En este capítulo se presentan las bases teóricas de redes neuronales y modelos ocultos de Markov. Se hablará de las redes feed-forward y de las redes recurrentes así como su localización en la arquitectura general del reconocimiento automático de voz.

2.1 Redes neuronales

Los modelos de redes neuronales están compuestos de varios nodos simples operando en paralelo y arreglados en patrones, simulando redes neuronales biológicas. Los nodos suman *N* entradas multiplicadas por el peso de la conexión correspondiente. Al resultado de la suma ponderada se le aplica una función de activación no lineal como se muestra en la *figura 2.1* . El nodo es caracterizado por un umbral interno o un desplazamiento Θ y por el tipo de una función no lineal. En la *figura 2.1* hay varios tipos de funciones no lineales: escalón, elementos de umbral lógico y no lineales sigmoidales.

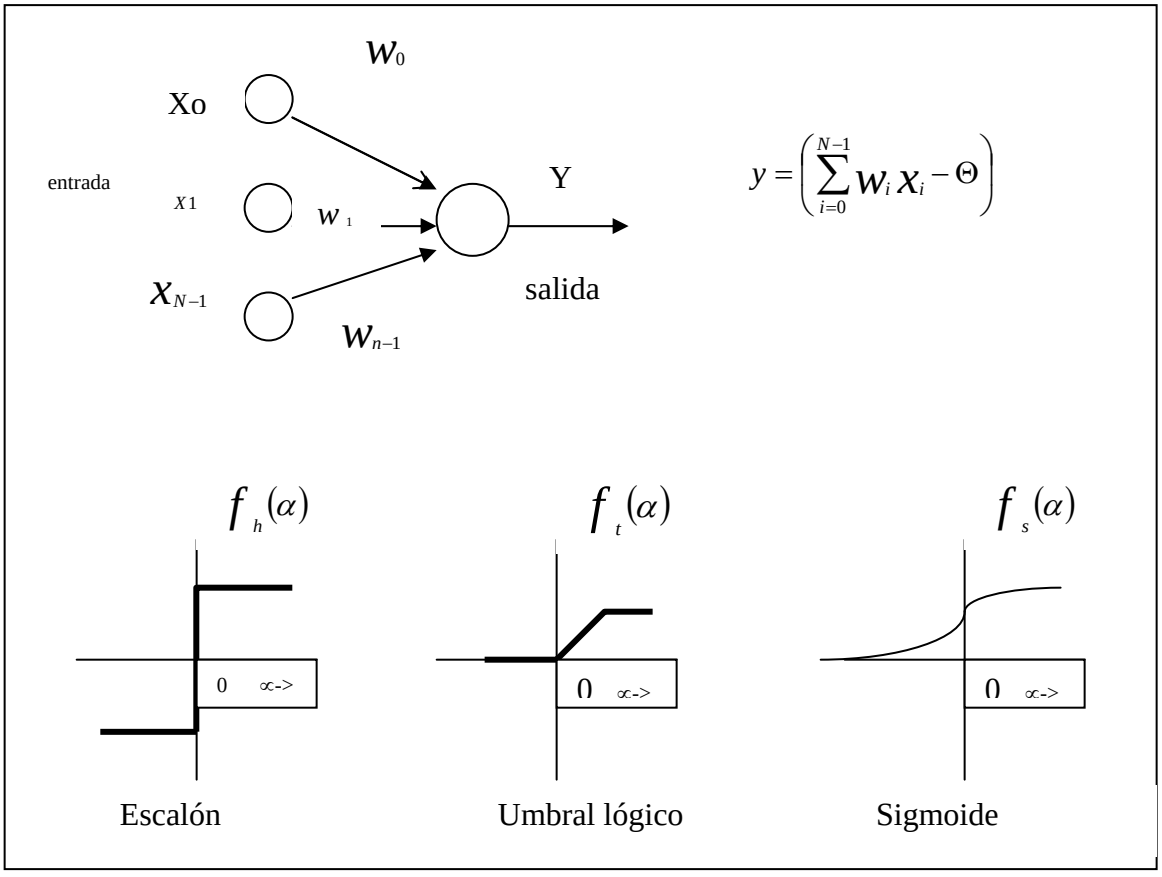

Figura 2.1 Suma ponderada de N nodos multiplicados por sus pesos correspondientes, activadas por una función no lineal. Abajo se muestran tres funciones no lineales.

En el desarrollo del entrenamiento del reconocedor se utiliza un red neuronal de tipo *feed-forward*, dicha red contiene ciertas características que la hacen diferente a los modelos ocultos de Markov y que ahora explicaremos.

2.1.1 El perceptrón multicapa

Entre las clases de redes neuronales que hay, los perceptrones multicapas han sido probados con éxito en la solución de varios tipos de problemas. Los perceptrones multicapa son redes *feed-forward* (alimentación hacia delante) con una o más capas de nodos entre los nodos de entrada y los nodos de salida. Estas capas adicionales contienen nodos ocultos que son directamente conectados a los nodos de entrada o de salida. Aquí es mostrado un perceptrón de tres capas con dos capas de nodos ocultos, *figura 2.2.*

