UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

PROGRAMA DE MAESTRÍA Y DOCTORADO EN INGENIERÍA

Sistema de reconocimiento de palabras clave para un robot de servicio

Tesis

que para optar por el grado de

Maestro en Ingeniería

Presenta

José David Galindez Vergara

TUTOR PRINCIPAL: DR. JESÚS SAVAGE CARMONA

Facultad de Ingeniería

México, D. F. enero 2015

Jurado asignado

Presidente: Dra. Medina Gómez Lucia

Secretario: Dr. Pérez Alcázar Pablo Roberto

Vocal: Dr. Savage Carmona Jesús

1 er. Suplente: Dr. Rivera Rivera Carlos

2 do. Suplente: M. I. Escobar Salguero Larry

Lugar donde se realizó la tesis: Laboratorio de Bio-robótica de la Facultad de Ingeniería de la Universidad Nacional Autónoma de México.

TUTOR DE TESIS:

DR. JESÚS SAVAGE CARMONA

FIRMA

Agradecimientos

Quiero agradecer al Posgrado en Ingeniería Eléctrica por darme la oportunidad de estudiar esta maestría y así aumentar mis conocimientos; a los profesores del departamento de Procesamiento Digital de Señales por todas sus enseñanzas; al doctor Jesús Savage Carmona por sus enseñanzas y apoyo y a mis colegas y amigos del Laboratorio de Bio-robótica y del posgrado por toda su ayuda y amistad.

Se agradece a la DGAPA-UNAM por el apoyo proporcionado para la realización de esta tesis a través del proyecto PAPIIT IN117612: Robot de Servicio para Asistencia a Adultos Mayores y en Sistemas Hospitalarios, y por supuesto agradezco al CONACYT ya que se contó con el apoyo de una beca para estudiar la maestría.

Resumen

Esta tesis consta de la implementación de un sistema de reconocimiento de voz, el cual será usado por un robot de servicio autónomo que realizará un conjunto de tareas en un ambiente doméstico según las órdenes que se le comuniquen de manera oral. Para realizar ésto, se procesa la voz mediante una serie de técnicas y se detectan las palabras clave contenidas, las cuales representan partes de las órdenes a seguir por el robot.

Para realizar el reconocimiento de voz primero se aplicaron técnicas de procesamiento digital de señales que adecuan la señal de voz para poder ser analizada; éstas se basan en el análisis LPC y cepstral, y sirven para obtener una serie de vectores que describen de la señal de voz. Después se procesaron los parámetros calculados para generar los modelos que describan las diferentes clases de voz a reconocer, utilizando técnicas de reconocimiento de patrones basados en los Modelos Ocultos de Markov que utilizan métodos de estadística y probabilidad para la generación de modelos, ésto en la fase de entrenamiento, y en la fase de reconocimiento se emplearon los modelos obtenidos para compararlos con los que se obtienen de la voz a reconocer y se decide a que clase pertenecen.

Índice general

Agradecimientos **3**

Resumen **5**

1. Introducción **13**

2. Fundamentos y Antecedentes **17**

2.1. Fundamentos del Reconocimiento de Voz 17

2.1.1. Modelo acústico fonético 17

2.1.2. Reconocimiento de patrones 18

2.1.3. Inteligencia Artificial 18

2.2. Procesamiento Digital de Voz 19

2.2.1. Modelado de la voz 20

2.2.2. Representación de la señal de voz 20

2.2.3. Cuantización vectorial (CV) 26

2.3. Reconocimiento de palabras aisladas utilizando CV 27

3. Modelos Ocultos de Markov **29**

3.1. Descripción de Modelos ocultos de Markov 29

3.1.1. El problema de evaluación 31

3.1.2. El problema de decodificación 32

3.1.3. El problema de aprendizaje 33

3.2. Modelos ocultos de Markov con funciones de probabilidad continuas . 36

3.3. Reconocimiento de palabras aisladas utilizando Modelos Ocultos de Markov 38

3.4. Reconocimiento de palabras clave en voz continua 39

4. Detección de palabras clave **41**

4.1. Estado del arte 41

4.2. Clasificación de los sistemas de reconocimiento de voz 42

4.2.1. Reconocimiento de voz continua 42

4.2.2. Detección de palabras clave 43

4.3. Redes Neuronales Artificiales 45

4.3.1. Aprendizaje 48

4.3.2. Reconocimiento de voz utilizando Redes Neuronales Artificiales 49

5. Desarrollo del sistema de reconocimiento de voz **51**

5.1. Motivación del trabajo 51

5.2. Desarrollo 52

5.3. Obtención de la señal de voz 53

5.4. Procesamiento digital de la señal de voz 53

5.5. Codificación de la señal 55

5.5.1. Obtención de coeficientes LPC 55

5.5.2. Obtención de coeficientes cepstrales 56

5.5.3. Obtención de coeficientes mel cepstrales 57

5.5.4. Cuantización vectorial 58

5.5.5. Obtención del codebook 59

5.6. Implementación de los Modelos Ocultos de Markov para el reconocimiento de palabras aisladas 59

5.6.1. Entrenamiento 60

5.6.2. Reconocimiento 61

5.7. Implementación de los Modelos Ocultos de Markov para el reconocimiento de palabras clave 61

5.7.1. Entrenamiento 61

5.7.2. Reconocimiento 63

6. Pruebas y resultados **65**

7. Conclusiones **71**

7.1. Conclusiones 71

7.2. Trabajo futuro 72

A. Código de algunas funciones y algoritmos en el lenguaje de programación C sharp **73**

A.1. Entrenamiento Viterbi de un Modelo Oculto de Markov 73

A.2. Clasificador utilizando un grupo de Modelos Ocultos de Markov . . . 77

A.3. Obtencion de coeficientes cepstrales y delta cepstrales 78

A.4. Transformacion de coeficientes LPC a cepstrales 79

B. Glosario **81**

Bibliografía **85**

Índice de figuras

2.1. Reconocimiento de palabras aisladas con CV 28

3.1. Gráfico de un Modelo Oculto de Markov 30

3.2. Reconocimiento de palabras aisladas con HMM 39

4.1. Neurona Artificial 45

4.2. Red Neuronal Artificial 47

5.1. Diagrama de bloques del reconocimiento 53

5.2. Diagrama de bloques del procesamiento digital 54

5.3. Gráfico de un Modelo Oculto de Markov no regresivo 60

5.4. Modelo Oculto de Markov General 62

6.1. Lista de palabras 66

6.2. Resultado de detección de palabras clave. 69

Capítulo 1

Introducción

La voz o el lenguaje hablado es la forma común de comunicación entre las personas, se aprende en los primeros años de vida de manera fácil, natural e inconsciente; sin notar realmente lo complejo que es este proceso dentro de nuestro cuerpo y mente. Los humanos se comunican por medio del habla de una manera tan natural sin ninguna dificultad y aun en las condiciones más adversas. Al ser la forma más directa de comunicación, es la manera idónea en que el hombre busca interactuar con dispositivos que utiliza en la vida diaria, como las computadoras y las máquinas.

El ser humano ha desarrollado sistemas que puedan auxiliarlo en diversas tareas; las máquinas lo ayudan en tareas mecánicas difíciles o rutinarias y las computadoras lo apoyan en tareas lógico matemáticas y todo lo que se puede derivar de ellas; estos sistemas pueden realizar este trabajo de forma autónoma, eficiente y a veces mejor que las mismas personas. La robótica se encarga de la creación de sistemas capaces de realizar estos dos tipos de tareas y el gran desafío de ésta es la construcción de robots totalmente autónomos que realicen las mismas acciones de los seres humanos inclusive de una mejor manera. Una tarea muy importante es la comunicación oral; pues por comodidad se necesita que estos sistemas realicen el trabajo de forma parecida a los humanos, ya que también se desea la interacción entre ambos. Por esta razón se desea la creación de sistemas capaces de entender el lenguaje hablado por las personas. El reconocimiento de voz tiene un amplio rango de aplicaciones, por ejemplo, el

dictado, la comunicación con un robot, asistencia de personas con problemas de visión, traducción y cualquier interacción de hombre máquina.

Mientras las tareas aritméticas y lógicas como la realización de cálculos, así como la memorización de una gran cantidad de datos se pueden realizar de forma fácil y rápida por los dispositivos digitales superando fácilmente a las seres humanos; otras tareas más complejas que realizan los humanos con naturalidad son muy difíciles de llevar a cabo perfectamente por estos dispositivos, una de estas es la comunicación oral, pues la voz no es un proceso determinístico si no que es un proceso pseudoaleatorio, por lo cual no es tan directo su análisis.

La producción de voz es un proceso no lineal, afectado por muchos factores como la entonación, el estado de ánimo, la edad y el género de la persona que habla; el proceso de reconocimiento de voz es aun más complejo, pues además de ser influenciado por estas variables es afectado por el ruido externo y los fenómenos acústicos que ocurren en el ambiente. Esto quiere decir que si nosotros pronunciamos alguna palabra dos veces, la grabamos y observamos la forma de onda de la señal nos daremos cuenta de que no son iguales, incluso si fueran frases. La voz se puede dividir en segmentos, siendo el más pequeño el fonema, el cual normalmente corresponde a una letra del alfabeto, y aunque para nosotros un mismo fonema suene casi igual, su forma de onda variará debido a la naturaleza de la producción de voz. Estas variaciones dependerán de cuales fonemas están alrededor del de interés, de la palabra a la que pertenecen, de la frase a la que pertenecen y de lo que comunicamos dada nuestra entonación, y al final tendrán variaciones según la persona que hable, la presencia de ruido y la propia naturaleza aleatoria de la voz. Debido a esto se busca representar la voz en una forma que estas variaciones sean más manejables.

El problema de reconocimiento automático de voz ha sido abordado por muchas personas durante mucho tiempo, han existido muchas contribuciones y avances al respecto, lo cual ha hecho mejorar estos sistemas continuamente; sin embargo, todavía queda mucho que avanzar en la materia. Ningún sistema es infalible y es susceptible a muchas condiciones adversas como el ruido, la diferencia de pronunciación de las personas y hasta el volumen con que se habla; además, tenemos que considerar diferentes detalles que se deben resolver en cada idioma que se desea implementar,

así como la manera en que se implementa, como por ejemplo si es una computadora o en un dispositivo lógico programable y si trabaja en tiempo real o no.

Hoy en día las tecnologías están aun lejos de compararse con la habilidad de un humano al reconocer voz, pues éste, además de reconocer en entornos tan difíciles, puede identificar la persona que habla, diferenciar un acento por medio de la entonación y comprender significados implícitos, hasta diferenciar si es verdad y aun deducir partes faltantes sin necesidad de escuchar todo. Por esta razón algunos pasos del reconocimiento de voz están inspirados o tratan de imitar la forma en que los humanos realizamos este mismo proceso, por ejemplo, utilizando métodos estadísticos, de Inteligencia Artificial o empleando redes neuronales artificiales que tratan de imitar a las redes neuronales biológicas. Cabe señalar que la capacidad de los seres humanos de realizar ésta y muchas otras tareas con tanta eficacia es la complejidad del cerebro humano que posee alrededor de 86 mil millones de neuronas, cada una realizando un pequeño proceso y estando conectada con miles de otras, además, este proceso es realizado en gran parte por el cerebro inconsciente que forma la mayor parte de la mente.

Esta tesis consta de la implementación de un sistema de reconocimiento de voz, el cual será usado por un robot de servicio autónomo que ejecutará tareas preestablecidas según las órdenes que se le comuniquen de manera oral. Estas tareas inicialmente constan de acciones comunes que se realizan en un ambiente habitacional, por lo que auxiliará en labores domésticas a personas que lo necesiten; por ejemplo personas de la tercera edad. Para realizar esto tendrá que procesar la voz mediante una serie de técnicas y reconocer palabras clave, las cuales representan partes de los comandos para realizar acciones; esto último tendrá que pasar a un sistema de procesamiento de lenguaje que interprete correctamente estas órdenes. Los algoritmos de reconocimiento de voz desarrollados en esta tesis serán utilizados por la robot Justina, en las tareas que ella ejecuta para ayudar a adultos mayores y para la atención hospitalaria.

El proceso de reconocimiento de voz se puede dividir en diferentes bloques. El primero es un bloque que incluye elementos de procesamiento digital de señales, los cuales adecuan la señal de voz para poder ser procesada, calculan diversos parámetros de ésta y al final tienen una serie de descriptores de la señal de voz, que generalmente

son vectores. El siguiente paso incluye el proceso de los parámetros calculados para generar los modelos que describan las diferentes clases de voz a reconocer, para esto se emplean técnicas de reconocimiento de patrones que utilizan procesos estadísticos o de inteligencia artificial para la generación de modelos y cuando se requiera realizar el reconocimiento se emplean los modelos obtenidos para compararlos con los que se obtienen de la voz a reconocer.

