Hartley

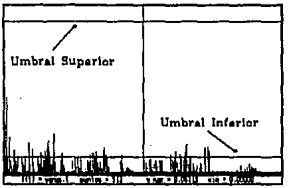


Ilustración 58. Haar

5.3.3 Clasificación por medio de red neuronal MDE

(Memoria Distribuida Esparcida)

Para la etapa de clasificación utilizamos una red neuronal basada en el modelo de P. Kanerva, *Memoria Distribuida Esparcida*[4], la cual esta encargada de clasificar las diferentes palabras, de acuerdo a la información que se le introduce. Esta red tiene la capacidad de poder

[4] *Refiérase al capítulo IV Memoria Distribuida Esparcida.*

"aprender" conforme se le va *"enseñando"* diferente información o patrones.

La información aprendida al estar distribuida en la totalidad de la red proporciona mayor potencial en su manipulación, y es precisamente esta característica la que empleamos para el reconocimiento del lenguaje natural.

Nuestro modelo de Memoria Distribuida Esparcida actúa de la siguiente forma:

La primera etapa consiste en un proceso interactivo de aprendizaje, al cual llamaremos *Aprendizaje Supervisado*, por medio del cual se le muestra a la red las características ya extraídas de cada palabra. Denominamos Aprendizaje Supervisado al hecho de guiar cuales son aquellas palabras que debe aprender y el orden en que debe hacerlo (es decir, el entrenamiento no es un proceso automático).

La forma en que la red neuronal va *aprendiendo* los diferentes patrones mostrados, posee características muy particulares que la hacen diferente de otras redes, y que en nuestro caso, consideramos se adecuan bastante a las necesidades del reconocimiento de voz.

La siguiente etapa será la de clasificación de una señal, la cual como ya se ha mencionado consistirá únicamente de un patrón binario. La red tratará de clasificar tal patrón de acuerdo a lo que se le ha enseñado. Por tanto, el arreglo de bits será la única alimentación que se le proporciona a la red, tanto en la etapa de entrenamiento como en la de clasificación.

5.3.3.1 Etapa de aprendizaje

De un total de m direcciones (cada dirección formada por un arreglo de k bits) se seleccionan todas aquellas que se encuentren alrededor de un radio de n bits de distancia (distancia de Hamming) comparado con nuestro patrón de referencia (que es el arreglo binario obtenido por la codificación de la palabra pronunciada). En las localidades de memoria asociadas a estas direcciones, se escribe la información contenida en el patrón de referencia, quedando la información *distribuida y esparcida* en toda la memoria, o mejor dicho en los *pesos* que conectan la capa de entrada de esta red neuronal con la capa intermedia. Por tanto, al ir mostrando diferentes patrones a la red, algunas características se reforzarán al irse reescribiendo sobre las mismas direcciones y muchas otras se escribirán únicamente una vez en otras direcciones.

5.3.3.2 Etapa de clasificación o reconocimiento

Ahora, para recuperar la información, nuevamente seleccionamos determinadas direcciones, semejante a la manera anterior, tomando en cuenta un nuevo patrón de referencia (el patrón que se desea clasificar o reconocer), entonces en las localidades de memoria asociadas a tales direcciones se obtiene que clase es la que tiene mayor ingerencia de acuerdo a las características de este patrón.

Si la clasificación no era satisfactoria entonces, se volvía a la etapa de entrenamiento supervisando cuales eran las palabras que se le debían *entrenar* más a la red puesto que no las había *aprendido* lo suficiente como para reconocerlas o las había perdido al tener mayor ingerencia otras palabras, cuyos patrones se almacenaban en más localidades de memoria de acuerdo a sus características y su correlación con el mapa de direcciones en, y por tanto, son -por decirlo de alguna manera- aprendidas con mayor rapidez.

El siguiente cuadro representa esquemáticamente el funcionamiento a grandes rasgos de la Red Neuronal Clasificadora.

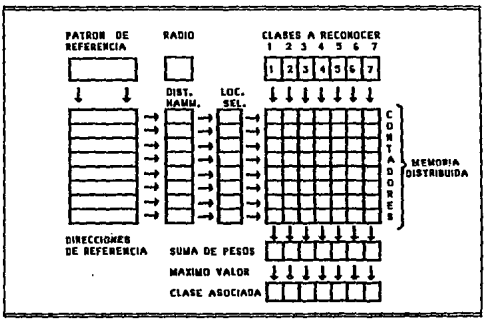


Ilustración 59.

5.4 Diseño experimental para el reconocimiento de voz

Para determinar el comportamiento del sistema, se realizó un diseño experimental que consistía en determinar los porcentajes de reconocimiento dados los siguientes grados de libertad: el número de clases o palabras, si la fase de entrenamiento era supervisada por un hombre, por una mujer o por ambos, si la clasificación, asimismo la realizaba un hombre, una mujer o los dos, y finalmente el número de entrenamientos supervisados necesarios para obtener un porcentaje de reconocimiento satisfactorio[5].

Después de realizar una serie de pruebas, pudimos identificar cuales eran los parámetros involucrados y que valores les correspondían. Así, se definió que la distancia máxima que debiese existir entre el patrón de referencia y cada una de las direcciones de la MDE debía ser de 25 bits, el arreglo binario que constituía tanto al patrón de referencia como a cada una de las direcciones debería ser de 64 bits, la longitud de las localidades de memoria dependían del número de clases o palabra que se quisieran clasificar o reconocer, y finalmente, los umbrales quedarían definidos de acuerdo a los valores anteriormente mencionados.

---

[5] Utilizamos la palabra *satisfactorio* considerando que con pocos pasos de entrenamiento se obtenían porcentajes de reconocimiento, en promedio, superiores al 80 %.

Con un número mayor de entrenamientos el porcentaje de reconocimiento se incrementa, pero es evidente que la cantidad de recursos utilizados también aumenta, y en muchas ocasiones por más que se siga entrenando a la red el porcentaje de reconocimiento no aumenta significativamente. De esta manera determinamos, a través de diversas observaciones y experimentos, que el punto ideal (esto es, cuando la relación entre el número de entrenamientos y el porcentaje de reconocimiento es máximo) es cuando se necesitan relativamente pocos pasos de entrenamiento y se obtienen porcentajes de reconocimiento superiores a un porcentaje fijo por nosotros establecido, que fue precisamente del 80 %.