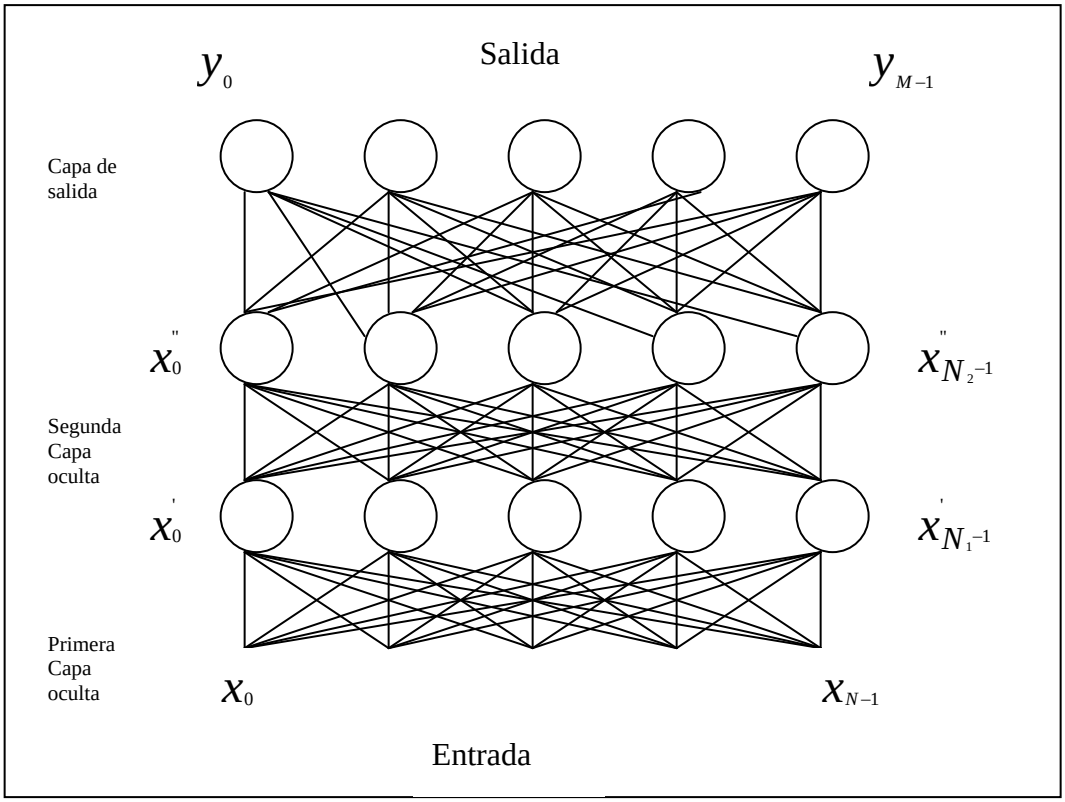

Figura 2.2 Perceptrón de tres capas con N entradas continuas con valor, M salidas, y 2 capas de unidades(nodos) ocultas.

En el estudio de las redes neuronales, la actividad χ_i, de una celda *i* es definida como el rango promedio de acción potencial alrededor del nivel espontáneo. La actividad de la celda *i* es comunmente escogida como una función de parámetros esenciales:

$$\chi_i = F\left(x; \vec{wi}, \Theta_i \right). \qquad (1.1)$$

Aquí la dureza de las conexiones sinápticas mediante el vector de entradas $\vec{x} = \{\chi_1, \chi_2, ..., \chi_N\}$ con otras celdas de la red desde *N* son caracterizados por el conjunto de pesos $\vec{wi} = \{w_{i1}, w_{i2}, ..., w_{iN}\}$. Un umbral esencial Θ_i determina el rango de valores para la cual la célula será manejada desde una actividad baja hasta una alta.

La actividad de la celda *i*, $F\left(\vec{x}; \vec{w_i}, \theta_i \right)$, puede ser descompuesta en dos partes. La primera es una cantidad escalar resultado de la acumulación de la entrada desde las otras celdas en la red, mediada por los pesos $\vec{w_i}$. Dos definiciones comunes para esta suma son:

$$d_i = \sum_{j=1}^{N} w_{ij} x_j - \theta_i, \qquad (1.2a)$$

y:

$$d_i = \left(\sum_{j=1}^{N} \left(w_{ij} - x_j \right)^2 \right)^{1/2} - \theta_i. \qquad (1.2b)$$

EL valor d_i es una medida de la similitud del patrón de actividad de entrada $\vec{x}$, a el conjunto de pesos $\vec{w_i}$ conectando la celda *i* a *N* celdas en la red. En la ecuación *1.2a* describe un producto interno entre el vector de patrón de actividad y el *i*-ésimo vector del peso de la matriz. En cada instancia, el umbral θ_i determina la escala para la medida de similitud, entonces las celdas con distancias similares o productos internos puedan ser más o menos acorde con el total de actividad de entrada. En la ecuación *1.2ª*, el umbral puede ser visto como un peso adicional (con signo negativo) conectando a la celda *i* hacia una entrada la cual siempre es igual a uno. Esto puede ser más compacto escrito como;

$$d_i = \sum_{j=1}^{N} w_{ij} x_j \quad ,$$

De esta forma, la suma incluye el peso $w_i, N+1 = -\theta_i$ y $x_{N+1} = 1$. A menos de que esté indicado de otra manera, la forma mostrada en la ecuación *1.2a* será empleada. Cada medida similar tiene ventajas y desventajas. No hay un criterio simple para determinar cual es la óptima, aunque dependería el diseño del criterio para un red neuronal, y el problema al cual la red neuronal será aplicada [MORGAN91].

2.1.2 Redes neuronales Espacio-temporales

Son sistemas de inteligencia artificial los cuales resuelven problemas en tiempo real a menudo requieren un tipo de razonamiento donde estén disponibles el tiempo y el espacio. Algunos sistemas de razonamiento temporal modelan el tiempo usando enfoques de tiempo y espacio. En este enfoque, la palabra es descrita como una secuencia de estados. Un estado es aquel que ocurre en un instante de tiempo. Otros sistemas modelan el tiempo usando un enfoque de eventos, donde un evento es asociado con un intervalo de tiempo más que con un punto en específico.