- En el primer capítulo presentamos la introducción de la tesis donde mencionamos la motivación del trabajo y objetivos del mismo así como algunos hechos importantes en torno al tema.
- En el segundo capítulo se describirá brevemente los fundamentos de procesamiento de señales que se utilizan para el adecuamiento de la señal voz y obtención de parámetros importantes para el reconocimiento, así como las técnicas de reconocimiento de patrones empleadas en el reconocimiento de palabras aisladas.
- En el tercer capítulo estudiaremos los métodos probabilísticos más utilizados en el reconocimiento de voz; los cuales se basan en los Modelos Ocultos de Markov, y la forma de aplicarlos al reconocimiento de palabras aisladas y al de palabras clave.
- En el cuarto capítulo se describirá el estado del arte en el área de reconocimiento automático del habla por una máquina y detección de palabras clave; mencionando algunos avances qué se han tenido y los métodos que se utilizan actualmente.
- En el quinto capítulo describiremos claramente como se desarrolló el sistema propuesto basado en los conceptos anteriores.
- En el sexto capítulo se listaran las pruebas que se realizaron y los resultados que se obtuvieron.
- En el séptimo se analizarán los resultados obtenidos, se presentarán las conclusiones y trabajo futuro.

Capítulo 2

Fundamentos y Antecedentes

2.1. Fundamentos del Reconocimiento de Voz

El problema del reconocimiento de voz por una máquina es un problema que ha sido abordado por muchos años y por muchos grupos de trabajo, teniendo progresos significativos aunque sin igualar la capacidad de reconocimiento de un ser humano. Existen distintas técnicas para lograr el reconocimiento de voz, las cuales, acorde a Rabiner y Juang [24] se pueden clasificar en tres grupos: Modelo acústico fonético, Reconocimiento de patrones e Inteligencia Artificial.

2.1.1. Modelo acústico fonético

Esta técnica se basa en la presunción de que a cada segmento de voz, por ejemplo un fonema, le corresponde un modelo acústico específico, es decir, tiene unas determinadas propiedades ya sea en el dominio del tiempo o la frecuencia, lo cual es hasta cierto punto correcto y por ello fue la primer forma de atacar el problema. Sin embargo, dada la alta variabilidad de estas propiedades ya que es un proceso con algunas características aleatorias, es difícil llegar a modelos exactos de cada fonema por lo que este intento no ha logrado funcionar de manera exitosa cuando es utilizado

de forma aislada.

2.1.2. Reconocimiento de patrones

Se intentan reconocer ciertos patrones presentes en la señal de voz sin la necesidad de identificar las características explícitas de cada segmento de voz. Normalmente se utilizan herramientas de estadística y probabilidad para modelar estos patrones. Este método requiere dos fases: el entrenamiento, en el cuál se aprenden estas características, y la clasificación de patrones, en la cual se realiza una comparación para determinar cual es el patrón más probable. Este método ha sido ampliamente utilizado por muchos de los sistemas de reconocimiento debido a factores como los siguientes:

- Es simple de implementar y se puede justificar y entender con fundamentos matemáticos y de teoría de la comunicación.
- Es robusto e invariante a distintas condiciones, por ejemplo: el vocabulario, el idioma o el tipo de segmento de voz.
- Ha probado tener resultados aceptables y provee un amplio rango de posibles mejoras.

2.1.3. Inteligencia Artificial

Esta propuesta podría considerarse como una combinación de las dos anteriores, pues intenta mecanizar la forma de reconocer de acuerdo a cómo un humano utiliza su razonamiento usando cierto conocimiento que posee y toma decisiones después de analizar diversas características. Para esto se pueden englobar diversas formas de conocimiento para realizar el reconocimiento, las cuales incluyen: acústico fonético, léxico, sintáctico, semántico y pragmático; que se describen a continuación.

Acústico: La evidencia de cuales de los segmentos definidos de voz son pronunciados

de acuerdo a las características espectrales o presencia o ausencia de otras características es usada para realizar el reconocimiento o complementarlo.

Léxico: El conocimiento de las palabras que existen en un idioma se usa para corregir los posibles errores en las hipótesis generadas.

Sintáctico: La combinación de palabras que forman frases gramaticalmente correctas.

Semántico: Validar frases que tienen sentido de acuerdo al contexto y con las frases anteriores .

Pragmático: Resolver ambigüedades de significado de acuerdo al uso del idioma.

Existen otros métodos, entre los que destacan las Redes Neuronales Artificiales que se pueden englobar en cualquiera de estos grupos o en otro acorde la forma en que se utilizan.

2.2. Procesamiento Digital de Voz

Independiente de los métodos utilizados, regularmente se requiere realizar algún procesamiento a la señal de voz al principio como se describe en Deller et al. [7] y Rabiner y Schafer [26], dentro de estos procesos es común el uso del preénfasis, el cual es un filtro que realza las componentes de mayor frecuencia de la voz, pues estas están atenuadas y poseen información valiosa de los segmentos de voz denominados sordos que son los sonidos correspondientes a las consonantes que no tienen un período. Otro proceso importante es la partición en bloques seguida de la aplicación de algún tipo de ventana, lo cual intenta disminuir la distorsión introducida al particionar.

Los siguientes procesos dependen de las técnicas empleadas, y pueden incluir: cálculo de correlación, potencia, cruces por cero y espectro de Fourier.

2.2.1. Modelado de la voz

El sistema de producción de voz de los seres humanos se puede modelar por medio de una señal de entrada que alimenta a un sistema o filtro y la señal de salida es la voz; las cuerdas vocales producen esa señal de excitación que pasa a través de distintos canales que hacen la función de filtros, los cuales son la garganta y el tracto vocal y nasal. La señal de excitación generalmente se modela como una combinación de un tren de pulsos a una determinada frecuencia y ruido gaussiano; y el tracto vocal generalmente se modela como un filtro IIR que cambia según el fonema pronunciado.

2.2.2. Representación de la señal de voz

En general para el proceso de reconocimiento de voz necesitamos un mecanismo de representar una señal de voz por medio de una serie de parámetros que la describan, ya que la misma forma de onda no permite su análisis directo, tratando generalmente de disminuir la cantidad de datos obtenidos, descartando lo no necesario o redundante pero teniendo una mejor representación de la información.

Predicción Lineal

Existen diversos métodos de representación de la voz, pero una forma muy utilizada se denomina predicción lineal Makhoul [23]. Esta técnica se basa en la suposición de que cada muestra de una señal digitalizada de voz se puede aproximar como la composición lineal de un número determinado de las muestras anteriores.

$$\hat{x}(n) = \sum_{k=1}^{p} a_k x(n-k) + e(n) \tag{2.1}$$

Pero debido a que existirá un error de aproximación $e(n)$.

$$x(n) = \sum_{k=1}^{p} a_k x(n-k) + e(n) \tag{2.2}$$

Si obtenemos la transformada de Fourier de esta ecuación y despejamos $X(z)$ tenemos lo siguiente:

$$X(z) = \frac{E(z)}{1 - \sum_{k=1}^{p} a_k z^{-k}} \tag{2.3}$$

Por lo que se puede interpretar la señal de voz $x(n)$ como un proceso de filtrado de una señal $e(n)$ a través de un filtro recursivo. Al analizar ésto podemos suponer que la señal $e(n)$ corresponde a la señal producida por las cuerdas vocales; la cual pasa por un filtro formado por el tracto vocal y nasal representado por estos coeficientes a los cuales se denominan coeficientes LPC. Al obtener estos coeficientes y después la señal de error se puede comprobar que ésta tiene la forma de ruido gaussiano cuando los segmentos de voz son del tipo sordos y de un tren de pulsos cuando corresponden a sonidos sonoros.

Entonces utilizando esta teoría podemos suponer que los coeficientes describen el tracto vocal en cada instante y éste a su vez está relacionado con los fonemas que se estén pronunciando; además están relacionados con el espectro de la voz; por esta razón se busca calcular estos coeficientes y un método de obtención de estos parámetros es el llamado minimización del error cuadrático medio: es decir buscar que el promedio del error al cuadrado (MSE) sea mínimo.

Para obtener estos coeficientes primero recordamos:

$$\hat{x}(n) = \sum_{k=1}^{p} a_k x(n-k) \tag{2.4}$$

Después calculamos el error de aproximación.

$$e(n) = x(n) - \hat{x}(n) \tag{2.5}$$

$$e(n) = x(n) - \sum_{k=1}^{p} a_k x(n-k) \tag{2.6}$$

Lo derivamos e igualamos a cero para obtener un mínimo.

$$e_{rms}^2 = E[x(n) - \sum_{k=1}^{p} a_k x(n-k)]^2 \tag{2.7}$$

$$\frac{\partial e(n)^2}{\partial a_i} = -2E(x_{n-i}[x(n) - \sum_{k=1}^{p} a_k x(n-k)]) = 0; i = 1, ..., p \tag{2.8}$$

$$R(i) - \sum_{k=1}^{p} a_k R(|k-i|) = 0 \tag{2.9}$$

Con lo que obtenemos un sistema de ecuaciones que podemos expresar matricialmente de la siguiente manera.

$$\begin{bmatrix} r(0) & r(1) & \dots & r(p-1) \\ \vdots & \vdots & \vdots & \vdots \\ r(p-1) & \dots & \dots & r(0) \end{bmatrix} \begin{bmatrix} a_1 \\ \vdots \\ a_p \end{bmatrix} = \begin{bmatrix} r(1) \\ \vdots \\ r(p) \end{bmatrix} \tag{2.10}$$

Por lo tanto necesitaríamos invertir la matriz obtenida para calcular los coeficientes a:

$$\begin{bmatrix} a_1 \\ \vdots \\ a_p \end{bmatrix} = \begin{bmatrix} r(1) \\ \vdots \\ r(p) \end{bmatrix} \begin{bmatrix} r(0) & \dots & r(p-1) \\ \vdots & \vdots & \vdots \\ r(p-1) & \dots & r(0) \end{bmatrix}^{-1} \tag{2.11}$$

Este sistema de ecuaciones se puede resolver con algoritmos eficientes que aprovechan la simetría; uno de estos es el llamado método Levinson Durvin Levinson [19] Durbin [8]. Se puede demostrar que el error es ortogonal a la señal de salida y a si mismo, lo que significa que está descorrelacionado por lo cual es ruido blanco o gaussiano y su espectro es plano.

Una vez teniendo estos coeficientes es útil calcular el error cuadrático medio y para simplificar esto podemos decir que el coeficiente $a_0 = 1$

$$e^2_{rms} = E[\sum_{k=0}^{p} a_k x(n-k)]^2 \tag{2.12}$$

$$e^2_{rms} = E[\sum_{k=0}^{p} a_k x(n-k) \sum_{j=0}^{p} a_j x(n-j)] \tag{2.13}$$

$$e^2_{rms} = \sum_{k=0}^{p} a_k \sum_{j=0}^{p} a_j E[x(n-k)x(n-j)] \tag{2.14}$$

$$e^2_{rms} = \sum_{k=0}^{p} a_k \sum_{j=0}^{p} a_j R[k-j] \tag{2.15}$$

Para los coeficientes óptimos la expresión anterior se puede representar de la forma:

$$e^2_{rms} = r_a(0) + 2\sum_{k=1}^{p} R(k) r_a(k) \tag{2.16}$$

donde:

$$r_a(k) = \sum_{n=0}^{p-k} a_n a_{n+k} \tag{2.17}$$

Esta expresión se denomina distancia Itakura-Saito y se puede demostrar que es una manera de comparar el espectro de las señales Rabiner y Juang [24] .

Esta última expresión es muy útil pues permite comparar la señal con sus coeficientes LPC, pero además es posible comparar otra señal con estos coeficientes LPC y así tener una medida de similaridad entre señales y además si ya tenemos los coeficientes LPC de una señal patrón entonces no necesitamos los coeficientes de la otra; simplemente se tiene que calcular la correlación y después aplicar la formula anterior.

Análisis cepstral

Los coeficientes LPC fueron unos descriptores muy utilizados en los primeros reconocedores de voz basados en la técnica de reconocimiento de patrones, pero se han ideado otros descriptores y entre los que han probado dar mejores resultados están los denominados coeficientes cepstrales Tokhura [31], Juang et al. [16]. Para obtener éstos, se parte igualmente de la idea que la señal de voz puede ser modelada como un proceso de filtrado de la señal proveniente de los pulmones y las cuerdas vocales que pasa a través del tracto vocal:

$$v(t) = h(t) * x(t) \tag{2.18}$$

Lo que en el espacio de Fourier queda reducido ala multiplicación de dos señales.

$$V(f) = H(f)X(f) \tag{2.19}$$

Si aplicamos el logaritmo a la transformada de Fourier de la señal de voz, lo cual

se asemeja al proceso que realiza el oído, podemos transformar la multiplicación en suma y así separar la señal referente a la señal de excitación y la referente al tracto vocal $h(t)$, ésta tendrá componentes frecuenciales mayores que la derivada de $x(t)$. por lo que en el espectro estarán separadas.

$$ln(V(f)) = ln(H(f)) + ln(X(f)) \tag{2.20}$$

La transformada inversa de Fourier de esta señal se llama cepstrum y de esta se derivan los coeficientes cepstrales.