TABLA DE RESULTADOS
RECONOCIMIENTO DE VOZ UTILIZANDO UNA MEMORIA DISTRIBUIDA ESPARCIDA

PARAMETROS

Radio en bits	: 25
Bits de Dirección	: 64
Bits de Memoria	: 10
Umbral Inferior	: 0.1
Umbral Superior	: 0.9

No. de Clases	Entrena	Prueba	Reconoc (%)[a]	Entrenamiento Supervisado[b]	Reconoc (%)
1	Hombre	Hombre	100	----------	100
	Hombre	Mujer	100	----------	100
	Mujer	Mujer	100	----------	100
	Mujer	Hombre	100	----------	100
	Mixto	Hombre	---	----------	100
	Mixto	Mujer	---	----------	100
2	Hombre	Hombre	100	----------	100
	Hombre	Mujer	100	----------	100
	Mujer	Mujer	95	----------	95
	Mujer	Hombre	90	----------	90
	Mixto	Hombre	---	----------	100
	Mixto	Mujer	---	----------	100
3	Hombre	Hombre	83	8 Pasos	87
	Hombre	Mujer	73	8 "	80
	Mujer	Mujer	73	7 "	90
	Mujer	Hombre	73	7 "	83
	Mixto	Hombre	---	11 "	87
	Mixto	Mujer	---	11 "	87
4	Hombre	Hombre	58	12 Pasos	88
	Hombre	Mujer	50	12 "	75
	Mujer	Mujer	80	10 "	90
	Mujer	Hombre	63	10 "	68
	Mixto	Hombre	---	19 "	80
	Mixto	Mujer	---	19 "	75
5	Hombre	Hombre	70	11 Pasos	78
	Hombre	Mujer	48	11 "	60
	Mujer	Mujer	72	11 "	76
	Mujer	Hombre	60	11 "	82
	Mixto	Hombre	---	25 "	82
	Mixto	Mujer	---	25 "	88

TABLA DETALLE PARA UNA CLASE[c]

Clase	Entrena	Prueba	Reconoc (%)[a]	Entrenamiento Supervisado[b]	Reconoc (%)
1	Hombre	Hombre	100	----------	100
	Hombre	Mujer	100	----------	100
	Mujer	Mujer	100	----------	100
	Mujer	Hombre	100	----------	100
	Mixto	Hombre	---	----------	100
	Mixto	Mujer	---	----------	100

TABLA DETALLE PARA DOS CLASES[c]

Clase	Entrena	Prueba	Reconoc (%)[a]	Entrenamiento Supervisado[b]	Reconoc (%)
1	Hombre	Hombre	100	----------	100
	Hombre	Mujer	100	----------	100
	Mujer	Mujer	100	----------	100
	Mujer	Hombre	90	----------	90
	Mixto	Hombre	---	----------	100
	Mixto	Mujer	---	----------	100
2	Hombre	Hombre	100	----------	100
	Hombre	Mujer	100	----------	100
	Mujer	Mujer	90	----------	90
	Mujer	Hombre	90	----------	90
	Mixto	Hombre	---	----------	100
	Mixto	Mujer	---	----------	100

TABLA DETALLE PARA TRES CLASES[c]

Clase	Entrena	Prueba	Reconoc (%)[a]	Entrenamiento Supervisado[b]	Reconoc (%)
1	Hombre	Hombre	80	2 Pasos	80
	Hombre	Mujer	100		80
	Mujer	Mujer	50	2 Pasos	100
	Mujer	Hombre	50		90
	Mixto	Hombre	---	3 Pasos	90
	Mixto	Mujer	---		80
2	Hombre	Hombre	80	2 Pasos	90
	Hombre	Mujer	90		90
	Mujer	Mujer	80	2 Pasos	80
	Mujer	Hombre	80		80
	Mixto	Hombre	---	4 Pasos	80
	Mixto	Mujer	---		100
3	Hombre	Hombre	90	4 Pasos	90
	Hombre	Mujer	30		70
	Mujer	Mujer	90	3 Pasos	90
	Mujer	Hombre	90		80
	Mixto	Hombre	---	4 Pasos	90
	Mixto	Mujer	---		80

TABLA DETALLE PARA CUATRO CLASES[c]

Clase	Entrena	Prueba	Reconoc (%)[a]	Entrenamiento Supervisado[b]	Reconoc (%)
1	Hombre	Hombre	100	2 Pasos	100
	Hombre	Mujer	90		100
	Mujer	Mujer	100	2 Pasos	100
	Mujer	Hombre	50		50
	Mixto	Hombre	---	3 Pasos	100
	Mixto	Mujer	---		100
2	Hombre	Hombre	80	3 Pasos	100
	Hombre	Mujer	90		80
	Mujer	Mujer	70	3 Pasos	100
	Mujer	Hombre	60		90
	Mixto	Hombre	---	3 Pasos	70
	Mixto	Mujer	---		70
3	Hombre	Hombre	30	3 Pasos	80
	Hombre	Mujer	10		70
	Mujer	Mujer	70	2 Pasos	80
	Mujer	Hombre	60		60
	Mixto	Hombre	---	7 Pasos	60
	Mixto	Mujer	---		70
4	Hombre	Hombre	20	4 Pasos	70
	Hombre	Mujer	10		50
	Mujer	Mujer	80	3 Pasos	80
	Mujer	Hombre	80		70
	Mixto	Hombre	---	6 Pasos	90
	Mixto	Mujer	---		60

TABLA DETALLE PARA CINCO CLASES[c]

Clase	Entrena	Prueba	Reconoc (%)[a]	Entrenamiento Supervisado[b]	Reconoc (%)
1	Hombre	Hombre	100	2 Pasos	100
	Hombre	Mujer	50		100
	Mujer	Mujer	90	2 Pasos	90
	Mujer	Hombre	80		70
	Mixto	Hombre	---	3 Pasos	90
	Mixto	Mujer	---		100
2	Hombre	Hombre	90	2 Pasos	90
	Hombre	Mujer	70		70
	Mujer	Mujer	90	2 Pasos	80
	Mujer	Hombre	90		90
	Mixto	Hombre	---	5 Pasos	80
	Mixto	Mujer	---		100
3	Hombre	Hombre	50	2 Pasos	70
	Hombre	Mujer	30		10
	Mujer	Mujer	80	3 Pasos	50
	Mujer	Hombre	40		70
	Mixto	Hombre	---	6 Pasos	80
	Mixto	Mujer	---		80
4	Hombre	Hombre	40	3 Pasos	60
	Hombre	Mujer	30		70
	Mujer	Mujer	40	2 Pasos	80
	Mujer	Hombre	40		90
	Mixto	Hombre	---	7 Pasos	80
	Mixto	Mujer	---		90
5	Hombre	Hombre	70	2 Pasos	70
	Hombre	Mujer	60		50
	Mujer	Mujer	60	2 Pasos	80
	Mujer	Hombre	50		90
	Mixto	Hombre	---	4 Pasos	80
	Mixto	Mujer	---		70

OBSERVACIONES:

a. Porcentaje de reconocimiento sin entrenar la red neuronal, esto es, únicamente presentando cada patrón de voz una sola vez.

b. El número de pasos de entrenamiento mostrado, es el total, es decir,tomando en cuenta la presentación inicial de los patrones de voz a la red neuronal.

c. Las tablas detalle, muestran cuantas veces fué entrenado cada patrón de una clase, ya que el número de entrenamientos necesario no fué igual para cada uno de éstos.