El parámetro del tiempo puede ser explícito en el estado o asumido como implícito, y los datos temporales como voz, están en una dimensión y pueden ser tratados como una clase de secuencias, mientras que los datos espaciales tienen una estructura tridimensional. Sin embargo, el razonamiento temporal y el razonamiento espacial son similares en el sentido de que el anterior afecta las relaciones de precedencia entre eventos y después entre objetos. Las relaciones de precedencia son un elemento fundamental de los patrones espacio-temporales. De hecho la parte más difícil de los patrones de aprendizaje de espacio-temporal podrían ser cómo, identificar, representar y guardar tales relaciones.

En relación a todo esto para representar y guardar el factor de tiempo o las relaciones de precedencia entre eventos, una red neuronal requiere de ciertas características especiales. Además se analizarán dos de las redes usadas para esta clase de problema, las cuales son: la red time-delay y la red recurrente, esta última será la que detallemos más a fondo.

2.1.2.1 Red neuronal recurrente

Un elemento muy importante en los patrones de aprendizaje de espacio-temporal es la identificación de las relaciones temporales entre las unidades de voz (palabras o fonemas). Se encontró que una red feed-forward es incapaz de aprender estas relaciones, las redes recurrentes si pueden hacerlo ya que guardan información temporal y de alguna forma manejan el aprendizaje de las relaciones temporales.

Las redes neuronales recurrentes tienen conexiones recurrentes que alimentan las unidades hacia atrás como una entrada adicional. Estas conexiones recurrentes pueden estar entre alguna de estas unidades de la red(entrada, unidades ocultas o de salida).
En el ejemplo que vemos en la *figura 2.3*, los contenidos de la capa oculta son copiados hacia la capa de contextos. En el siguiente ciclo del proceso los contenidos de la capa de contexto son alimentados hacia atrás, a una capa oculta junto con la nueva información, dando un espacio corto de memoria para los eventos anteriores. En otras palabras, la red neuronal puede integrar información temporal fijando hacia atrás a el punto de inicio

usando un número limitado de neuronas. Además la capacidad de memoria no es lineal con respecto a las unidades, en contraste con las redes time-delay las cuales usan memoria lineal.

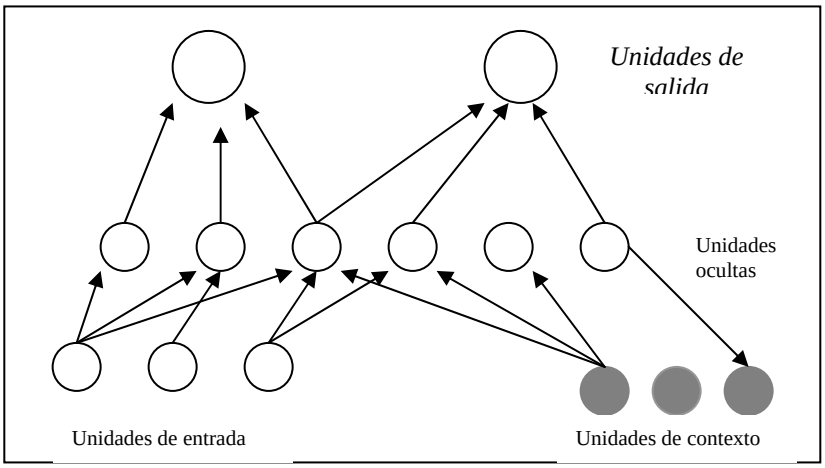

Figura 2.3 Red neuronal recurrente

Las redes *time-delay* no integran información temporal sobre el tiempo explícito, en el sentido de que aquellos desarrollan sus operaciones en un solo ciclo de trabajo. Integran la información obtenida durante un período de tiempo fijo, siendo representados espacialmente sobre la capa de entrada. Además con el enfoque de las time-delay la capacidad de la integración temporal es absolutamente limitada, por lo tanto esta es la habilidad de aprendizaje de los patrones temporales.

Sin embargo, no se usan las redes neuronales recurrentes para el reconocimiento automático de voz muy seguido, ya que la red no tiene información directa acerca del historial de la secuencia de patrones, teniendo que aprenderlo primero por los pesos recurrentes de la red. Todo este proceso lleva mucho tiempo de entrenamiento lo cual es muy malo donde se tengan corpus muy grandes. Para esto se recurre al algoritmo de aprendizaje cascade-correlation.

2.1.2.2 El algoritmo recurrente de aprendizaje cascade-correlation (RCC)

La arquitectura de dicho algoritmo recurrente es un método para generar un tipo especial de red neuronal. Ofrece muchas ventajas sobre otras redes neuronales, como darnos una topología mínima multicapa definiendo su propio tamaño durante el entrenamiento.

Esta arquitectura fue propuesta por Fahlman y Lebiere [FAHLMAN91], con el propósito de generar un red donde cada unidad aprenda una tarea específica tan rápido como sea posible, evitando el movimiento aleatorio de las unidades ocultas en el espacio, como sucede en la propagación hacia atrás estándar. Dicha red es creada en forma de cascada. Empieza sin unidades ocultas y las añade una por una durante el entrenamiento mientras los errores sobrepasan un determinado umbral. Una pila de unidades oculta es entrenada y la única con el mejor puntaje es escogida y añadida a la red. Cada unidad nueva recibe la activación de todas las demás previamente instalada y desde su propia cadena recurrente. Sin embargo su activación nunca es pasada a las unidades previamente instaladas, así de este modo se crea la estructura de cascada.