Los coeficientes cepstrales c_n son los que conforman la potencia de este espectro.

$$ln(V(f))^2 = \sum_{n=-\infty}^{\infty} C_n e^{-j2\pi fn} \tag{2.21}$$

Estos coeficientes se pueden derivar de los coeficientes LPC o del espectro de Fourier, existen distintas variaciones de éstos; las más conocidas son: los mismos coeficientes multiplicados por una ventana que realza los coeficientes intermedios y atenúa los extremos, ya que estos tienen información más relevante. Otra forma de coeficientes son los llamados coeficientes mel cepstrales (MFCC), los cuales son resultado de cambiar la escala del espectro de frecuencia a una escala logarítmica llamada mel, la cual trata de imitar la forma de percepción del oído humano.

Los coeficientes cepstrales se pueden obtener de los coeficientes LPC de la siguiente forma:

$$C_0 = r(0) \tag{2.22}$$

$$C_1 = -a_1 \tag{2.23}$$

$$C_i = -a_i - \sum_{k=1}^{i-1} \frac{i-k}{i} C_{i-k} a_k; i = 2, .., M \tag{2.24}$$

$$C_i = -\sum_{k=1}^{M} \frac{i-k}{i} C_{i-k} a_k; i = M+1, ... \tag{2.25}$$

Como se explica en Huang et al. [13] para calcular los coeficientes mel cepstrales se siguen los siguientes pasos:

1. División de la señal en tramas.
2. Calcular la potencia del espectro de Fourier.
3. Calcular el logaritmo.
4. Aplicar un banco de filtros triangulares distribuidos en la escala mel.
5. Obtener la transformada coseno discreta.
6. Eliminar los coeficientes de orden alto.

Una vez calculados los coeficientes cepstrales normalmente se utilizan los coeficientes delta, es decir las razones de cambio de primer y hasta segundo orden de estos coeficientes, utilizando para ello coeficientes consecutivos.

Existen algunas otras formas de representar una señal de voz, como el espectro de Fourier en si, los coeficientes denominados PLP Jamaati et al. [15] y las wavelets Gargour et al. [10]; pero los anteriores son los más utilizados.

2.2.3. Cuantización vectorial (CV)

Al tener codificada la señal de voz hemos reducido la cantidad de datos a analizar pero podemos reducirlo aun más al aplicar el proceso de cuantización vectorial

Lloyd [22], el cual consiste en realizar un mapeo de un espacio vectorial de varias dimensiones según las que se estén trabajando al espacio de los números naturales, es decir a una región del espacio vectorial R^n le corresponde un subindice, por lo que cualquier vector que se encuentre dentro de dicha región será equivalente. Esto reduce drásticamente la cantidad de información almacenada, sin embargo, conlleva el peligro de que se pierde información y el grado de desempeño igualmente consistirá en cuan redundante es la información eliminada. Lo ideal sería que un grupo de centroides le correspondiera siempre un fonema con todas sus variaciones pero no suele ser así.

Para realizar este proceso primero se requiere generar códigos de las distintas regiones, y a cada región le corresponde un determinado centroide representativo y a cada nuevo vector se le asigna una región dependiendo que centroide se encuentre más cercano pero teniendo en cuenta que esta cercanía es una medida de distorsión que puede variar según sea el caso analizado. Además existen diversas técnicas de cuantizacion vectorial de las que destacan k medias y el algoritmo LBG Linde et al. [20].

2.3. Reconocimiento de palabras aisladas utilizando CV

Este método consiste en generar un conjunto de centroides (codebook) correspondientes a cada palabra distinta que se busca reconocer mediante una serie de repeticiones de la misma; y al reconocer una nueva palabra se calcula la distancia acumulada de la distancia mínima a un centroide de cada codebook, y la palabra correspondiente a dicho codebook con esta distancia mínima será la palabra reconocida. Ver figura 2.1

$$D_{Palabraj} = \sum_{i=1}^{Nbloques} min[d(bloque_i, centride_k^{palabraj})] \tag{2.26}$$

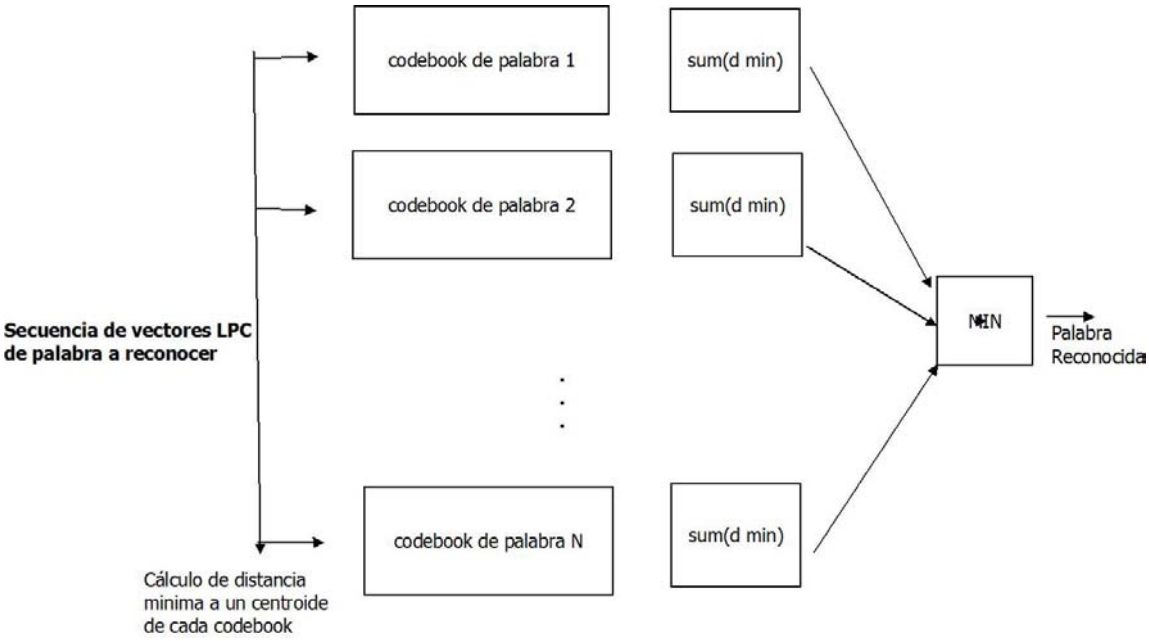

Figura 2.1: Reconocimiento de palabras aisladas con CV

$$\text{Palabra Reconocida} = argmin[D_{Palabraj}] \quad 1 \leq j \leq Npalabras \tag{2.27}$$

Este método posee muchas variantes en cuanto a los coeficientes utilizados, el número de centroides y mejoras al considerar métodos estadísticos para corregir errores y falsos positivos. Además se puede realizar este proceso particionando la señal para obtener un conjunto de centroides para cada sección de la palabra en vez de cada palabra.

Capítulo 3

Modelos Ocultos de Markov

Los modelos Ocultos de Markov son una herramienta matemática que fue propuesta y estudiada a finales de los años 60 y principios de los 70 por Baum y sus colegas: Baum y Petrie [3], Baum y Eagon [2], Baum et al. [4] y utilizada inicialmente en el campo de la biología; pero que en los 80 fue ampliamente utilizada para el reconocimiento automático de voz, ya sea reconocimiento de palabras aisladas, clave o en forma continua; esto debido a su gran versatilidad y su riqueza matemática. Estos modelos y su aplicación en el reconocimiento de voz fueron exhaustivamente estudiados por Rabiner [25].

3.1. Descripción de Modelos ocultos de Markov

Un modelo oculto de Markov es un modelo probabilístico que contiene dos variables aleatorias, las cuales son: el estado S y la observación O, generalmente el estado S está oculto y solo se puede tener acceso a la observación O. El modelo se puede ver como un conjunto de estados conectados entre si con determinadas probabilidades de transición entre estados y con una función de probabilidad de observación en cada estado. Esto se puede observar en la Figura 3.1

Para describir un modelo λ utilizamos un vector π que describe la probabilidad

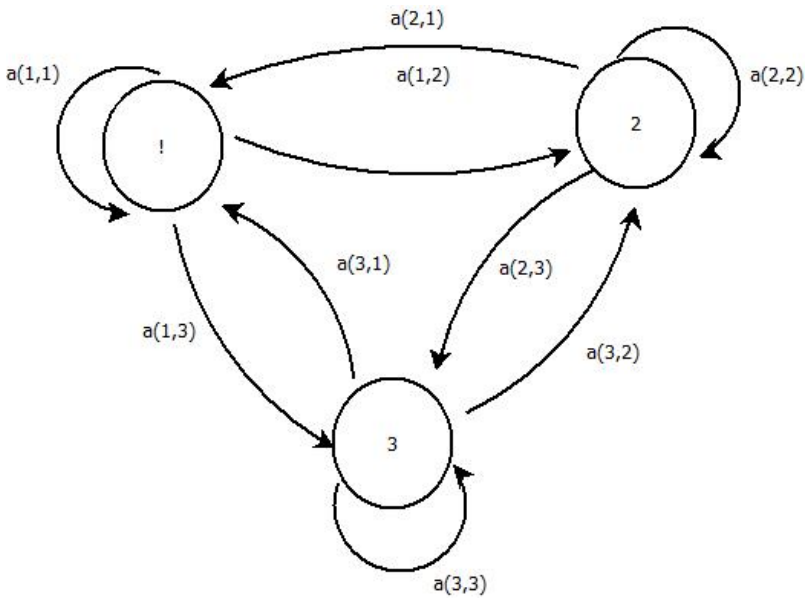

Figura 3.1: Gráfico de un Modelo Oculto de Markov

de estados iniciales, una matriz A de probabilidad de transición entre estados y otra matriz de emisión B de probabilidad de tener una determinada observación en cada estado. Generalmente se utiliza la N para indicar el número de estados y M para el número posible de observaciones diferentes; por lo que el vector π será de N elementos, la matriz A de NxN y la matriz B de NxM. Un Modelo Oculto de Markov es una herramienta matemática muy versátil y existen diversas formas de utilizarla; este modelo puede representar un segmento de voz: una palabra, un fonema, una silaba o incluso todo un lenguaje y un estado de éste puede representar una fracción de dicho segmento; por lo que una sucesión de estados puede generar una sucesión de observaciones relacionadas con este segmento.

Teniendo en cuenta lo anterior existen tres problemas básicos de interés referentes al uso de los Modelos Ocultos de Markov. Los cuales son los siguientes.

3.1.1. El problema de evaluación

Este problema consiste en que al tener un modelo $\lambda(\pi, A, B)$ y una secuencia de observaciones $O = (o_1, o_2, ...o_T)$; cómo obtener de forma eficiente la probabilidad de que este modelo haya generado la secuencia de observaciones $P(O/\lambda)$. Como fue mencionado, un modelo puede representar un segmento de voz y si tenemos una secuencia de observaciones podemos calcular la probabilidad que este modelo haya generado esta secuencia de observaciones para decidir si esta secuencia de observaciones corresponde al segmento de voz que representa el modelo, sin preocuparnos por la secuencia de estados que haya tenido.

Este problema se puede resolver intuitivamente calculando la probabilidad de que cada secuencia de estados posible en el modelo haya generado la secuencia de observaciones y sumando cada una de éstas; sin embargo existen muchas combinaciones de secuencias N^T, y calcular la probabilidad de cada una requiere T multiplicaciones, donde T es el número de observaciones; esto resulta en una gran cantidad de cálculos, por lo que se ha diseñado un algoritmo denominado forward Rabiner [25], el cual calcula esta misma probabilidad de una manera recursiva realizando menos cálculos. Este método se describe a continuación:

Se basa en la idea de resolver el problema considerando primero que solo tenemos una observación y realizando el cálculo correspondiente, resolviendo para cada estado y sumando, para después considerar que solo hay dos observaciones y resolviendo utilizando el resultado previo y así sucesivamente hasta completar toda la secuencia.

1. Iniciación $$\alpha_1(i) = \pi_i bi(o_i) \quad 1 \leq i \leq N \tag{3.1}$$
2. Iteración $$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(O_{t+1}) \quad 1 \leq t \leq T-1 \quad 1 \leq j \leq N \tag{3.2}$$
3. Término $$P(O/\lambda) = \sum_{i=1}^{N} \alpha_T(i) \tag{3.3}$$

Este método requiere TN^2 operaciones que es mucho menor que TN^T para valores de T grandes. Existen algunas consideraciones en la implementación de este algoritmo

como que las cantidades se van haciendo muy pequeñas, por lo que es conveniente que se aplique un factor de escala al ir realizando los cálculos para no salirse del rango que se pueda representar en una computadora.