5.5 Análisis de resultados

De acuerdo con los datos presentados en las tablas anteriores, podemos estimar, extrapolando tales datos, el comportamiento general del sistema, utilizando para ello herramientas matemáticas y estadísticas.

El método utilizado fue el *ajuste de curvas*, esto es, ajustamos una función que se acople lo mejor posible a los datos experimentales obtenidos. De tal forma, se usó un ajuste de curvas obtenido por Regresión Polinomial y por Mínimos Cuadrados (que en realidad es una Regresión Lineal, caso particular de la Regresión Polinomial).[6]

Se probaron cinco diferentes tipos de funciones, las tres primeras obtenidas a través del método de Regresión Lineal (Mínimos Cuadrados), mientras que las otras dos se obtuvieron aplicando Regresión Polinomial. Los tipos de funciones son:

A. Lineal	$y = ax + b$
B. Exponencial	$y = ae^{bx}$
C. Potencia	$y = ax^b$
D. Pol. 2do. grado	$y = ax^2 + bx + c$
E. Pol. 3er. grado	$y = ax^3 + bx^2 + cx + d$

[6] Ver apéndice III para más información.

La siguientes tablas, muestran los diferentes tipos de funciones que se trataron de ajustar a los puntos obtenidos experimentalmente para cuando el parlante fuese un hombre, una mujer o ambos, así como su "grado de ajuste" definido por el coeficiente de correlación.

Curvas de Ajuste para datos de Hombre

Tipo de Función	F u n c i ó n y = f(x)	Coeficiente de Correlación
A	$6.1190X - 10.7857$	0.93
B	$0.9153e^{0.5193X}$	0.957
C	$0.8222X^{1.6439}$	0.96
D	$1.0595X^2 - 3.4166X + 5.1071$	0.984
E	$0.2474X^3 - 2.2813X^2 + 9.3282X - 7.1428$	0.995

Curvas de Ajuste para datos de Mujer

Tipo de Función	F u n c i ó n y = f(x)	Coeficiente de Correlación
A	$5.6310X - 9.7143$	0.939
B	$0.8919e^{0.5132X}$	0.938
C	$0.8067X^{1.8179}$	0.965
D	$0.8630X^2 - 2.1369X + 3.2321$	0.982
E	$0.1489X^3 - 1.1482X^2 + 5.5360X - 4.1428$	0.987

Curvas de Ajuste para datos Mixtos

Tipo de Función	Función y = f(x)	Coeficiente de Correlación
A	$7.0952X - 9.1786$	0.976
B	$2.001e^{0.2465X}$	0.896
C	$1.7616X^{1.0122}$	0.988
D	$0.5952X^2 + 1.7380X - 0.25$	0.989
E	$0.1565X^3 - 1.5183X^2 + 9.8012X - 8$	0.993

Como se puede observar en la tablas anteriores, el sistema se comporta de manera regular, en los tres casos (para datos de hombre, de mujer y mixtos), es decir, las curvas siempre tienen un "grado de ajuste" de acuerdo al siguiente orden:

1. Polinomio de 3er. grado ..... Mejor
2. Polinomio de 2do. grado ..... |
3. Función de potencia ..... |
4. Función Lineal ..... ↓
5. Función Exponencial ..... Peor[7]

[7] Solo en los datos para hombre, se invierte el orden de la función exponencial y de la lineal, ya que esta última se ajusta mejor para estos datos.

Sin embargo, a pesar de que el polinomio de tercer grado, tiene el mejor ajuste (y mientras más elevado sea el grado del polinomio el ajuste se irá perfeccionando)[8], no proporciona gran ayuda para tratar de definir el comportamiento del sistema, ya que cabe recordar que los métodos de ajuste de curvas son sumamente eficientes para la *interpolación*, es decir, para obtener valores de puntos que aunque no están definidos en los datos observados, sí caen dentro del rango de éstos. No obstante, el error en que se incurre al tratar de hacer una extrapolación (obtención de información a partir de datos que están fuera del rango comprendido por los datos observados) es muy grande y se incrementa más aún cuando se utilizan funciones que varían rápidamente; tal es el caso de la función exponencial y el polinomio de grado tres o mayor.

Por otra parte, experimentalmente hemos observado que la tendencia del comportamiento del sistema es mucho más hacia un polinomio de segundo grado que a uno de tercero o mayor.

Finalmente, también hemos observado que el comportamiento aparente del sistema es regular, es decir se comporta de manera semejante para circunstancias semejantes, ya que como se puede visualizar en las tablas, las ecuaciones para cada uno de los tipos de funciones son muy parecidas (coeficientes con valores muy parecidos) y asimismo, los coeficientes de correlación asociados

---

[8] El polinomio para el cual se tuvo un ajuste perfecto (con un coeficiente de correlación igual a uno), se obtuvo utilizando una *interpolación de Newton*, el cual para los tres casos resultó ser un polinomio de séptimo grado.

Para el primer caso, con los datos de hombre, se muestra el polinomio obtenido:

$0.0246x^7-0.7638x^6+9.6138x^5-62.93x^4+228.7472x^3-455.8055x^2+458.1142x-176$

No obstante que el ajuste tiene un coeficiente de correlación perfecto, el grado de error en el que se incurre al interpolar es bastante grande, pero lo es mucho más al tratar de extrapolar información, ya que la variación de la función es sumamente rápida.

a cada una de estas funciones son también muy semejantes independientemente de que los datos provengan de un hombre, de una mujer o de ambos.

Por todas estas razones, concluimos que un polinomio de grado dos, podría modelar **aproximadamente** el comportamiento general del sistema.

Las subsecuentes gráficas muestran el ajuste hecho con los polinomios de la forma $y = ax^2 + bx + c$, que son precisamente los correspondientes a las tablas antes descritas.

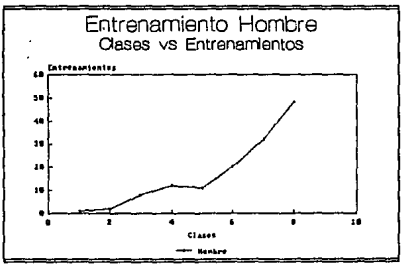


Ilustración 60.

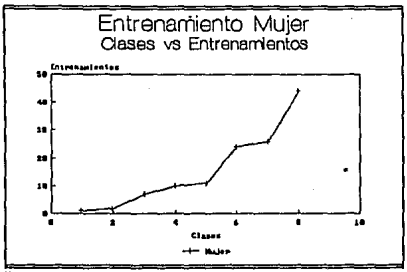


Ilustración 61.