El algoritmo de aprendizaje trata de maximizar la correlación entre la salida de las nuevas unidades y el error residual que está tratando de eliminar. Para esto usa una función simétrica sigmoidal para todas las unidades con un rango de 0.0 a 1.0 y el algoritmo *quickprop* para entrenar los pesos de salida [KIRSCHNING98].

2.1.2.3 La arquitectura de la red neuronal de cascada paralela

Las unidades ocultas en la red neuronal RCC cooperan en la solución del problema con todas, y cada una de las nuevas unidades ocultas, afectadas por la activación de las anteriores. Cada nueva unidad se especializa más en el problema utilizado que en la adquisición de conocimiento por las unidades ocultas. Sin embargo, para el aprendizaje de un número grande de patrones de entrenamiento debe ser dividido en lecciones muy pequeñas las cuales serán entrenadas por separado una atrás de la otra, de la más simple a la más compleja. Esto es conocido como descomposición del problema, y es usado para facilitar el entrenamiento y aprendizaje de grandes conjuntos de ejemplos. Después del

entrenamiento cada lección separadamente de la red es re-entrenada con todos los ejemplos en un sólo conjunto de entrenamiento.
El re-entrenamiento es hecho en la misma forma en que las lecciones fueron entrenadas, esto es, manteniendo las conexiones de las entradas de las unidades estáticas instaladas anteriormente, así como su respectiva liga recurrente. Los grupos de unidades ocultas generadas durante el entrenamiento de cada lección crecen en forma de cascada una arriba de la otra. Cuando la red es finalmente re-entrenada con el conjunto completo de entrenamiento, al menos un grupo de unidades es creado, llamando a cada grupo módulo.

2.2 Redes neuronales y reconocimiento de voz

Dentro del proceso de reconocimiento la red neuronal sirve de clasificador. Tomando como punto de partida el primer elemento de la estructura general de un reconocedor de voz, es decir, una vez obtenida la señal de voz que se desea reconocer, es necesario calcular un conjunto características esenciales, que serán las entradas a la red neuronal. Dichas características de la señal de voz se reflejan en las propiedades del tracto vocal de la palabra que se produce en determinado momento. Todo esto obtenidas directamente de un espectro de voz. Para comprender mejor esto, introduciremos los conceptos de *frames* y ventanas de contexto.

Los frames son porciones de la señal digital de la cual se puede extraer la información más importante, de tal manera que se reduzca la información que esta asociada con esa porción de señal. Básicamente para obtener un *frame* se debe:

- Elegir una porción de la señal de aproximadamente 30 milisegundos el cual corresponde al *frame*
- Crear un vector de parámetros que aproxime al espectro de esa porción.
- Avanzar 10 milisegundos y repetir el proceso hasta haber obtenido las características de la señal completa.

En este proceso las duraciones de cada *frame* deben ser los suficientemente largos para hacer una correcta y completa extracción de las características, pero lo más corto posible para que la señal permanezca dentro de los límites de dicho *frame*.

Para el proceso de extracción de características de cada *frame*, se encuentran las técnicas de procesamiento de señales como: la predicción lineal perceptual (PLP), Mel Frequency Cepstral Coefficients (MFCC), normalización de la energía y supresión de ruido.

2.2.1 Los vectores MFCC de características

Una de las técnicas más usadas para la obtención de parámetros es la codificación MFCC (PLP). Aquí se reduce el número de muestras de la señal de voz a un conjunto de coeficientes que representan las concentraciones de energía y anchos de frecuencia en la señal.

Debido a la naturaleza continua de la voz, ésta no sólo depende de las características espectrales en un instante en el tiempo, sino que depende de la variación de las características mientras pasa el tiempo.

Ya que se obtuvieron los *frames* de la señal, se introduce información contextual para hacer una clasificación más precisa. Aquí no sólo se tomará el vector de características que se desea reconocer, sino también las características de otros cuatro vectores localizados entre -60 y 60 milisegundos de dicho vector. La *figura 2.4* nos muestra dicho proceso.

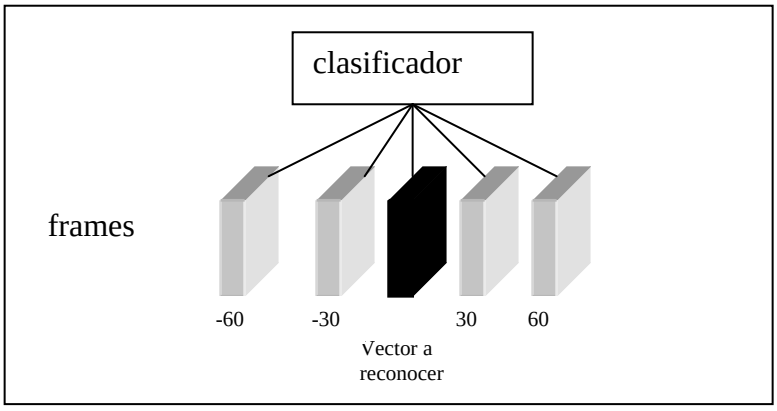

Figura 2.4 Ventanas de contexto

Al obtener la ventana de contexto, el conjunto de características será introducido al clasificador. El objetivo es clasificar cada unidad del habla de acuerdo con el conjunto de unidades especificadas (silabas, fonemas, etc.). En esta etapa es donde entra una de las dos técnicas de clasificación más conocidas, el enfoque de redes neuronales y los modelos ocultos de Markov.[1] El conjunto de características son la entrada al clasificador y como salida se obtiene una probabilidad por cada unidad fonética. Una vez que se procesa toda la señal de voz, resulta una matriz de probabilidades de tamaño *C x F* en donde *C* es el número de unidades que queremos clasificar y *F* es el número de segmentos. Entonces se realiza una búsqueda para encontrar la secuencia con mayor probabilidad de reconocimiento.