De una manera similar tenemos al algoritmo backward Rabiner [25] que trabaja de forma semejante pero iniciando con la ultima muestra hacia la primera.

1. Iniciación $$\beta_T(i) = 1 \quad 1 \leq i \leq N \tag{3.4}$$
2. Iteración $$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j) \quad t = T-1, ..., 1 \quad 1 \leq i \leq N \tag{3.5}$$

3.1.2. El problema de decodificación

Dado un modelo $\lambda(\pi, A, B)$ y una secuencia de observaciones $O = (o_1, o_2, ...o_T)$, se debe decidir la secuencia de estados $Q = (q_1, q_2, ...q_T)$ más probable en producir esta secuencia de observaciones. Esto sirve cuando un modelo representa todo un conjunto de segmentos de voz y necesitamos saber cuales de esos segmentos están presentes en la secuencia de observaciones.

Igualmente se puede resolver intuitivamente calculando la probabilidad de cada secuencia posible y seleccionando la mayor, pero igualmente requiere TN^T cálculos, así que se opta generalmente por utilizar el denominado algoritmo Viterbi Viterbi [32], Forney [9], que de forma recursiva va obteniendo las rutas parciales más probables a lo largo del camino, descartando las que nunca podrían ser y al final calcula la ruta más probable. El algoritmo es el siguiente:

1. Se calcula la probabilidad de que la primera observación haya sido generada por cada estado del modelo utilizando el vector π y la matriz B.
2. Se calcula la probabilidad para cada estado que haya habido una transición a este y la primera observación haya sido generada por cada estado predecesor utilizando los resultados anteriores y la matriz A y se selecciona los estados predecesores más probables de cada estado, para después obtener la probabilidad

de que la segunda observación haya sido generada por cada estado utilizando la matriz B y el resultado anterior.

3. Se registra el índice del estado predecesor más probable de cada estado en la muestra dos.

4. Se repiten los dos pasos anteriores sobre cada observación.

5. Al final tendremos la probabilidad de que cada estado sea el último de la secuencia y se selecciona el estado más probable y con éste utilizamos el registro que llevamos del estado predecesor más probable a cada estado en cada posición de la secuencia para obtener toda la secuencia de estados.

1. Iniciación	$\delta_1(i) = \pi_i bi(o_i) \quad 1 \leq i \leq N$	(3.6)
	$\psi_1(i) = 0$	(3.7)
2. Recursión	$\delta_t(j) = max[\delta_{t-1}(i)a_{ij}]b_j(o_t) \quad 2 \leq t \leq T \quad 1 \leq j \leq N$	(3.8)
	$\psi_t(j) = armax[\delta_{t-1}(i)a_{ij}] \quad 2 \leq t \leq T \quad 1 \leq j \leq N$	(3.9)
3. Término	$p* = max[\delta_T(i)]$	(3.10)
	$q*_T = argmax[\delta_T(i)]$	(3.11)
4. Ruta	$q*t = \psi_{t+1}(q*t+1), \quad t = T-1, T-2, ..., 1$	(3.12)

Estas operaciones igualmente nos dan probabilidades muy pequeñas y como las operaciones son únicamente multiplicaciones podemos aplicar el logaritmo a las operaciones y únicamente realizar sumas para prevenir que se desborde el sistema.

3.1.3. El problema de aprendizaje

Obviamente se necesitara tener un modelo para utilizar los métodos anteriores, el aprendizaje o ajuste del modelo consiste en que dada una secuencia de observaciones y un modelo inicial, como ajustar este modelo para que la probabilidad de que este

haya generado la secuencia de observaciones sea lo más alta posible, es decir obtener los valores de sus matrices. Este problema no tiene una solución cerrada, además de que no existe una forma única de escoger el número de estados del modelo o el número de elementos de la matriz B. Para resolver este problema, generalmente utilizan algoritmos iterativos; estos utilizan los parámetros iniciales del modelo y la secuencia de observaciones para recalcular unos nuevos parámetros más cercanos a los deseados y volver a calcularlos hasta llegar a un máximo local por un cierto límite impuesto por alguna probabilidad calculada o por el cambio entre los parámetros. Uno de estos es el algoritmo Baum Welch Rabiner [25] que se describe a continuación.

Primero definimos la variable $\xi_t(i,j)$ como la probabilidad de estar en el estado i en el tiempo t y pasar al estado j en el tiempo t+1, dada una secuencia y el modelo. Se calcula como sigue: utilizando las variables α y β de los algoritmos forward y backward.

$$\xi_t(i,j) = \frac{\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)}{\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)} \tag{3.13}$$

Tenemos la variable $\gamma_t(i)$ como la probabilidad de estar en el estado i en el tiempo j dado el modelo y la secuencia.

$$\gamma_t(i) = \sum_{j=1}^{N}\xi_t(i,j) \tag{3.14}$$

Utilizando estas variables podemos deducir las fórmulas de estimación:

$$\pi(i) = \gamma_1(i) \tag{3.15}$$

$$a_{ij} = \frac{\sum_{t=1}^{T-1}\xi_t(i,j)}{\sum_{t=1}^{T-1}\gamma_t(i)} \tag{3.16}$$

$$b_j(k) = \frac{\sum_{t=1,o(t)=k}^{T} \gamma_t(i)}{\sum_{t=1}^{T} \gamma_t(i)} \tag{3.17}$$

Utilizando estas fórmulas y un modelo inicial se calculan estos parámetros y se vuelve al primer paso hasta que se cumpla un cierto requisito de concurrencia.

El otro algoritmo utiliza el algoritmo Viterbi para obtener una secuencia de estados y con esta re-calcula los parámetros del modelo hasta que la probabilidad aumente a un grado razonable. El algoritmo se detalla a continuación:

1. Se presenta un modelo inicial y la secuencia de observaciones a la que se debe ajustar el modelo.
2. Se calcula el algoritmo Viterbi que nos proporciona la secuencia de estados más probable.
3. Con estas dos secuencias alineadas(estados y observaciones) se re-calculan los parámetros del modelo como sigue.

$$\pi_i = \frac{\text{Número de veces que se comenzó en el estado i}}{\text{Número de secuencias}} \tag{3.18}$$

$$A(i,j) = \frac{\text{Número de veces que se transitó del estado i al estado j}}{\text{Número de transiciones}} \tag{3.19}$$

$$B(i,j) = \frac{\text{Número de veces que se obtuvo la observación i en el estado j}}{\text{Número de observaciones}} \tag{3.20}$$

4. Como en el algoritmo anterior, se vuelve al primer paso hasta que se cumpla un cierto requisito de concurrencia.

Como se mencionó, estos algoritmos llegan aun máximo local, el cual depende del modelo inicial, por lo que al variar éstos podemos obtener resultados diferentes, así que

para obtener los mejores resultados se deberían de proporcionar unas condiciones iniciales cercanas a la solución real.

3.2. Modelos ocultos de Markov con funciones de probabilidad continuas

Como se explica en Taylor [28] en este tipo de modelos la matriz de emisión B ya no representa la probabilidad discreta de cada símbolo sino que representa una función de densidad de probabilidad continua, que generalmente está representada por una combinación lineal de funciones de probabilidad continua, estas funciones generalmente son gaussianas y podemos representar esta combinación como sigue:

$$b_j(o) = \sum_{k=1}^{M} c_{jk} N(o, \mu_{jk}, \sigma_{jk}) \tag{3.21}$$

Ya que la función de probabilidad gaussiana es vectorial, la representamos como sigue:

$$N(o, \mu, \Sigma) = \frac{1}{\sqrt{(2\pi)^N |\Sigma|}} e^{-\frac{1}{2}(o-\mu)\Sigma^{-1}(o-\mu)} \tag{3.22}$$

y

$$\sum_{k=1}^{M} c_{jk} = 1, 1 \leq j \leq N \tag{3.23}$$

$$c_{jk} \geq 0, 1 \leq k \leq M \tag{3.24}$$

Donde M es el número de gausianas por estado; así que la matriz B es de NxM y representa los pesos c_{jk} ; además necesitamos NxM vectores de medias de y NxN

matrices de covarianza; aunque normalmente se busca que los componentes de la función gausiana estén descorrelacionados para facilitar los cálculos y así tener vectores de varianzas.

Las fórmulas de reestimación equivalente del algoritmo Baum Welch de los parámetros de un modelo con función de probabilidad continua son:

$$c_{jk} = \frac{\sum_{i=1}^{T} \gamma_t(j,k)}{\sum_{i=1}^{T} \sum_{k=1}^{M} \gamma_t(j,k)} \tag{3.25}$$

$$\mu_{jk} = \frac{\sum_{i=1}^{T} \gamma_t(j,k).O_t}{\sum_{i=1}^{T} \gamma_t(j,k)} \tag{3.26}$$

$$\Sigma_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k).(O_t - \mu_{jk})(O_t - \mu_{jk})'}{\sum_{t=1}^{T} \gamma_t(j,k)} \tag{3.27}$$

Donde $\gamma_t(j,k)$ es la probabilidad de estar en el estado j en el instante t tomando solo en cuenta la k ésima componente; y se calcula como sigue:

$$\gamma_t(j,k) = \left[\frac{\alpha_t(j)\beta_t(j)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)}\right] \left[\frac{c_{jk} N(O_t, \mu_{jk}, \Sigma jk)}{\sum_{m=1}^{M} c_{jm} N(O_t, \mu_{jm}, \Sigma jm)}\right] \tag{3.28}$$

También podemos utilizar un algoritmo basado en el entrenamiento Viterbi descrito anteriormente que fue denominado entrenamiento k medias, el cual es el siguiente:

1. Podemos partir de un modelo inicial y aplicar el algoritmo Viterbi con las muestras de entrenamiento para tener una alineación de observaciones y estados o suponer una alineación uniforme a lo largo de todos los estados.

2. Con esta alineación tenemos un grupo de observaciones por cada estado: entonces por cada estado se realiza la cuantización k medias donde el número de centroides corresponde a M el número de gausianas.

3. Los centroides obtenidos serán los vectores de medias, y con estas podemos calcular la matriz de covarianza utilizando todas las observaciones que pertenecen a ese centroide y con el número de observaciones pertenecientes a cada centroide divididas entre el total de observaciones del estado en cuestión podemos obtener los valores c_{jk}.

4. La matriz A se calcula de la misma manera que en el entrenamiento Viterbi.

5. Aplicamos el algoritmo Viterbi y regresamos al paso 2 hasta que el algoritmo converja.

Cabe señalar que originalmente el algoritmo utiliza la cuantizacion vectorial k medias pero se podría utilizar cualquier método o incluso solo utilizarse para obtener el modelo inicial a utilizarse con el algoritmo anterior

3.3. Reconocimiento de palabras aisladas utilizando Modelos Ocultos de Markov

Para reconocer palabras aisladas utilizando Modelos Ocultos de Markov podemos utilizar la cuantización vectorial para general un conjunto de todas las palabras código utilizando todas las palabras a reconocer y con este se tiene un grupo de secuencias de estados por cada palabra que servirá para entrenar un modelo por cada palabra y para reconocer una de estas palabras se calcula la probabilidad de que cada modelo haya generado dicha secuencia de estados, ya sea con el algoritmo forward o Viterbi y se escoge el que resulte con mayor probabilidad. Igualmente se pueden utilizar funciones de probabilidad continua en vez de utilizar cuantizacion vectorial. Generalmente se utiliza una combinación lineal de funciones gausianas. Esto de puede observar en la Figura 3.2.