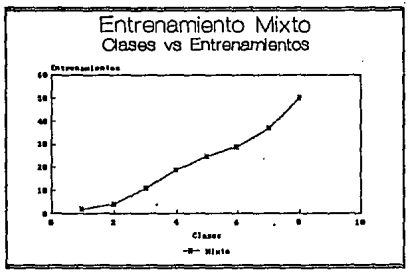


Ilustración 62.

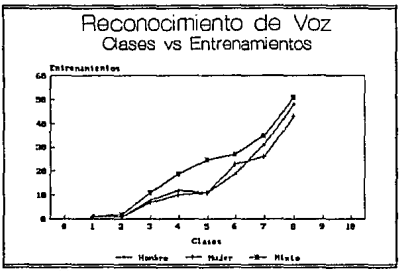


Ilustración 63.

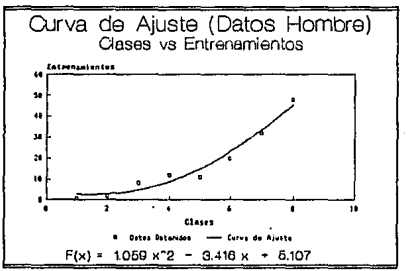


Ilustración 64.

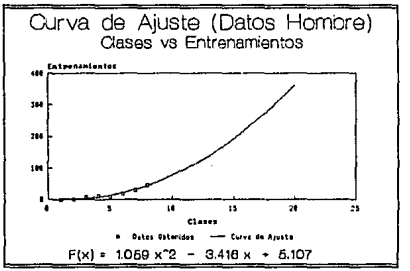


Ilustración 65.

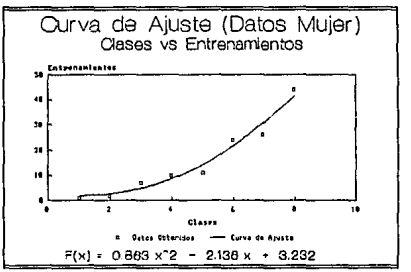


Ilustración 66.

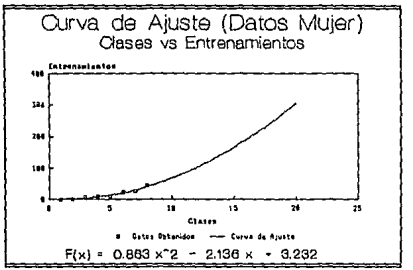


Ilustración 67.

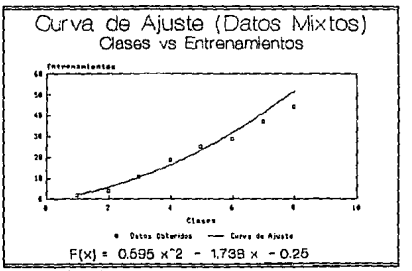


Ilustración 68.

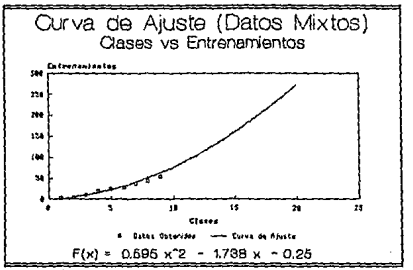


Ilustración 69.

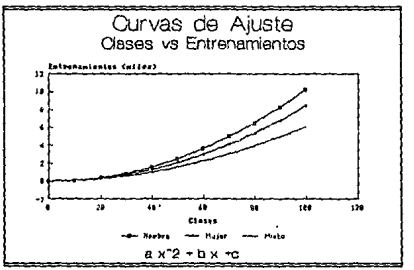


Ilustración 70.

5.6 Conclusiones

De acuerdo con el análisis de resultados hecho en el punto anterior, queda claro que para una red del tipo Memoria Distribuida Esparcida, cada patrón a reconocer necesita ser entrenado, en general, tantas veces como el total de patrones que se deseen clasificar. Rápidamente puede observarse que ésto puede resultar molesto si se desea entrenar a la red con un vocabulario mediano, ya que el número de entrenamientos sería extensamente largo y muy probablemente el porcentaje de reconocimiento se vería disminuido.

DISCUSION Y CONCLUSIONES

DISCUSION Y CONCLUSIONES

Con el presente trabajo hemos tratado de demostrar como es posible realizar un reconocedor de voz utilizando relativamente pocos recursos, fácil de implementar y sin necesitar equipo costoso o especializado. El porcentaje de reconocimiento compite con los métodos y sistemas tradicionales, pero además supera a éstos en su capacidad al no importar la persona que entrene al sistema y la persona que hable en la prueba misma de reconocimiento. La idea es tomar las "características esenciales" de la señal de voz, de tal manera que, en primer término, la cantidad de información a procesar no sea demasiado grande y en segundo, que el proceso de los datos no involucre demasiada manipulación de la misma ni cálculos exhaustivos que se reflejarían directamente en el tiempo de respuesta del sistema.

Por otra parte, hemos demostrado cómo se pueden realizar tareas complejas (propias del mundo "real"), utilizando técnicas modernas que tratan de simular o imitar el funcionamiento del cerebro humano. La información en este tipo de modelos no se encuentra localizada en un solo lugar, sino que se encuentra distribuida en la totalidad de la red (tal como lo hace el cerebro). Aunado a todo esto, la capacidad de estas "redes neuronales" de procesar la información de manera paralela (y no secuencialmente como se haría en una máquina tipo Von Newmann), hacen de éstas una herramienta muy poderosa y potencialmente aplicables a una gran cantidad de problemas.

El modelo de Memoria Distribuida Esparcida (MDE) que fué utilizado, tiene un fundamento matemático formal bien definido y tal vez bastante complejo. Se basa en características propias de un espacio binario (esto es, donde el dominio es tan solo de 0's o 1's) n-dimensional y sobre todo en propiedades de la estadística matemática.

Además, una ventaja adicional del algoritmo mostrado para el procesamiento de la información en la MDE es el hecho de que se puede tener acceso a la representación del conocimiento (mejor que información), cuestión que en muchos otros sistemas es confuso, oculto o muy complejo cuando se trata de realizar tareas complicadas y que además es, uno de los principales puntos de interés para la otra rama de la computación que ha intentado asimismo, entender y simular el comportamiento del cerebro, la llamada **Inteligencia Artificial.**

Cabe citar que otra idea que hemos estado manejando en el diseño del prototipo es el de hacer una transformación de un dominio temporal a uno espacial, es decir, el procesamiento de la señal de voz que en un principio está estrictamente relacionada con el tiempo, al pasar la información a la red neuronal ya no depende del tiempo sino más bien de una organización espacial definida característica (como una imagen), con lo cual evitamos el fuerte problema del defasamiento de la señal de voz que es común que se produzca cuando una palabra es pronunciada por personas diferentes (e incluso por una misma persona, ya que nunca pronunciamos una misma palabra exactamente igual). Y aún cuando para esta transformación utilizamos técnicas convencionales propias para análisis de señales en tiempo, la "verdadera manipulación" comienza en la red neuronal -MDE- donde la información, que ahora podemos

llamar conocimiento, son las características propias de esa determinada señal en cuestión.