2.2.2 Búsqueda de Viterbi

Dicha búsqueda está basada a partir de la matriz anterior con la secuencia de palabras más probables. Aquí se usa información estadística de las duraciones máximas y mínimas de cada palabra con el objetivo de restringir las opciones. A medida que los límites se afinan, el nivel de reconocimiento aumenta.

No sólo se busca la ruta con mayor probabilidad, sino que es un requisito que la palabra resultante este dentro de un vocabulario que se necesita reconocer. Aquí hay varios caminos a seguir pero hay uno en especial para una palabra que maximizará la puntuación de las probabilidades. En la *figura 2.5* vemos un ejemplo de dicha búsqueda para la palabra UNO.

[1] De ahora en adelante nos referiremos a los Modelos Ocultos de Markov como HMM, por su significado en inglés (Hidden Markov Models).

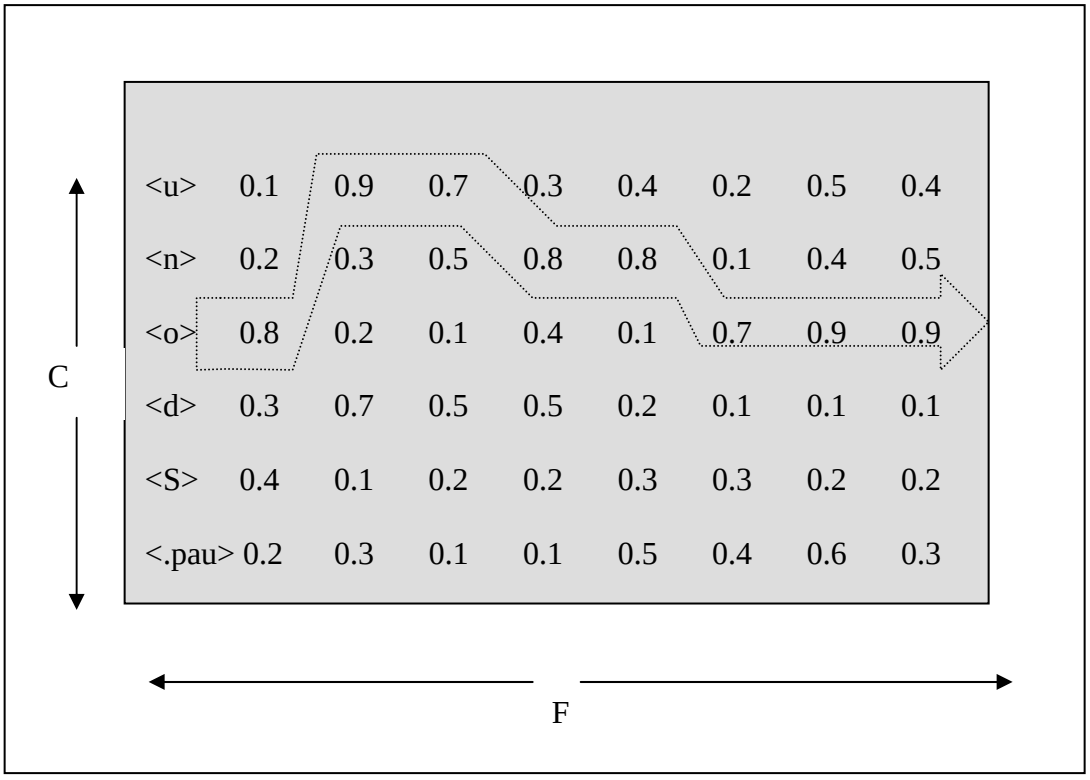

Figura 2.5 Espacio de la búsqueda de Viterbi

Finalmente, podemos decir que uno de los enfoques más importantes dentro de lo que ha sido el reconocimiento de sistemas automáticos de voz han sido las redes neuronales. En el presente capítulo se explicaron los elementos base para poder comprender un poco como se relaciona el campo de redes y de reconocimiento. En la siguiente sección comentaremos sobre otro enfoque el cual, al igual que este, es importante pero no tan explotado.

2.3 Antecedentes de modelos ocultos de Markov

En esta sección, se comentará un poco de historia sobre los HMM, definiremos algunos conceptos importantes como: cadenas de Markov y modelos ocultos de Markov. Además mostraremos cual es la relación que hay entre dichos modelos y los sistemas automáticos de reconocimiento de voz.

Desde finales de los 70's, cuando los modelos ocultos de Markov fueron aplicados a sistemas de reconocimiento de voz, se desarrollaron técnicas para estimar probabilidades sobre estos sistemas en específico. Estas técnicas permiten a los modelos ocultos de Markov llegar a ser eficientes, robustos y flexibles de manera computacional. También les ha permitido servir de base de muchos sistemas de reconocimiento con vocabularios grandes e independientes del locutor por sus características alcanzadas.

Los modelos ocultos de Markov fueron introducidos por Baum en un artículo, el cual propuso este modelo como un método estadístico de estimación de las funciones probabilísticas de una cadena de Markov[2]. Esencialmente, los modelos ocultos de Markov son métodos para modelar sistemas con tiempo dependiente del comportamiento caracterizado por procesos comunes de corta duración y la transición entre ellos. Un HMM puede ser pensado como una máquina de estado finito donde las transiciones entre los estados dependen de la ocurrencia de algún símbolo. Asociado con cada transición de estado una salida de la distribución de probabilidad describe la probabilidad con la cual un símbolo ocurrirá durante la transición, y una probabilidad de transición indicando la probabilidad de esta transición. En general, un HMM es un subproceso estocástico asociado a una secuencia probabilística de símbolos. Este tipo de modelo está formado por una cadena de Markov y un conjunto de funciones de probabilidad asociada a cada estado; es decir, los estados ya no son simplemente símbolos sino que ahora se les asocian grupos con distribuciones de probabilidad.