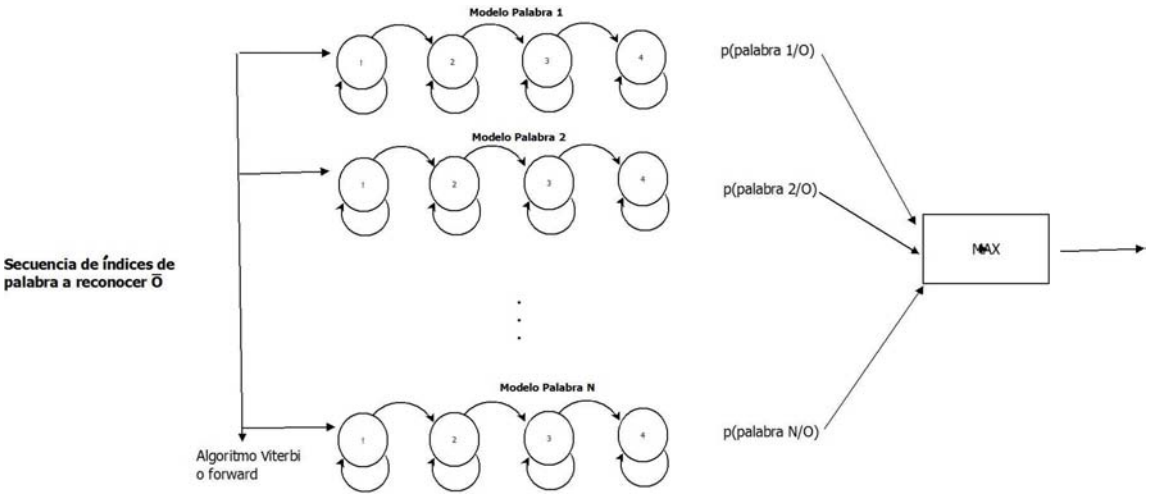

Figura 3.2: Reconocimiento de palabras aisladas con HMM

3.4. Reconocimiento de palabras clave en voz continua

El reconocimiento de voz continua es más complejo, pues aunque no nos parezca, la forma de onda de las palabras varían cuando están inmersas en una trama continua, además de que son afectadas por factores como la entonación.

El reconocimiento de palabras clave busca detectar si dentro de una trama continua de voz se encuentra alguna palabra perteneciente a un vocabulario definido. Este se puede conseguir mediante el empleo de un método indirecto que consiste en realizar el reconocimiento continuo de voz para reconocer cada palabra pronunciada y después verificar si se encuentran dichas palabras clave. Un sistema de reconocimiento de voz continuo necesita tener entrenados los modelos de todas las posibles oraciones que se puedan pronunciar por lo que normalmente posee modelos de segmentos de voz pequeños como por ejemplo fonemas los cuales se concatenan para formar todas las palabras y oraciones posibles. Este método necesita utilizar sistemas de reconocimiento de voz de vocabulario extenso que son muy complejas y requieren una base de datos muy extensa para el entrenamiento pero que además no permitiría incluir nuevas palabras de forma fácil.

La forma directa de detección de palabras clave consiste en buscar específicamente solo dichas palabras dentro de la trama de voz; y la forma más común de hacerlo consiste en obtener los modelos ocultos de Markov de cada palabra que deseamos detectar de forma aislada para después generar un modelo que incluya las palabras de interés , este modelo tiene estados específicos pertenecientes a cada palabra, además estados que corresponden a palabras ajenas, al silencio o ruido. Ya teniendo éste, se puede utilizar el algoritmo de Viterbi para encontrar los estados que corresponden a las observaciones obtenidas e identificar las palabras que se dijeron.

Cuando el vocabulario es amplio, generalmente es necesario incluir métodos de inteligencia artificial como en Carmona [5] para mejorar la tasa de reconocimiento. Se puede incluir información gramatical para indicar qué oraciones son correctas, información semántica para indicar qué oraciones tienen sentido e información contextual para saber qué frases son las más probables en cada situación.

Capítulo 4

Detección de palabras clave

4.1. Estado del arte

Los sistemas de reconocimiento de voz han sido desarrollados desde hace ya décadas; y aunque en ese tiempo han ido mejorando su desempeño, todavía están alejados de la capacidad que tiene una persona para reconocer el habla. Además la mayoría de estos sistemas siguen trabajando con el mismo principio de funcionamiento basado en el uso de Modelos Ocultos de Markov como es mencionado en Huang y Deng [14] y algunos se han inclinado por el uso de Redes Neuronales Artificiales Kirschning [17] Tebelskis [29]. Estos sistemas se diferencian en la forma en que implementan estos métodos; pues como pudimos ver existe una gran variedad de maneras en que se puede implementar, y algunos procesos extras que se pueden realizar, utilizando aparentemente los mismos métodos; por ejemplo la forma de codificar la voz con sus posibles variaciones en si misma, así como la opción de codificar de manera continua o discreta. Además el uso en un determinado lenguaje podría significar tomar a consideración algunos parámetros. Muchos de los sistemas más recientes han optado por codificar los segmentos de voz por medio de los coeficientes mel cepstrales (MFCC) y se utilizan de manera continua a través de combinación de funciones de probabilidad gaussianas.

4.2. Clasificación de los sistemas de reconocimiento de voz

Los sistemas de reconocimiento de voz se pueden clasificar de acuerdo a la manera en que se requieran utilizar y podemos nombrar tres grupos.

Reconocimiento de palabras aisladas. Reconocimiento de voz continua. Detección de palabras clave.

Los sistemas de reconocimiento de palabras aisladas son los más sencillos, sobre todo si son de vocabulario reducido, pero su estudio sirve como punto de partida para los sistemas más complejos.

4.2.1. Reconocimiento de voz continua

Los reconocedores de voz continua son sistemas que deben reconocer todas las palabras que se pronuncian de una manera ininterrumpida o la mayoría de estas; requieren una gran cantidad de trabajo en su desarrollo; pero además una gran cantidad de muestras de voz para su entrenamiento y también la incorporación de información de tipo lingüística, por ejemplo un diccionario de las palabras así como las estructuras gramaticales que pueden formar. Los sistemas que utilizan Modelos Ocultos de Markov entrenan un modelo o modelos por cada distinto segmento de voz que utilizan como base y concatenan estos para reconocer las palabras, como normalmente deben reconocer un vocabulario amplio necesitan utilizar segmentos de voz pequeños como difonemas, trifonemas , sílabas o fonemas aunque entre más pequeño sea el segmento menor sera la capacidad de reconocimiento, por ejemplo en el caso de fonemas solo se tendrían que entrenar alrededor de 40 para el idioma inglés o de 25 para el español, pero su tasa de reconocimiento sería más baja, otra opción es generar variaciones de los segmentos de acuerdo para el contexto en el que estén, como vimos los segmentos de voz están influenciados por el contexto en que son pronunciados, por ejemplo los segmentos vecinos o si son acentuados o no, como ya dijimos estos sistemas requieren de un extenso trabajo y vocabulario de

entrenamiento.

En general poseen el mismo principio de funcionamiento y la ecuación fundamental del reconocimiento de voz es la siguiente:

$$Wmax = argmaxP(W/X) = argmax\frac{P(W)P(X/W)}{P(X)} \tag{4.1}$$

Como P(x) es invariante a cada secuencia de palabras W podemos reducirla a:

$$Wmax = argmaxP(W/X) = argmaxP(W)P(X/W) \tag{4.2}$$

Donde X representa el vector de observaciones de la señal de voz, Wmax es la secuencia de palabras más probable P(W) se obtiene por medio del modelo del lenguaje y P(X/W) por medio de los modelos acústicos.

4.2.2. Detección de palabras clave

En la detección de palabras clave los sistemas deben reconocer ciertas palabras de interés dentro de una trama de audio que puede contener muchas otras palabras llamadas ajenas, utilizan generalmente Modelos Ocultos de Markov y consideran un modelo para cada palabra clave y un modelo que representa las palabras ajenas y algunas veces otro que represente el ruido de fondo y se concatenan normalmente poniendo estados de transición entre ellos. La principal diferencia de éstos es que normalmente el vocabulario que manejan es más reducido y no se interesan por las palabras ajenas, por lo que cualquier palabra fuera del vocabulario será descartada. Una forma de lograr esto es utilizar un sistema de reconocimiento de voz continua de vocabulario extenso, el cual produce la serie de palabras más probables de haber sido pronunciadas, y entonces en estas buscar dichas palabras clave. Esta manera produce buenos resultados, pero tiene la desventaja de tener más costo de computación y una gran cantidad de muestras de entrenamiento, por lo que normalmente se opta por un sistema basado en la primera idea.

Estos sistemas tienen la ventaja que no necesitan de un corpus de voz tan extenso pues normalmente buscan detectar un número reducido de palabras y en teoría se puede mejorar la tasa de reconocimiento. Este tipo de sistemas generalmente consideran reconocer una sola palabra clave embebida en una trama de palabras ajenas, por lo tanto se pueden representar por un modelo como en la figura. Por ejemplo el sistema propuesto por Wilpon et al. [33] fue ideado para dirigir llamadas telefónicas y busca detectar una palabra clave que indica alguna opción, para esto tiene un modelo para cada palabra clave y lo concatena con modelos de palabras ajenas, realizando diversas pruebas teniendo modelos separados de palabras ajenas más comunes o incluyéndolas en un mismo modelo. Por su parte el modelo propuesto por Leow et al. [18] consta de modelos de las palabras clave hechos por la concatenación de los modelos de sus fonemas por separado, además de incluir el modelo de palabras ajenas, el cual detecta si hay palabras de interés para después realizar un proceso de verificación a partir de los resultados que dio la evaluación de Markov y paralelamente utilizando redes neuronales. En contraste, el sistema propuesto en esta tesis busca la detección de palabras clave en una trama de voz que pueda contener varias palabras clave, y debe detectarlas todas.

Han existido diferentes propuestas de cómo modelar las palabras ajenas, por ejemplo Rohlicek et al. [27] sugiere modelar las palabras fuera del vocabulario como segmentos de las palabras clave. Por el contrario en [1] no se modelan las palabras ajenas, sino que modelan las palabras clave simplemente por una sucesión de índices del codebook y se buscan dentro de la trama de voz y ya teniendo ciertas candidatas a palabras clave se realiza el reconocimiento utilizando modelos Ocultos de Markov como si fueran palabras aisladas; por su parte en Tejedor y Colás [30] estudian diferentes formas de modelar las palabras ajenas, desde utilizar 49 fonemas diferentes, hasta tener solo 3 modelos que engloban a todos los fonemas; en Lleida et al. [21] modelan las palabras ajenas como una combinación de clases de silabas y en Cuayáuitl y Serridge [6] se modelan utilizando fonemas, silabas y palabras enteras.

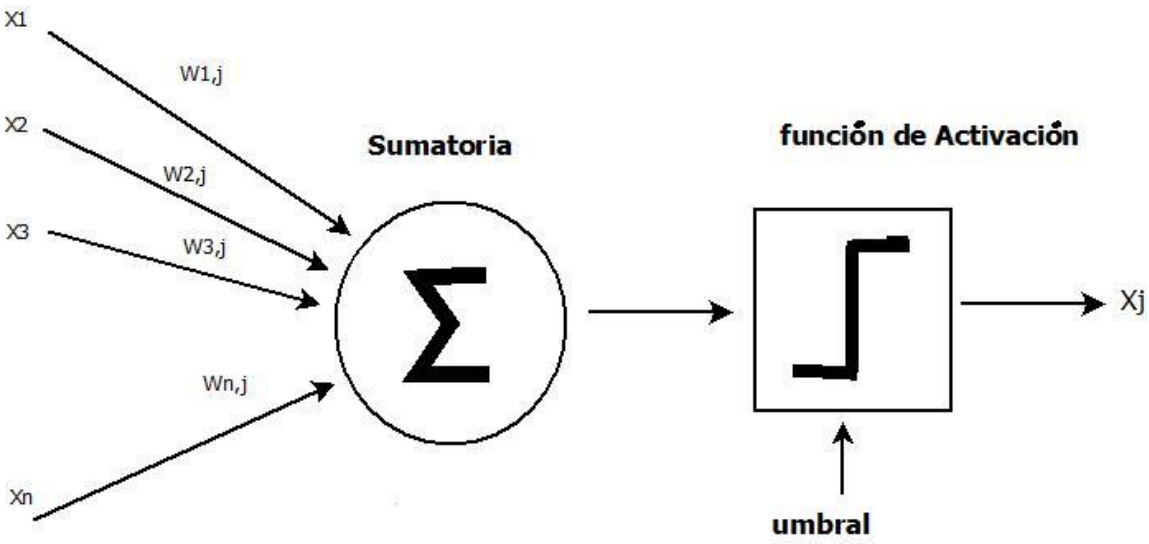

Figura 4.1: Neurona Artificial

4.3. Redes Neuronales Artificiales

Las Redes Neuronales Artificiales también han sido muy utilizadas en el reconocimiento de voz, ésto es motivado por el hecho que los seres humanos son capaces de realizar este proceso de manera aparentemente sencilla y todo debido a la capacidad del cerebro humano. Las Redes Neuronales Artificiales modelan de una manera simplificada el funcionamiento de las neuronas biológicas, las cuales básicamente poseen muchas conexiones entre sí llamadas sinapsis y cada neurona realiza un proceso simple; este proceso consta de recibir y transmitir pulsos de acuerdo a los que recibe, que aunque este proceso no es tan veloz, se logran altas velocidades porque todos los procesos se realizan paralelamente Haykin [12], esta es la forma en que se realiza cualquier proceso de pensamiento en los humanos incluyendo el reconocimiento de la voz.