Por otra parte, en general, cuando se habla de redes neuronales, se utiliza el término "entrenamiento-aprendizaje", y es que realmente un modelo neuronal formal aparenta esa tarea propia del cerebro, ya que cuando se le agrega nuevo conocimiento a la red, no pierde la información anteriormente aprendida, por lo tanto el término "memoria" también es válida dentro de este contexto, y la Memoria Distribuida Esparcida, ha demostrado firmemente esta idea.

Las aplicaciones para este tipo de modelos en general, y en particular para nuestro sistema prototipo de reconocimiento de voz parecen ser infinitas. Se podría aplicar como controlador de procesos donde la intervención humana directa es peligros o aún más, imposible; como controlador de mecanismos biomédicos para personas parapléjicas; como un medio didáctico en general o para personas autistas o con dislexia; para automatizar procesos cotidianos; para resolver el problema de comunicación que dio origen a este tema, ... la telefonía; para realizar interfases con un robot para tareas específicas; o simplemente para tener una interfase más natural y eficiente con la computadora, etc., etc.

Si bien es cierto que el sistema propuesto tiene un desempeño adecuado, también lo es que podemos hacer algunas modificaciones para mejorar su funcionamiento. En primer lugar, sería conveniente el realizar la MDE no por programa (software) sino en hardware, con lo cual el tiempo de respuesta sería mucho menor y por tanto sería mas eficaz. Además el modelo se presta bastante para poder realizarlo con componentes electrónicos y tecnología VLSI. Segundo,

se pueden utilizar las señales mismas de la computadora (a través de los slots) para la sincronización y alimentación del circuito, en lugar de utilizar fuentes de alimentación y temporizadores externos (como lo hacemos nosotros, donde necesitamos 5 volts para los elementos TTL y -15 y +15 volts para la etapa de potencia, además de un temporizador LM555). También se pueden utilizar tarjetas especiales que ya contienen integrado un convertidor analógico digital de mucho mejor calidad y resolución (aunque obviamente ello aumentaría el costo) para tener así mayor control de la señal a procesar sin tener aunado al "ruido" propio de la señal de voz que se produce al pronunciar en diferentes ocasiones una misma palabra, la distorsión o "truncamiento" de la información debido a la calidad de los dispositivos utilizados. En tercer término, y tal vez el más importante, sería conveniente implementar un algoritmo de aprendizaje para la red donde el entrenamiento fuera iterativo y automático, sin tener que intervenir en cada paso de entrenamiento para indicar que la señal en turno pertenece a determinada clase, como se hace en un **"entrenamiento supervisado"**. En otras palabras, se tendría que idear un método como el LMS o el back-propagation para que de alguna manera se tuviera una función de "gradiente" que determinara como afectar a los pesos de la red automáticamente de acuerdo a las características de la señal. Ello es un buen reto, ya que si la red ha mostrado que se necesitan pocos pasos de entrenamiento para un buen desempeño, haciendo el proceso de entrenamiento automático (también llamado no supervisado), la red tendría clara ventaja sobre los otros modelos neuronales existentes. Por último, se ha mencionado que el éxito o fracaso de una MDE depende en mucho de la correlación que exista entre los datos de entrada a aprender o clasificar (dirección de referencia) y las N direcciones de la memoria; de esta manera, se pueden crear algoritmos "híbridos" que tomen ideas de otros modelos para que

la correlación entre ambos elementos sea óptima. Uno de estos algoritmos "híbridos" es el propuesto por David Rogers, en su Memoria Genética, que toma las características de la MDE y del Algoritmo Genético de Holland. En éste, se eliminan aquellas localidades de memoria que tienen poca correlación con la información de entrada y se crean nuevas direcciones que posean características de otras direcciones que sí tienen una alta correlación (por eso se alude a la genética, donde las mejores características para la sobrevivencia se heredan a las generaciones posteriores).

El camino por recorrer es todavía largo, pero consideramos que se han dado en los últimos años adelantos substanciales en la forma de analizar y atacar los problemas, han surgido nuevas ideas, nuevos métodos, nuevos... retos. Los resultados son aún insospechados. Y aún cuando sabemos que estamos lejos de comprender en su totalidad como funciona una verdadera "red neuronal humana", con este trabajo, nosotros hemos querido ser parte de ese cambio.

aeh-aitl
Junio, 94

APENDICE I

MATERIAL Y DIAGRAMA DEL CIRCUITO ELECTRONICO PARA EL RECONOCIMIENTO DE VOZ

APENDICE I

MATERIAL Y DIAGRAMA DEL CIRCUITO ELECTRONICO PARA EL RECONOCIMIENTO DE VOZ

Material Utilizado:

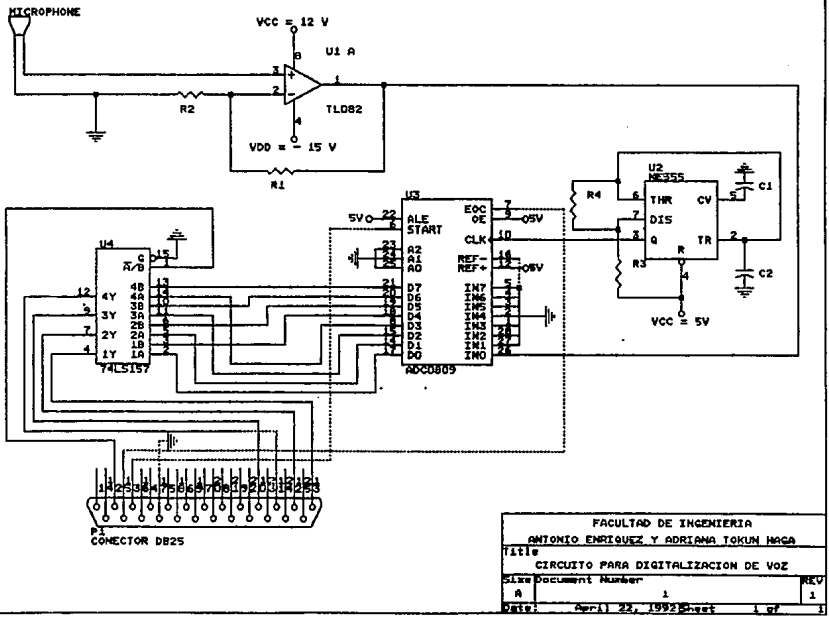
- Micrófono de baja impedancia
- Convertidor Analógico-Digital ADC0809 de 8 bits con multiplexor de 8 canales
- Amplificador Operacional TL084 entrada jfet
- Multiplexor LS157N cuadruple de 2 entradas
- Temporizador 555
- Bus de ocho canales
- Conector DB-25 de 28 pines
- Resistencias