2.3.1 Definición de un HMM

Los Modelos de Markov describen un proceso de probabilidad el cual produce una secuencia de eventos o símbolos observables. Son llamados ocultos porque hay un proceso de probabilidad subyacente que no es observable, pero afecta la secuencia de eventos observados [Morgan91].

Los modelos ocultos de Markov pueden ser vistos como el modelo de un proceso, el cual produce una secuencia de eventos acústicos perteneciendo a una unidad específica, o

[2] Ver Apéndice C

palabra, en un vocabulario dado. Las variaciones entre las secuencias de observaciones de la misma clase, como la longitud de una palabra y pronunciación, son modelados por la naturaleza del elemento estocástico de un HMM. Un sistema automático de reconocimiento de voz generalmente tendrá *n* modelos ocultos de Markov, uno para cada clase. El modelo reconoce la palabra cuando el estado final es alcanzado [RABINER86].

Entonces podemos definir los estados de la cadena la cual genera datos observables donde se tiene:

1.- Un alafabeto de salida $\mathcal{Y}$ ={0,1,...,b-1}

2.- Un espacio de estados *l* = {1,2,...,c} con un único estado inicial s_0

3.- Una ditribución de probabilidad de la transición entre los estados $p(s'|\, s)$, y

4.- Una distribución de probabilidad de salida *q(y|s,s')* asociadas con las transiciones del estado s al estado *s'*.

Entonces la probabilidad de obervar una cadena de salida HMM $y_1, y_2, ..., y_k$ está dada por $p(y_1, y_2, ..., y_k) = \sum_{s_1,\cdots,s_k} \prod_{i=1}^{k} p(s_i \mid s_{i-1}) q(y_i \mid s_{i-1}, s_i)$

Ahora definiremos lo que es un HMM de manera formal:
Sea λ la quíntupla: $\lambda = \{N, M, \pi, A, B\}$

- donde **N** es el número de estados del modelo.
  y sea S = {1,2,3,...,N} el conjunto con *N* estados en el modelo.
- **M** es el número de símbolos (*V*) de observación
  Y $V=\{v_1, v_2, v_3, ..., v_M\}$ el conjunto con *M* símbolos.
- **π** es la distribución de los estados
  $\pi = \{\pi_i\}$
  $\pi_i = P[q_1 = i] 1 \le i \le N$
- **A** es la probabilidad de transición de un estado a otro

A={ a_{ij} }

$a_{ij} = P[q_{t+1} = j \mid q_t = i], 1 \le i, j \le N$ donde $q_t = i$ es el estado i en el tiempo t

- **B** son las probabilidades de cada símbolo dado cada estado

  B={ $b_j(k)$}

  $b_j(k) = P[O_t = v_k \mid q_t = j], 1 \le k \le M$

  O_t es la observación v_k en el tiempo *t*

La notación compacta $\lambda = (A, B, \pi)$ es usada para representar a los HMM. Especificar un HMM nos envuelve en escoger el numero de estados, *N*, tantos como números de símbolos discretos haya, *M*, y especificando el árbol de probabilidades de densidad, *A*, *B*, y π.

En la función de probabilidad existen dos diferentes funciones para calcular las probabilidades dependiendo del tipo de variable que se utilice, estas pueden ser discretas o continuas: en el caso de las variables discretas la función calcula una probabilidad por cada valor discreto de la variable, la función consiste en obtener distribuciones normales para los valores de las variables y haciendo uso de la media y de la varianza se puede calcular la probabilidad.

2.3.2 Estructura general de un reconocedor basado en HMM

La estructura general de un reconocedor basado en HMM puede ser vista como un módulo donde entra la señal de voz, normalmente ésta es almacenada como un archivo de sonido en formato *wav*. Después, pasa a un análisis espectral el cual produce una secuencia del vector de características, posteriormente se aplica la técnica de Vector Quantization (VQ) la cual describiremos más adelante. En este paso se obtiene una secuencia de símbolos. En la fase de entrenamiento, se aplica el método de HMM y se crea un modelo para cada palabra o fonema según sea el caso. En la fase de reconocimiento son calculadas las mayores probabilidades y se procede al reconocimiento de la palabra obteniendo los resultados de dicho reconocimiento. Este proceso lo podemos ver en la *figura 2.6*.