Podemos representar una neurona con el siguiente diagrama 4.1

La forma de funcionamiento se pude describir matemáticamente al representar cada neurona como un elemento que suma ponderadamente todas sus entradas y después se aplica una función no lineal de activación con un cierto umbral; que se

puede representar de diferentes maneras pero al final lo podemos simplificar a que bajo el umbral es cero y sobre este es 1.

$$y_j = \sum_{i=1}^{n} w_{i,j} x_i \tag{4.3}$$

$$x_j = f(y_j) \tag{4.4}$$

Entonces al conectar varias de estas neuronas se construye una red neuronal como la de la figura: 4.2. Una red debe cumplir que dadas ciertas entradas y una asignación de pesos específica; se produzcan las salidas requeridas, las cuales describan estas entradas. Por lo tanto estas conexiones y los pesos asignados a cada una de ellas son lo que determina la respuesta a los estímulos y por ende representa el conocimiento de esta red.

De acuerdo a la forma en que se conectan las neuronas se pueden formar distintas topologías, y las más comunes son: no estructurada, por capas, recurrente y modular. Cada topología es más útil para una determinada tarea; por ejemplo: la no estructurada sirve más para completar patrones; la que tiene distintas capas sirve para reconocer patrones y la recurrente para reconocer secuencias de patrones.

Teóricamente se podría resolver cualquier problema de clasificación siempre y cuando se utilice la topología apropiada, con la asignación de pesos precisa y que los parámetros de entrada cumplan cierto patrón sin necesidad de que se tenga que analizar éste. A continuación se listan las principales ventajas de las Redes Neuronales Artificiales:

- Se pueden implementar con un alto grado de paralelismo por lo que es posible implementarlas eficientemente en dispositivos de este tipo.
- Presentan un alto grado de robustez al ruido y fallos en la estructura debido a que el conocimiento está disperso en toda la red.
- Los pesos de la red pueden ser actualizados en tiempo real para para mejorar

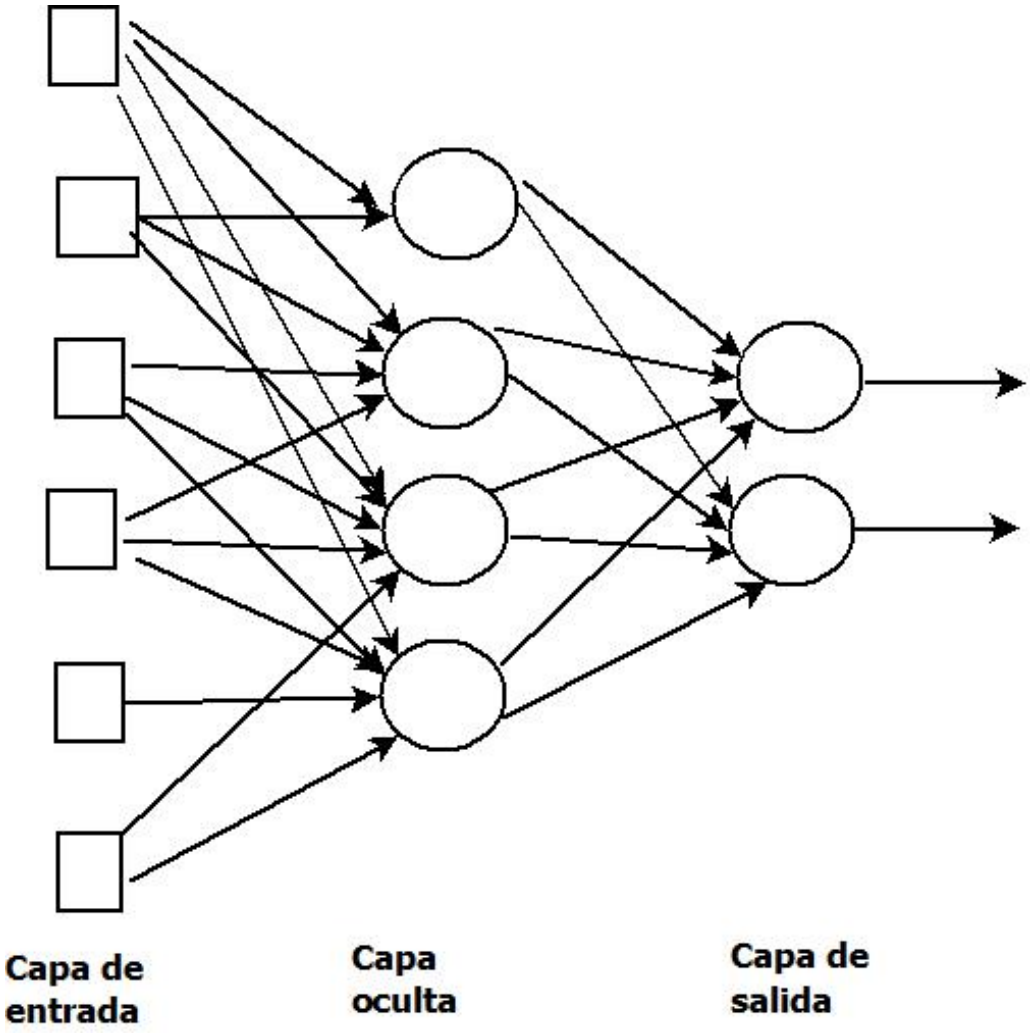

Figura 4.2: Red Neuronal Artificial

el rendimiento.

- Pueden aproximar cualquier sistema no lineal.

4.3.1. Aprendizaje

El aprendizaje consiste en ajustar los pesos de las neuronas de modo que a determinadas entradas nos resulten las salidas deseadas. Cabe señalar que este proceso necesita tener una topología establecida con una cantidad de neuronas igualmente establecidas, la cual se escoge por medio de la experiencia propia o de alguien más o por prueba y error. Para esto no existe una solución cerrada y generalmente se realiza (al igual que con los Modelos Ocultos de Markov) con algoritmos recursivos que mediante la asignación de unos pesos iniciales arbitrarios se busca dirigirlos hacia la solución idónea. Los métodos de entrenamiento más comunes son: feedback propagation Haykin [12] y algoritmos genéticos.

Feedback propagation

Este método comienza utilizando una asignación inicial aleatoria de pesos para obtener los resultados de la red dadas las muestras de entrenamiento como entradas y mediante el uso de una fórmula de reestimación se ajustan los pesos basándose en los resultados obtenidos y la cercanía a los resultados deseados, éste ajuste dirige los pesos hacia un máximo local por lo que se realiza este proceso recursivamente hasta que el error obtenido no cambie.

Algoritmos genéticos

Este método igualmente parte de pesos aleatorios pero en este caso utiliza muchos candidatos, analiza el desempeño de cada uno y escoge a los mejores, es decir los que se acerquen más al resultado deseado, se realizan pequeños cambios o mutaciones a éstos y se recombinan entre si para obtener nuevos candidatos y se realiza el mismo

procedimiento hasta que se encuentra la combinación de pesos que tenga el mejor desempeño.

4.3.2. Reconocimiento de voz utilizando Redes Neuronales Artificiales

Existen infinidad de maneras que podemos utilizar las Redes Neuronales en el reconocimiento de voz: cada quien puede idear una manera; pero en general podemos utilizar una red en capas y a las entradas podremos tener algún tipo de representación de la voz como los coeficientes LPC o cepstrales o los indices del cuantizador; a la salida tendremos la palabra que se dijo o algún segmento de voz correspondiente; además en la entrada podemos tener toda la serie de bloques de voz que obtenemos o solamente uno o unos pocos, pero en este caso necesitaríamos usar una red recurrente Graves et al. [11] la cual tiene realimentación de sus salidas o capas ocultas a sus entradas o capas ocultas anteriores lo que da como resultado una red que necesita algoritmos más complejos de entrenamiento.

Estas redes también se pueden utilizar en combinación con otros procesos para mejorar la tasa de reconocimiento, por ejemplo usar los resultados de las probabilidades obtenidas para verificar si tenemos un acierto o falsa alarma. Igualmente se pueden utilizar en el análisis gramático o semántico de las frases.

Capítulo 5

Desarrollo del sistema de reconocimiento de voz

5.1. Motivación del trabajo

El objetivo de esta tesis es el desarrollo de un sistema de reconocimiento de voz en español que será utilizado por un robot de servicio autónomo que realizará ciertas tareas caseras; por lo que tendrá que detectar secuencias de palabras clave que pertenecen a órdenes especificas. Esto como parte de un proyecto del laboratorio de bio-Robótica del departamento de Procesamiento Digital de Señales que pertenece al posgrado de Ingeniería de la Universidad Nacional Autónoma de México. En este laboratorio se han desarrollado diferentes robots, enfocándose en los de tipo humanoide debido a que se participa en diversos torneos de robótica destacando el Torneo Mexicano de Robótica y la competencia internacional llamada RoboCup, una de las categorías en la que participa es la denominada "athome" en la cual se simula un ambiente habitacional donde el robot debe superar un conjunto de pruebas relacionadas con el servicio domestico o el apoyo a los adultos mayores. Para esto el robot, además de tener una estructura mecánica que le permita moverse y manipular objetos posee diversos módulos programados; por ejemplo de visión computacional, navegación, planificación de acciones, síntesis y reconocimiento de voz.

Se pretende desarrollar un sistema de reconocimiento de voz desde cero que permita manipular libremente cada bloque así como las palabras y frases que maneje debido a que la mayoría de los sistemas necesitan una gran cantidad de muestras de entrenamiento, están adecuados para el idioma inglés y ninguno es perfecto; además que permita libremente la integración con el robot y la implementación de mejoras futuras. El sistema esta planeado para reconocer solo las palabras clave, ya que de esta manera puede funcionar con un vocabulario reducido pero que reconozca las órdenes posibles que pueda ejecutar el robot sin necesidad de reconocer cada palabra pronunciada y así poder procesar distintas formas de dar una misma orden. También tendrá un bloque de reconocimiento de palabras aisladas que le permita al robot preguntar algún dato y el usuario se lo proporcione con una sola palabra.

5.2. Desarrollo

Como ya se ha mencionado, un sistema de reconocimiento de voz se puede describir como una serie de bloques que comprenden tanto procesamiento digital de señales como técnicas de reconocimiento de patrones basadas en estadística y probabilidad; y la estructura general se puede describir a grandes rasgos en la Figura 5.1. Claramente podemos diferenciar dos procesos: entrenamiento y reconocimiento; los cuales comparten algunos pasos.

A continuación se especifica como se realizó cada una de los pasos listados, los cuales fueron programados en el lenguaje de programación C Sharp debido a que permite la comunicación con el dispositivo kinect que posee los micrófonos y además se pueden programar los algoritmos de manera eficiente y permite la comunicación con los módulos adicionales que se encuentran en el robot.

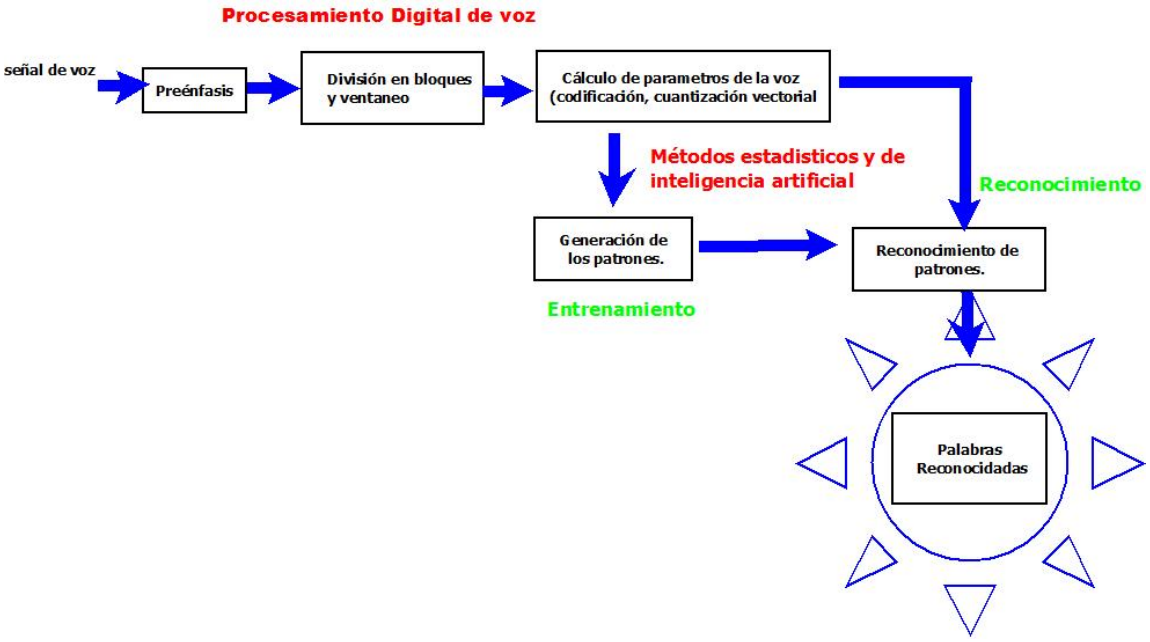

Figura 5.1: Diagrama de bloques del reconocimiento

5.3. Obtención de la señal de voz

La señal de voz se obtiene mediante el uso del dispositivo kinect, el cual realiza un muestreo de 16 KHZ y posee un arreglo de cuatro micrófonos que por medio del uso del controlador del kinect se puede programar el filtrado espacial del sonido con el propósito de captar el audio proveniente de una determinada dirección, filtrar las demás fuentes de audio y obtener una señal más limpia.