  $R_1 = 10\ K\Omega$ $R_2 = 100\ \Omega$

  $R_3 = 1\ K\Omega$ $R_4 = 22\ \Omega$

- Capacitores

  $C_1 = 0.01\ \mu F$ $C_2 = 47\ nF$


MICROPHONE
VCC = 12 V
U1 A
TL082
R2
VDD = -15 V
R1
U2
NE555
THR
CV
DIS
Q
TR
R
R4
R3
C1
C2
VCC = 5V
U3
ALE
START
EOC
OE
CLK
REF-
REF+
5V
ADC0809
U4
74LS157
P1
CONECTOR DB25
FACULTAD DE INGENIERIA
ANTONIO ENRIQUEZ Y ADRIANA TOKUN HAGA
Title
CIRCUITO PARA DIGITALIZACION DE VOZ
Size
A
Document Number
1
REV
1
Date: April 22, 1992
Sheet 1 of 1

APENDICE II

RELACION DE LOS VOLTAJES DE ENTRADA Y DE SALIDA DEL CIRCUITO ELECTRONICO PARA EL RECONOCIMIENTO DE VOZ

APENDICE II

RELACION DE LOS VOLTAJES DE ENTRADA Y DE SALIDA DEL CIRCUITO ELECTRONICO PARA EL RECONOCIMIENTO DE VOZ

En el sistema electrónico diseñado, el voltaje de la salida digital será proporcional al de la entrada analógica (o señal de voz), la cual tendrá la siguiente relación:

$$V_{OUT} = \frac{V_{IN}}{V_{REF}} = \frac{V_{IN}}{V_{CC}}$$

$$V_{REF} = V_{CC} = 5\ V$$

$$\therefore\ 0 \le V_{IN} \le 5\ V$$

pero como tenemos solo 8 bits => la máxima resolución será de:

$$\sum_{i=1}^{n} a_i 2^{-i}$$

si n = 8

$$\sum_{i=1}^{n} a_8 2^{-i}$$

para la máxima resolución:

$$a_i = 1\ \forall\ i$$

por lo tanto la máxima resolución será de $0.99609 \approx 0.9961$

lo cual es igual a tener todos los bits en uno (1111 1111).

1	1	1	1	1	1	1	1	=>	todos los bits
↓	↓	↓					↓		prendidos
2^{-1}	2^{-2}	2^{-3}		.	.	.	2^{-8}		

pero al leer del puerto paralelo 1111 1111 = 255 ¿Cómo saber que $255_{(binario)} \approx 0{,}9961_{(decimal)}$?

ejemplos:

V^{IN}	V^{OUT}	$V^{binario}$	$V^{decimal}$
0.997	1111 1111	255	0.99609
0.57	1001 0001	145	0.56641
0.566	1001 0000	144	0.56250
0.56	1000 1111	143	0.55859
0.038	0110 0001	97	0.37891
0.037	0101 1110	94	0.36719
0.009	0000 0010	2	0.00781

Una manera consiste en regresar el valor leído a su valor binario (por ejemplo: 255 => 1111 1111) y aplicar un algoritmo que pase éste número binario a su correspondiente en valor decimal (por ejemplo: 1111 1111 => 0.99609) esto es:

$$\sum_{i=1}^{n} a_i 2^{-i}$$

Pero existe otra manera, y es el dividir el número leído entre un valor $256 = 2^8$, por ejemplo:

1111 1111 = 255 => 255/256 = 0.99609

1000 1111 = 143 => 143/256 = 0.55859

lo cual proporciona el mismo valor que al aplicar la sumatoria.

Demostración:

Sea un arreglo de bits de tamaño n, donde el bit más significativo (MBS) se encuentra a la izquierda:

$$\sum_{i=1}^{n} a_i 2^{-i} = \sum_{i=1}^{n} a_i 2^{n-i}$$

$$\sum_{i=1}^{n} a_i 2^{-i} = \frac{\sum_{i=1}^{n} a_i 2^{n-i}}{2^n}$$

$$2^n \sum_{i=1}^{n} a_i 2^{-i} = \sum_{i=1}^{n} a_i 2^{n-i}$$

$$\sum_{i=1}^{n} a_i 2^n 2^{-i} = \sum_{i=1}^{n} a_i 2^{n-i}$$

$$\sum_{i=1}^{n} a_i 2^{n-i} = \sum_{i=1}^{n} a_i 2^{n-i}$$

l.q.q.d.

APENDICE III

AJUSTE DE CURVAS

APENDICE III

AJUSTE DE CURVAS

En general, los resultados obtenidos experimentalmente de un fenómeno determinado proporcionan un conjunto de datos discretos. No obstante, en ocasiones se requieren estimaciones de puntos entre esos valores discretos, ó aún mas importante, cuando se desea modelar mediante una ecuación el comportamiento del fenómeno para cualquier intervalo de su dominio o simplemente cuando se quiere tener una versión simplificada de una función muy complicada. Una manera de hacerlo, es calcular valores de la función en un conjunto de valores discretos a lo largo del rango de interés. Después se puede obtener un función más simple ajustando estos valores. A estos métodos se les conoce en conjunto como **Ajuste de Curvas.**

Existen dos esquemas generales en el ajuste de curvas, que se distinguen entre sí con base a la cantidad de error asociada con los datos. Primero, donde los datos muestran un grado significativo de error o "ruido", la estrategia es derivar una curva simple que represente el comportamiento general de los datos. Ya que cada punto individual puede estar incorrecto, no es necesario intersectar cada uno de ellos. En lugar de esto, la curva se diseña de tal manera que siga un patrón sobre los puntos tomados como un todo. A un procedimiento de esta naturaleza se le conoce como regresión con **mínimos cuadrados,** caso particular de la regresión polinomial.

Segundo, donde se conoce que los datos son muy exactos, el proceso es ajustar una curva o una serie de curvas que pasen exactamente por cada uno de los puntos. Estos datos generalmente se derivan de tablas. Dentro de esta categoría caben los métodos de **Interpolación de Newton** y la **Interpolación Polinomial de Lagrange.** No obstante, el grave problema de estos métodos es que el grado del polinomio que modela los datos está relacionado de manera directa con la cantidad de datos que se tienen; es decir, si tenemos un tabla de 5 datos, el grado del polinomio será 4, para 6 datos se tendrá un polinomio de 5o. grado y así sucesivamente. Además, el grado en que se incurre al realizar una **interpolación** (estimación de valores entre puntos discretos conocidos) puede ser muy grande, pero cuando se desea un **extrapolación** (estimación de valores fuera del rango de puntos discretos conocidos) el error en que se incurre es "enorme".