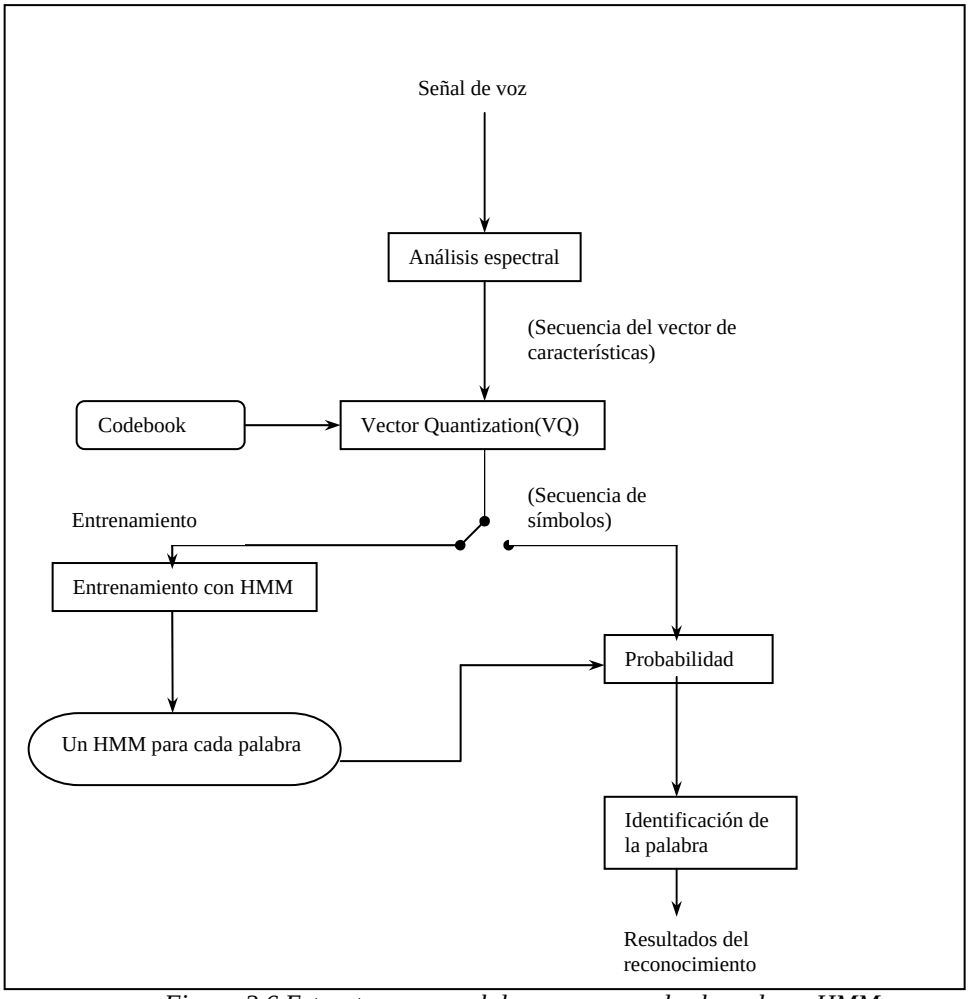

Figura 2.6 Estructura general de un reconocedor basado en HMM

En el enfoque de HMM cada palabra es modelada por una red de transición la cual tiene un pequeño número de *N* estados, los cuales corresponden a las unidades fonéticas que se quieren reconocer (fonemas o palabras) existan. Cada estado físicamente corresponde en un sentido indeterminado a un conjunto de eventos temporales en la palabra pronunciada. En la *figura 2.7* aparece un ejemplo con 5 estados.

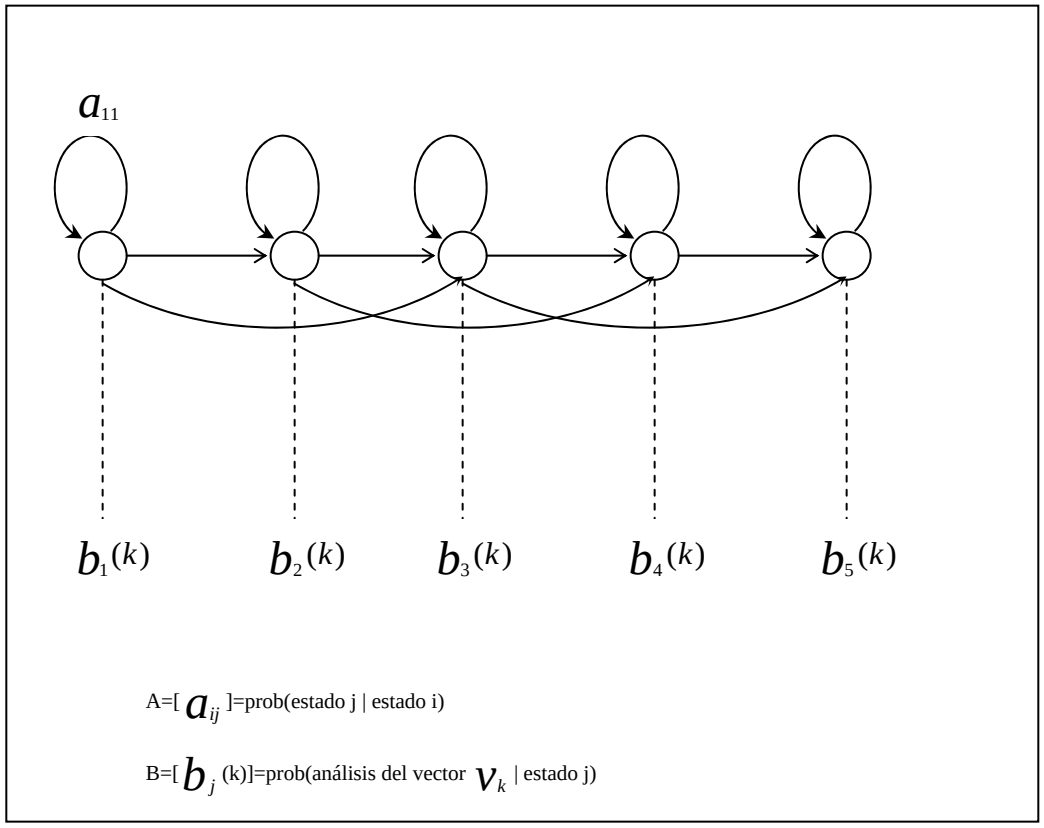

Figura 2.7 Ejemplo de HMM para el reconocimiento de una palabra.