5.4. Procesamiento digital de la señal de voz

Para obtener los descriptores de la señal de voz se realizaron los pasos que se describen en la figura 5.2.

En primer lugar se realiza un proceso de filtrado paso altas de primer orden, el cual sirve para resaltar las frecuencias relativamente altas de la voz que corresponden

Figura 5.2: Diagrama de bloques del procesamiento digital

a los denominados sonidos sordos o aperiódicos, los cuales tienen un nivel de energía bajo. Este proceso se denomina pre énfasis y lo podemos representar con la función de transferencia:

$$H(Z) = 1 - aZ^{-1}, \quad 0.9 < a < 1.0 \tag{5.1}$$

Y también con la ecuación en diferencias:

$$y(n) = x(n) - ax(n-1) \tag{5.2}$$

En segundo lugar se realizó una división en bloques equiespaciados de voz y se aplica cierto traslape entre bloques para garantizar la continuidad; este traslape normalmente es dos tercios del tamaño del bloque. En esta prueba se utilizaron distintas combinaciones de longitud de bloques y traslape y las que dieron buenos resultados fueron tramas de 30 mili segundos de duración obtenidas cada 10 mili segundos, como la frecuencia de muestreo es de 16 khz, se obtienen bloques de 480 muestras cada 160 muestras.

En tercer lugar se multiplicó cada bloque por una ventana de Hamming con el fin de disminuir la distorsión causada a la señal, que se puede analizar en el dominio de la frecuencia. La ventana de Hamming se modela con la siguiente ecuación:

$$y(n) = x(n)w(n) \tag{5.3}$$

$$w(n) = 0.54 - 0.46cos(\frac{2\pi n}{N-1}), \quad 0 \leq n \leq N-1 \tag{5.4}$$

5.5. Codificación de la señal

Ya que tenemos los bloques de la señal de voz se procede a obtener los descriptores de éstos.

5.5.1. Obtención de coeficientes LPC

Para obtener los coeficientes LPC a partir de estos bloques se obtuvo un vector de correlación cruzada de p+1 términos, donde p es el orden del vector de coeficientes LPC. En esta prueba se probaron diferentes órdenes y al final se escogió p=15. Por lo tanto se obtendrían vectores de 16 términos: de r(0) a r(15).

$$r(m) = \sum_{n=0}^{N-1-m} x(n+m), m = 0, 1, ..., p \tag{5.5}$$

Con este vector de correlación se construyó la matriz de correlación y se resolvió el sistema de ecuaciones siguiente.

$$\begin{bmatrix} r(0) & r(1) & \dots & r(p-1) \\ \vdots & \vdots & \vdots & \vdots \\ r(p-1) & \dots & \dots & r(0) \end{bmatrix} \begin{bmatrix} a_1 \\ \vdots \\ a_p \end{bmatrix} = \begin{bmatrix} r(1) \\ \vdots \\ r(p) \end{bmatrix} \tag{5.6}$$

Pero en este trabajo este sistema se resolvió por medio de un algoritmo llamado Levinson-Durbin Levinson [19] Durbin [8], el cual aprovecha las simetrías de la matriz y consta de los siguientes pasos:

$$E^0 = r(0) \tag{5.7}$$

$$k_i = \frac{r(i) - \sum_{j=1}^{i-1} \alpha_j^{i-1} r(|i-j|)}{E^(i-1)}, 1 \leq i \leq p \tag{5.8}$$

$$\alpha_i^{(i)} = k_i \tag{5.9}$$

$$\alpha_j^{(i)} = \alpha_j^{(i-1)} - k_i \alpha_{i-j}^{(i-1)} \tag{5.10}$$

$$E^(i) = (1 - k_i^2) E^(i-1) \tag{5.11}$$

5.5.2. Obtención de coeficientes cepstrales

Para la obtención de los coeficientes cepstrales se utilizó un algoritmo recursivo a partir de los coeficientes LPC, el cual se describe a continuación:

$$C_0 = r(0) \tag{5.12}$$

$$C_1 = a_1 \tag{5.13}$$

$$C_m = a_m + \sum_{k=1}^{m-1} \frac{m-k}{m} C_{i-k} a_k, \quad m = 2, ..., P \tag{5.14}$$

$$C_m = \sum_{k=1}^{P} \frac{m-k}{m} C_{i-k} a_k, \quad m = P+1, ..., q \tag{5.15}$$

Donde p es el orden de los coeficientes LPC y q el de los coeficientes cepstrales que normalmente es 1.5p.

Estos coeficientes se multiplicaron por una ventana que resalta los coeficientes intermedios y atenúa los de los extremos, ya que los primeros son más significativos

que los últimos:

$$W_c(m) = 1 + \frac{Q}{2} sin(\frac{\pi m}{Q}), \quad 1 \leq m \leq Q \tag{5.16}$$

$$c^1(m) = c(m) * W_c(m) \tag{5.17}$$

A partir de los coeficientes cepstrales se obtuvieron los coeficientes delta cepstrales que son la derivada de los anteriores y describen como van cambiando los coeficientes cepstrales a lo largo del segmento de voz, para obtenerlos se aproxima la derivada utilizando los coeficientes contiguos.

$$\Delta c_t(m) = G \sum_{k=-K}^{K} k c_{i-k}(m) \quad 1 \leq m \leq Q \tag{5.18}$$

Donde k es la mitad de la cantidad de muestras que se utilizan y G es un factor que se utiliza para que ambos grupos de coeficientes sean de magnitudes similares.

5.5.3. Obtención de coeficientes mel cepstrales

Como se explica en Huang et al. [13], los pasos para el cálculo de los coeficientes mel cepstrales fueron los siguientes.

1. División de la señal en tramas de 1024 muestras y 160 muestras de traslape
2. Se aplicó una ventana de Hamming a cada trama.
3. Se calculó la potencia del espectro de fourier con la FFT.
4. Se obtuvo el logaritmo del espectro.
5. Se aplico un banco de filtros triangulares distribuidos en la escala mel de 40 regiones.

6. Se obtuvo la transformada coseno discreta.

7. Se eliminaron los coeficientes de orden alto quedándose con solo 13.

8. Se obtuvieron los coeficientes delta y aceleración de los anteriores.

La escala mel esta motivada por el hecho de que la percepción del sonido en los humanos se realiza de manera logarítmica en la amplitud y en el dominio del la frecuencia.

Las ecuaciones que relacionan la escala mel y lineal de las frecuencias son:

$$B(f) = 1125ln\left(1 + \frac{f}{700}\right)[mels] \tag{5.19}$$

y

$$B^{-1}(m) = 700exp\left(\frac{m}{1125}\right) - 700[HZ] \tag{5.20}$$

Estas ecuaciones resultan en una equivalencia a los 1000 HZ.

5.5.4. Cuantización vectorial

Una vez obtenidos los descriptores, se realiza el proceso de cuantizacion que se refiere a realizar un mapeo de estos que se encuentran en R^n al espacio de los naturales. Esto se logra definiendo un set de centroides llamado codebook que representan todos los vectores presentes en R^n y cualquier vector será representado por alguno de estos vectores y para escoger cual se determina una medida de distancia y el centroide más cercano al vector será el que lo represente, entonces en lugar de tener una infinita cantidad de vectores posibles solo se trabajará con un set finito de índices que corresponden a cada centroide.

5.5.5. Obtención del codebook

Existen distintos métodos para obtener este set de centroides; cada uno con ventajas y desventajas; pero el que se utilizó es el llamado algoritmo LBG Linde et al. [20], el cual consta de los siguientes pasos:

1. Se obtiene el promedio de todos los vectores que se van a analizar.
2. Este promedio se varía un poco y se obtienen dos vectores, los cuales serán los primeros centroides.
3. Se calcula la distancia de cada vector a estos centroides y se agrupan en regiones basados en la distancia mínima.
4. Se realiza un promedio en cada grupo para actualizar el valor del cenroide.
5. Se vuelve al paso 3 hasta que los centroides no cambien o el cambio sea mínimo.
6. Se varían ligeramente los centroides para tener el doble de los que se tenían.
7. Se vuelve al paso 3 hasta que se tenga el número de centroides establecido.

5.6. Implementación de los Modelos Ocultos de Markov para el reconocimiento de palabras aisladas

Se programó la implementación de un Modelo Oculto de Markov por cada palabra aislada que se busca reconocer y este modelo es del tipo no regresivo, es decir solo es posible transitar de un estado al siguiente y al mismo, para esto se establece esta restricción en el modelo inicial. Esto se puede observar en la Figura 5.3 . Después se entrenó por medio del entrenamiento Viterbi. El número de estados de cada modelo que dio mejores tasas de reconocimiento fue 6 y cada estado tiene una función de probabilidad discreta de 128 posibilidades que corresponde a cada índice

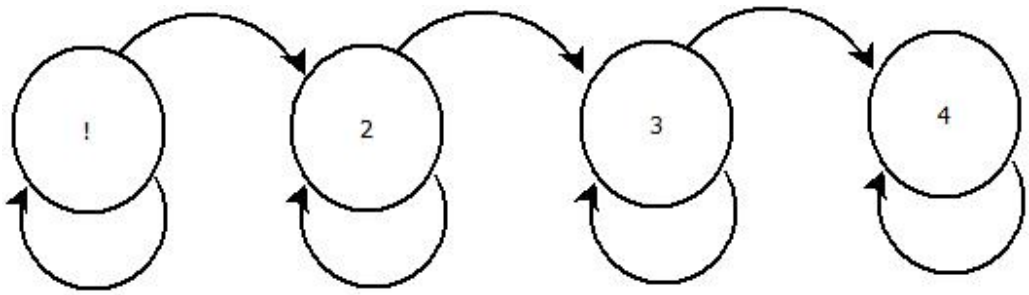

Figura 5.3: Gráfico de un Modelo Oculto de Markov no regresivo

del codebook descrito anteriormente, el modelo inicial se escogió asignándole a la matriz A la forma no regresiva y con una alineación uniforme de las observaciones con los estados se obtuvo la matriz B. A continuación se listan los pasos seguidos en las dos etapas requeridas.

5.6.1. Entrenamiento

1. Se realizó la grabación de las distintas palabras a entrenar, cada una 10 veces.

2. Se realizó el procesamiento digital de cada grabación hasta obtener una serie de vectores de 16 coeficientes LPC por cada una de ellas.

3. Se obtuvieron 25 coeficientes cepstrales a partir de los coeficientes LPC y también se les agregó los coeficientes delta cepstrales.

4. Utilizando la totalidad de los coeficientes cepstrales y delta se realizó la cuantización vectorial obteniendo un codebook de 128 palabras.

5. Teniendo el codebook se reemplazó la sucesión de vectores delta-cepstrales por los índices correspondientes del codebook.

6. Por medio de estos índices se entrena un Modelo Oculto de Markov por cada palabra.

5.6.2. Reconocimiento

1. Se realizó la grabación de 10 repeticiones adicionales de las palabras a reconocer.
2. Se obtuvieron los coeficientes LPC de cada palabra grabada.
3. Se obtuvieron 25 coeficientes cepstrales a partir de los coeficientes LPC y también se les agrego los coeficientes delta cepstrales.
4. Se le asignó a cada vector un índice correspondiente acorde al codebook obtenido anteriormente.
5. Con esta sucesión de índices se utiliza el algoritmo forward y Viterbi para determinar la probabilidad de que cada modelo haya generado dicha secuencia y se selecciona la mayor, se establece un umbral de probabilidad por estadística y si la probabilidad no alcanza el umbral se clasifica como palabra ajena.