En las siguientes páginas se darán los fundamentos del ajuste de datos por medio de una Regresión Lineal (Mínimos Cuadrados) y en caso más general, por una Regresión Polinomial, aclarando cómo se utilizaron estos métodos para intentar **"modelar matemáticamente"** (aunque de manera aproximada y tal vez con un gran índice de error) el funcionamiento general del Sistema de Reconocimiento de Voz descrito en el capítulo V.

III.1 Regresión Lineal

El ejemplo más simple de una aproximación por mínimos cuadrados es el ajuste de una línea recta a un conjunto de parejas de datos observadas: $(x_1,y_1),(x_2,y_2),...,(x_n,y_n)$. De esta

manera, si estableciéramos como función de aproximación una recta, su ecuación sería:

$$f(x) = y = a0 + a1x + E$$

donde E es el error entre el modelo y las observaciones, por tanto se puede representar como ...

$$E = y - a0 + a1x$$

Ahora, si definimos a Sr, como la suma del error cuadrático de la forma siguiente ...

$$Sr = \sum E_i^2 = \sum (y_i - a0 - a1x_i)^2, \text{para } i = 1..n$$

entonces, haciendo que Sr, sea mínimo, esto es, que varíe lo mínimo con respecto a a0 y a a1, entonces tenemos las ecuaciones siguientes:

$$\frac{\delta Sr}{\delta a0} = -2\sum (y_i - a0 - a1x_i) = 0$$

por tanto

$$\sum y_i - \sum a0 - \sum a1x_i = 0 \quad \text{... (1)}$$

$$\frac{\delta Sr}{\delta a1} = -2\sum (y_i - a0 - a1x_i)(-x_i) = 0.$$

entonces

$$\sum y_i x_i - \sum a0x_i - \sum a1x_i^2 = 0 \quad \text{...(2)}$$

A las ecuaciones (1) y (2) se les llama ecuaciones normales y se les acostumbra escribir:

$$\Sigma y_i = na0 + \Sigma a1 x_i$$

$$\Sigma y_i x_i = \Sigma a0 x_i + \Sigma a1 x_i^2$$

De donde:

$$a1 = \frac{n \Sigma y_i x_i - \Sigma x_i \Sigma y_i}{n \Sigma x_i^2 - (\Sigma x_i)^2}$$

$$a0 = \bar{y} - a1 \bar{x} \text{ ; donde: } \bar{y} = \Sigma y_i$$

$$\bar{x} = \Sigma x_i / n$$

Para determinar el grado de error, podemos utilizar las siguientes expresiones:

Si

$$r^2 = (S_t - S_r)/S_r$$

entonces

$$r = (S_t - S_r)/S_r)^{1/4}$$

donde :

$$S_t = \Sigma(y_i - \bar{y})^2$$

$$S_r = \Sigma(y_i - a0 - a1 x_i)^2$$

S_t representa la desviación de cada valor con respecto a la media (error cuadrático medio), y S_r, como ya hemos mencionado, es la diferencia cuadrática que existe entre el valor real y el valor obtenido por el modelo, en otras palabras, $S_r=(y_i-f_i)^2$. El coeficiente de correlación está denotado por r, y r^2 representa el coeficiente de determinación. Si $r^2 = 1$ y $S_r = 0$ entonces tenemos un ajuste perfecto.

La regresión lineal proporciona una técnica muy poderosa para ajustar datos a una mejor línea. Sin embargo, se ha predispuesto que la relación entre las variables dependiente e independiente es lineal. Este no es siempre el caso, y en cualquier análisis de regresión el primer paso debe ser el trazar y visualizar los datos para decidir si es correcto o aceptable el aplicar el modelo lineal. En algunos casos, técnicas como la regresión polinomial serán apropiadas. En otros, se pueden hacer transformaciones que expresen los datos de manera que sean compatible con la regresión lineal. Ejemplos de ello son los modelos exponencial y potencia .

Para ajustar los datos a una función no lineal del tipo $y=_ax^b$ (tipo potencia de x), podemos "linealizar" tal función de la siguiente manera:

$$y=_ax^b$$

$$\log(y) = \log(_ax^b)$$

$$\log(y) = \log(a) + b*\log(x)$$

$$\downarrow \quad \downarrow \quad \downarrow$$

$$Y = A0 + A1\,X$$

por lo tanto ...

$Y = \log(y),\ X = \log(x)$

y como $A0 = \log(a)$ entonces $a = \text{antilog}(A0)$

$A1 = b$

Para ajustar a una función exponencial del tipo $y = {}_a e^{bx}$ podemos utilizar de manera similar el método anterior :

$$y = {}_a e^{bx}$$

$$\ln(y) = \ln({}_a e^{bx})$$

$$\ln(y) = \ln(a) + {}_b x * \ln(e)$$

$$\ln(y) = \ln(a) + bx$$

$$\downarrow \quad \downarrow \quad \downarrow$$

$$Y = A0 + A1 * X$$

por lo tanto ...

$Y = \ln(y),\ X = x$

$A0 = \ln(a)$, entonces $a = \exp(A0)$

$A1 = b$

III.2 Regresión Polinomial

El procedimiento de mínimos cuadrados se puede extender fácilmente y ajustar datos a un polinomio de m-ésimo grado :

$$y = a_0 + a_1x + a_2x^2 + ... + a_mx^m$$

En este caso, la suma de los cuadrados de los errores es:

$$Sr = \sum (y_i - a0 - a1x_i - a2x^2_i - ... - amx^m_i)^2, \text{ para } i = 1 .. n$$

Siguiendo los mismos pasos de la regresión lineal, se toma la derivada "parcial" con respecto a cada uno de los coeficiente del polinomio para obtener:

$$\frac{\delta Sr}{\delta a0} = -2\sum (y_i - a0 - a1x_i - a2x^2_i - ... - amx^m_i)$$

$$\frac{\delta Sr}{\delta a1} = -2\sum x_i(y_i - a0 - a1x_i - a2x^2_i - ... - amx^m_i)$$

$$\frac{\delta Sr}{\delta a2} = -2\sum x^2_i(y_i - a0 - a1x_i - a2x^2_i - ... - amx^m_i)$$

.

.