Todo el modelo de Markov es caracterizado por la probabilidad de la observación del símbolo b_j(k) para el k-ésimo elemento del *libro de código* (codebook) en cada estado *j*, la probabilidad de transición del estado a_{ij} que describe como el nuevo estado *j* puede ser alcanzado desde el estado anterior *i*, y la probabilidad del estado inicial π_i. π_i indica la probabilidad que el término *i*-ésimo estado, es el estado inicial de transición.

A=(a_{ij}) y B=(b_j(k)) son respectivamente llamados la matriz de transición de estados y matriz de frecuencia de probabilidad. A, B y $\pi = (\pi_i)$ consecuentemente representan

variaciones temporales y espectrales en cada palabra. Todos ellos calculando y usando los datos de entrenamiento [FURUI89].

En la fase de reconocimiento para la salida desconocida, la probabilidad de la secuencia observada, es generada desde una secuencia de estados y calculada para cada palabra del vocabulario, y la palabra con la más alta probabilidad es seleccionada como la identificación correcta. El alineamiento en el tiempo es obtenido indirectamente a través de la secuencia de estados. No obstante aunque el proceso de entrenamiento es complicado, el proceso de reconocimiento es muy simple.

2.3.3 Libro de código

En la codificación de la señal, cierto período es obtenido de la misma, y el patrón en este período es representado como un sólo código. Este procedimiento es realizado almacenando los patrones típicos de la señal (vectores de código o modelos) , y asignando un código a cada patrón. La tabla que indica la correspondencia entre los patrones y los códigos es llamado *libro de códigos*. Donde una señal es comparada con cada patrón en un predeterminado intervalo, y la señal en cada período es delineada por un código indicando el patrón que es más similar a la señal. Este libro también nos prové de un conjunto apropiado de patrones los cuales minimizan la distorción cuando varios tipos de señales son representadas por un número limitado de patrones. Hay dos métodos de generación de libros de código, uno basado en aprendizaje aleatorio y otro basado en clusters. El primero permite a los vectores ser seleccionados aleatoriamente de los datos de entrenamiento y almacenados como vectores de códigos. El segundo método está basado en el algoritmo de Lloyd [FURUI89], en el cual los datos de entrenamiento son agrupados en conjuntos no sobrepuestos y son calculadas las medias que minimizan la distorción promedio. Estas medias son almacenadas como vectores de código.

2.3.4 Vector Quantization

Los cuantizadores de vectores operan sobre los vectores de características obtenidos de la señal de voz. VQ[3] depende de la creación de un libro de código óptimo tal

[3] Nos referiremos a Vector Quantization como VQ.

que la distorsión promedio remplazando cualquier vector de características por la entrada más cerca del libro de códigos es minimizada [Morgan91]. Para un *libro de código* (el cual definiremos enseguida) de tamaño *Q* indexado con q, el objetivo es seleccionar el conjunto de vectores del libro de códigos. La ventaja de VQ es que permite etapas subsecuentes en el reconocedor para que su complejidad disminuya. Estas etapas sólo necesitan direccionar las variaciones entre los vectores Q, o todos los que sean posibles. Las relaciones entre los vectores Q podrían ser pre-calculadas, salvando el tiempo de procesamiento. Estas relaciones pueden incluir métricas de distorsión, o la probabilidad que de que el vector $\vec{c}_q$ estará seguido por una entrada $\vec{c}_{qj}$.

2.3.5 Los tres problemas de los HMM

Existen tres problemas principales que deben ser resueltos cuando utilizamos el HMM.

- El problema de evaluación
- El problema de la secuencia de estados ocultos no cubiertos.
- El problema de entrenamiento

La solución al primer problema es utilizado para marcar cada palabra basada en la secuencia de observación para reconocer una palabra no conocida. La solución al segundo problema es usada para desarrollar un entendimiento del significado físico de los estados del modelo. Y la solución al tercer problema es empleada para obtener óptimamente los parámetros del modelo para cada palabra del modelo usando las pronunciaciones de entrenamiento [FURUI89].

2.3.6 Otras consideraciones

En la actualidad hay muchas investigaciones en las cuales son empleados los HMM, pero en nuestro país son pocas las instituciones donde se aplican dichos modelos para el reconocimiento automático de voz. Es por eso que se decidió hacer un reconocedor de propósito general de alto desempeño basado en estos modelos, pues aún no existe una investigación fuerte en esta área aplicada al tema propuesto.

En esta tesis se hizo un reconocedor autómatico de voz utilizando este enfoque. Las herramientas para hacerlo fueron dadas por el CSLU Toolkit el cual permitió desarrollar dicho reconocedor con otro enfoque como redes neuronales. Algunos de los problemas a los que nos enfrentamos son: desarrollar un mal modelo de los HMM, no distribuir correctamente los porcentajes de datos para entrenamiento, desarrollo y prueba, al ocupar un corpus muy grande la complejidad del entrenamiento esta puede aumentar, entre otros, por eso se realizaron varias pruebas buscando el modelo óptimo el cual nos permita obtener un nivel de reconocimiento aceptable. Cabe mencionar que en el laboratorio de reconocimiento autómatico de voz de la UDLA se han hecho pocos experimentos con los HMM aplicados en reconocedores ya que en sus objetivos sólo se han enfocado a corpus de dígitos o pequeñas bases de archivos de voz las cuales no necesitan un modelado complejo.

En el capítulo que a continuación se presenta se hablará específicamente sobre el ambiente de desarrollo del CSLU tanto para redes neuronales como de HMM. En ese capítulo tomarán sentido algunos de los conceptos ya mencionados en ambos temas obteniendo así una relación entre dicho ambiente y los enfoques dados, permitiéndonos desarrollar los reconocedores.