.

$$\frac{\delta Sr}{\delta am} = -2\sum x^m_i(y_i - a0 - a1x_i - a2x^2_i - ... - amx^m_i)$$

Estas ecuaciones se pueden igualar a cero y reordenar de tal forma que se obtenga el siguiente conjunto de ecuaciones normales:

$$a0n + a1\sum x_i + a2\sum x^2_i + \ldots + am\sum x^m_i = \sum y_i$$

$$a0\sum x_i + a1\sum x^2_i + a2\sum x^3_i + \ldots + am\sum x^{m+1}_i = \sum x_i y_i$$

$$a0\sum x^2_i + a1\sum x^3_i + a2\sum x^4_i + \ldots + am\sum x^{m+2}_i = \sum x^2 y_i$$

$$\vdots$$

$$a0\sum x^m_i + a1\sum x^{m+1}_i + a2\sum x^{m+2}_i + \ldots + am\sum x^{2m}_i = \sum x^m_i y_i$$

Resolviendo este sistema de ecuaciones algebraicas podemos encontrar los "m" coeficientes del polinomio. El coeficiente de correlación se calcula igual que en la forma anterior y con él podemos determinar que polinomio se ajusta mejor a nuestros datos experimentales.

INDICE DE ILUSTRACIONES

INDICE DE ILUSTRACIONES

BIBLIOGRAFIA

BIBLIOGRAFIA

1. Ainsworth, William A.
Speech Recognition by Machine
Peregrinus on Behalf of the Institution of Electrical Engineers
Londres, 1988

2. Anderson Charles
A Conditional Probability Interpretation
of Kanerva's Sparse Distribuited Memory
m/s 23-100 Jet Propulsion Laboratory
Pasadena, CA 91109

3. Arbib Michael
Brains, Machines and Mathematics
Springer-Verlag
Nueva York, 1987

4. Atlas, Les
Nonlinear Classification by Trained Multilayer
Perceptrons and Trained Calssifications Trees
IEEE Proceedings, Vol. 78, Octubre 1990

5. Baher H.
Analog and Digital Processing
J. Wiley
Chichester, 1990

6. Boylestad Robert, Nashelsky Louis
Electrónica, Teoría de Circuitos
Prentice Hall
México, 1982

7. Bracewell, R.
The Hartley Transform
Oxford University Press
Nueva York, 1986

8. Brigham Oran
Fast Fourier Transform
Prentice Hall, 1974

9. Brigham Oran
Fast Fourier Transform and Its Applications
Prentice Hall, 1988

10. Burr David
Experiments on Neural Net Recognition
of Spoken and Written Text
IEEE Transactions on Acoustics,
Speech and Signal Processing
Vol. 36, Julio 1988

11. Burrus C.
D.F.T./F.F.T. and Convolutions
J. Wiley
Nueva York, 1985

12. Burton D. K., Shore J. E.
Isolated Word Speak Recognition Using Multisection
Vector Quantization CodeBooks
IEEE Transactions on Acoustics,
Speech and Signal Processing
Vol. 33, Abril 1985

13. Chapra Steven, Canale Raymon
Métodos Numéricos para Ingenieros
Mc Graw Hill
México, 1987

14. Clifford J.
Advanced Speech Processing in Military
Computer-Based Systems
IEEE Proceedings, Vol. 79, Noviembre 1991

15. Clifford Lau
Artificial Neural Networks:
Paradigms, Applications and Hardware Implementation
IEEE
New York, 1992

16. Dixon Rex
Automatic Speech and Speaker Recognition
IEEE
Nueva York, 1979

17. Fino, B.J.
Relations Between Haar and Walsh-Hadamard Transforms
IEEEE Proceedings, Vol. 60, No. 5

18. Flanagan James
Speech Analisis Synthesis and Perception
Head Acoustics Research Departament, Bell Laboratories
Murray Hill, New Jersey

19. Freeman James, Skapura David
Neural Networks Algorithms,
Aplications and Programming Techniques
Addison-Wesley, 1991

20. Gómez Eduardo
Implementación, Comparación y Uso de las Transformadas:
Fourier, Hartley, Haar, Walsh y Método de Máxima Entropía
para el Análisis de Espectros
Universidad La Salle
México, 1992

21. Haton J. P.
Fundamentals in Computer Understanding
Speech and Vision
Cambridge University
Cambridge 1987

22. Hetch Robet, Nielsen A.
Neurocomputing
Addison Wesley
Massachusetts, 1990

23. Kanerva Pentti
Sparse Distribuited Memory
MIT Press
Cambridge, 1978

24. Karger
Speech and Speaker Recognition
Shoeder-Basel, 1985

25. Keener James
Principles of Applied Mathematics
Addison-Wesley
Redwood City, 1988

26. King Robert
Digital Filtering in One and Two Dimensions
Plenum
New York, 1989

27. Lippmann Richard
An Introduction to Computing with Neural Networks
IEEE ASSP Magazine
Abril 1987

28. Lippmann Richard
Neuronal Network Classifiers from Speach Reconigtion
Lincoln Laboratory Journal, Vol. 1 No. 1
MIT 1988

29. Lippmann Richard
Review of Neuronal Network
MIT Lincoln Laboratory, Lexington, MA 02173
USA 1989

30. Minsky Marvin, Papert Seymour
Perceptrons (An Introduction to Computational Geometry)
The MIT Press, Inst. Technology Cambridge
London, 1988

31. Morgan Nelson
Artificial Neural Networks, Electronic Implementations
IEEE, Computer Society
Los Alamitos, California, 1990

32. O'Shaughnessy Douglas
Speech Communication, Human and Machine
Addison-Wesley
Massachusetts, 1987

34. Rabinner Lawrence
Digital Processing of Speech Signal
Prentice Hall
New Jersey, 1987

35. Rabinner Lawrence
Digital Signal Processing
IEEE
Nueva York, 1972

36. Rabinner Lawrence
Multirate Digital Signal Processing
Prentice Hall
New Jersey, 1985

37. Rabinner Lawrence
Theory and Application of Digital Signal
Prentice Hall
New Jersey, 1975

38. Reddy Rag
Speech Recognition
Departament of Computer Science
IEEE, Academic Press.Carnegie Mellon University
Pittsburgh Pensilvania, 1974

39. Rogers David
Statistical Prediction with Kanerva's
Sparse Distributed Memory
Research Intitute for Advanced Computer Science
MS230-5, NASA Ames Research Center
Moffett Field, CA 94035

40. Rogers David
Weather Prediction Using a Genetic Memory
Research Institute for Advanced Computer Science
Moffett Field, CA

41. Schafer Ronald
Speech Analisys
IEEE
New York, 1979

42. Surkan Alvin, Di Liping
Fast Trainable Pattern Classification by
a Modification of Kanerva's SDM Model
Departament of Computer Science and Engineering
Departament of Geography University of Nebraska-Lincoln

43. Waibel Alex, Hanasawa T. y otros
Phoneme Recognition Using Time-Delay Neural Networks
Tecnical Report Telephony Research Laboratories, Japan
IEEE Transactions on Acoustics, Speech and Signal Processing
Vol. 79, Marzo, 1989

44. Yuhas Ben
Neural Networks Models of Sensory Integration
for Improved Vowel Recognition
IEEE Proceedings, Vol. 78, Octubre 1990