5.7. Implementación de los Modelos Ocultos de Markov para el reconocimiento de palabras clave

5.7.1. Entrenamiento

Teniendo los modelos para cada una de las palabras se construyó un modelo general que engloba todas las palabras, además de incluir estados adicionales para las palabras ajenas y el ruido y las transiciones, el cual se puede observar en la figura 5.4

Sin embargo al probar el reconocimiento en distintas frases que contenían estas palabras clave, la tasa de reconocimiento decrecía conforme aumentaba el número de palabras clave a detectar; así que se modificaron algunos detalles del método anterior y el algoritmo resultante fue el siguiente:

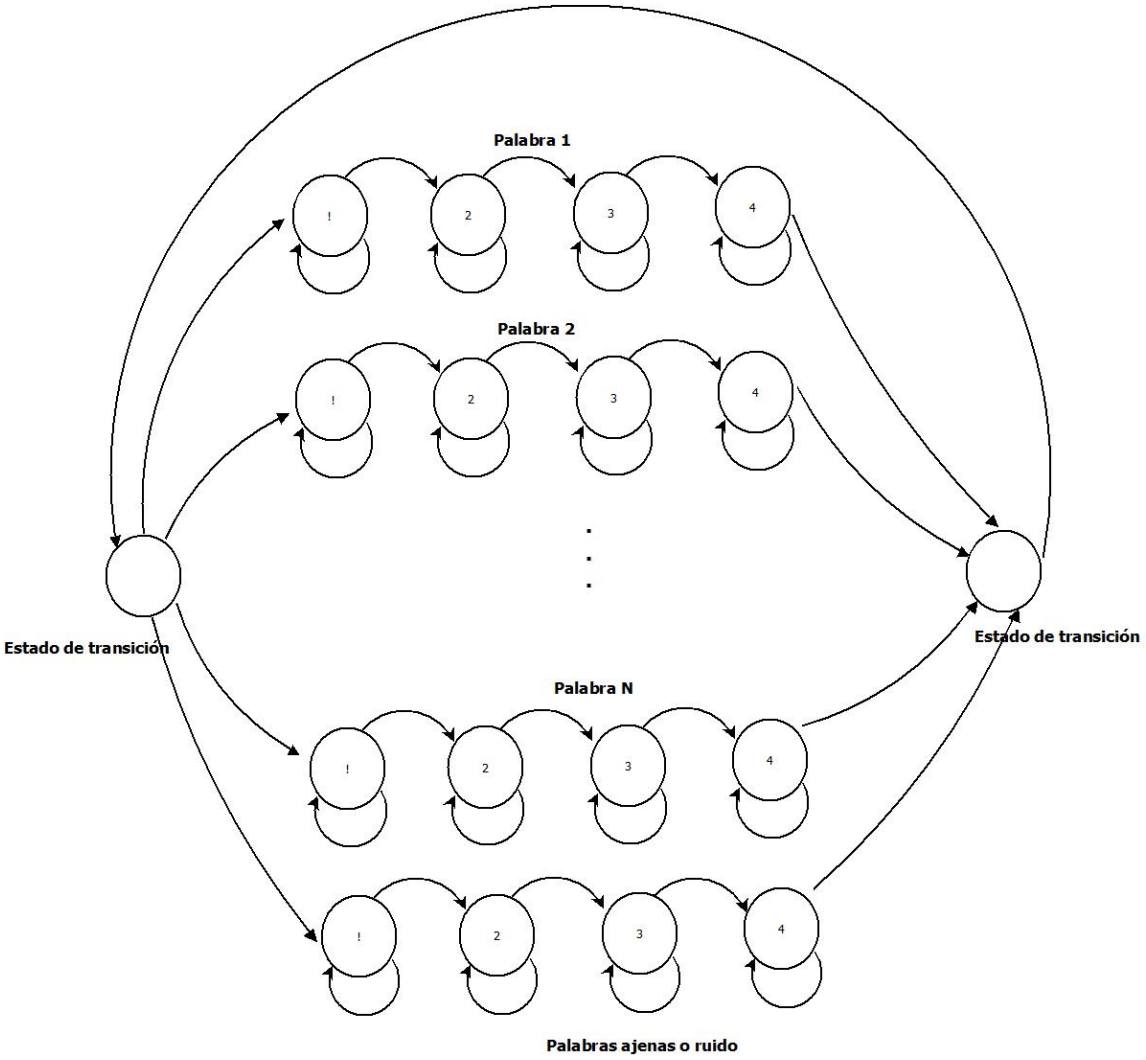

Figura 5.4: Modelo Oculto de Markov General

1. Se obtuvo la grabación de 10 repeticiones de 10 distintas frases de entrenamiento y otras 10 repeticiones de 10 frases para el reconocimiento, teniendo al final 200 frases de una sola persona.

2. Las 100 frases de entrenamiento se etiquetaron, registrando por cada una las palabras clave, ajenas o silencio que contenía y sus limites; para realizar esto último se escribió un programa que permitía observar la forma de onda y escuchar fragmentos de la señal.

3. Se realizaron los mismos procesos de procesamiento de señales descritos anteriormente para la obtención de una serie de coeficientes cepstrales concatenados con sus coeficientes delta.

4. Se realizó el proceso de cuantización vectorial utilizando los coeficientes cepstrales de la totalidad de las frases para generar un codebook de 128 símbolos y una sucesión de índices por cada frase.

5. Se realizo el entrenamiento utilizando los índices obtenidos de la cuantización vectorial para generar un modelo por cada palabra clave diferente que se tuviera, las palabras ajenas se englobaron en un solo modelo y el ruido o fondo se uso para generar otro modelo.

6. Utilizando las etiquetas de las frases se obtuvo una gramática en la que se especificaba que palabras pueden suceder a cada una y con qué probabilidad y se armó un Modelo Oculto de Markov usando todos los modelos obtenidos y estableciendo las transiciones entre modelos por medio de la gramática

5.7.2. Reconocimiento

1. Se realizaron los mismos procesos de procesamiento de señales descritos anteriormente para la obtención de una serie de coeficientes cepstrales concatenados con sus coeficientes delta.

2. Se obtuvo una serie de índices por cada frase a reconocer usando el codebook obtenido en el entrenamiento.

3. Con la sucesión de índices obtenida y el Modelo Oculto de Markov entrenado se usó el algoritmo Viterbi para obtener una secuencia de estados.

4. Se realizó un proceso de interpretar la secuencia de estados tomando en cuenta a qué palabras corresponde cada estado, cuántas veces transitó por cada estado y las probabilidades resultantes para determinar que palabras clave se encontraban en cada frase.

Se realizó un proceso similar al anterior utilizando los coeficientes mel cepstrales y otro usando Modelos Ocultos de Markov con funciones de probabilidad continua. Después de obtener la secuencia de palabras probable se aplicó un algoritmo para limpiar la cadena de texto obtenida, pues algunas veces presentaba palabras adicionales, también completaba frases cuando faltaba unicamente una palabra y no había duda de cual era, esto utilizando la gramática de frases posibles.

Capítulo 6

Pruebas y resultados

En primer lugar se probó el reconocimiento de palabras aisladas con 1000 grabaciones constituidas de 20 repeticiones de 50 palabras diferentes, las cuales se pueden observar en la tabla 6.1 10 usadas para el entrenamiento y 10 para probar el reconocimiento; obteniendo los siguientes resultados:

87.6 % de palabras reconocidas correctamente utilizando cuantización vectorial, un codebook de 64 símbolos, 25 coeficientes cepstrales con sus 25 delta cepstrales y un Modelo Oculto de Markov de 5 estados.

88.4 % de palabras reconocidas correctamente utilizando cuantización vectorial, un codebook de 128 símbolos, 25 coeficientes cepstrales con sus 25 delta cepstrales y un Modelo Oculto de Markov de 5 estados.

89.6 % de palabras reconocidas correctamente utilizando cuantización vectorial, un codebook de 128 símbolos, 25 coeficientes cepstrales con sus 25 delta cepstrales y un Modelo Oculto de Markov de 6 estados.

93 % de reconocimiento correcto utilizando un Modelo Oculto de Markov de 5 estados y funciones de probabilidad continua con una mezcla de 8 gaussianas de 25 coeficientes cepstrales con sus 25 delta cepstrales.

91 % de reconocimiento correcto utilizando un Modelo Oculto de Markov de 6

casa	perro	robot	uno	dos
tres	cuatro	cinco	seis	siete
ocho	nueve	diez	once	doce
trece	catorce	quince	si	no
ve	hola	mesa	sala	señor
silla	tele	trae	tú	yo
adiós	comida	cama	lleva	mamá
puerta	abuelo	amigo	basura	cierra
cocina	cuarto	diario	entra	hijo
niño	sal	sígueme	sillón	ventana

Figura 6.1: Lista de palabras

estados y funciones de probabilidad continua con una mezcla de 8 gaussianas de 25 coeficientes cepstrales con sus 25 delta cepstrales.

Estos resultandos fueron idénticos o prácticamente idénticos usando el algoritmo Viterbi o forward. Utilizando una corrección a partir de la estadística se pudieron corregir algunos errores o por lo menos detectarlos, ésto se logró al establecer un rango en la probabilidad resultante de los algoritmos utilizados para que el reconocimiento sea aceptado como válido, entre más riguroso es este rango impuesto se eliminan la mayoría de falsos positivos pero también se empiezan a invalidar verdaderos positivos y viceversa. Los algoritmos se corrieron en una máquina con un procesador de 1.65 GHZ y memoria RAM de 4 GB y el tiempo de respuesta fue alrededor de un tercio de segundo usando cuantización vectorial y casi un segundo usando funciones de probabilidad continua. También se probo el reconocimiento con grupos de diez a doce palabras y el reconocimiento llegó a casi el 100 % en algunos casos. Ésto es útil ya que en varios casos el robot preguntará algo que esta limitado aun rango pequeño de respuestas.

Con estos modelos se probó el reconocimiento de voz continuo concatenando los modelos de cada palabra usando cuantización vectorial y los 25 coeficientes cepstrales con sus coeficientes delta; se utilizó el algoritmo Viterbi para obtener la secuencia

de estados más probable y en base a ésta las palabras que contienen la frase, y se reentrenó el modelo utilizando la alineación obtenida y se obtuvo una exactitud de 80 % reconociendo 2 diferentes frases utilizando un modelo de 7 palabras. Sin embargo, al aumentar el número de palabras y frases al doble solo se obtuvo un reconocimiento de 40 %.

Para mejorar el reconocimiento se optó por generar los modelos y la cuantización vectorial utilizando directamente las frases. Se realizó el reconocimiento de frases completas utilizando 10 frases diferentes con 10 repeticiones para prueba y 10 para entrenamiento y un modelo para cada uno de 15 estados y un codebook de 128 índices obteniendo 99 % de reconocimiento de las mismas frases de entrenamiento, pero 70 % de reconocimiento en las frases de prueba.

Por último se realizó el reconocimiento por medio de un solo Modelo Oculto de Markov que contenía todas las palabras clave y concatenando las palabras clave, ajenas y el ruido de fondo por medio de una gramática simple generada a partir de las frases grabadas; las frases utilizadas fueron las siguientes:

1. robot ve a la cocina
2. robot ve a la sala
3. robot trae el diario
4. robot trae la comida
5. robot trae una bebida
6. robot trae la medicina
7. robot abre la puerta
8. robot cierra la puerta
9. robot limpia la mesa
10. robot sígueme

Donde las palabras clave son los sustantivos y verbos y las palabras ajenas son los conectores como los artículos y las preposiciones, en esta prueba se usaron 15 palabras clave, un modelo que englobaba las palabras ajenas y otro el ruido y silencio de fondo.

Después de obtener una secuencia de palabras claves posible se realiza un proceso de corrección de errores completando las frases cuando falta una sola palabra y esta palabra es la única opción de formar una frase coherente o eliminado palabras sobrantes cuando al hacerlo obtenemos igualmente frases coherentes. Este proceso se puede ampliar al tener un contexto establecido y reducir la cantidad de frases posibles y al preguntar alguna palabra faltante para completar la frase. También se usa la probabilidad resultante para validar el resultado.

Se realizaron tres pruebas usando un HMM de 128 índices y 85 estados, 5 por cada una de las 15 palabras clave, mas 5 de las palabras ajenas y 5 del ruido de fondo: la primera, un HMM con cuantizacion vectorial y coeficientes cepstrales obteniendo un reconocimiento de 74 % de las frases y 84 % al aplicar el algoritmo de corrección. La segunda, un HMM con cuantizacion vectorial y coeficientes mel cepstrales obteniendo un reconocimiento de 30 % de las frases y 55 % al aplicar el algoritmo de corrección. Y la tercera, un HMM con funciones de probabilidad continua y coeficientes cepstrales obteniendo un reconocimiento de 50 % de las frases y 80 % al aplicar el algoritmo de corrección.

Los resultados de la detección de las palabras clave y falsas alarmas de la prueba con mejores resultados se pueden ver en la tabla 6.2

palabra	Ocurrencias	Detectadas	Falsas alarmas	Porcentaje de detección
robot	100	100	1	100%
ve	20	12	2	60%
cocina	10	7		70%
sala	10	9	1	90%
trae	40	38		95%
diario	10	8		80%
comida	10	10	3	100%
bebida	10	7	2	70%
medicina	10	10	8	100%
abre	10	3	1	30%
cierra	10	9		90%
puerta	20	19	7	95%
limpia	10	9	6	90%
mesa	10	10	6	100%
sígueme	10	10	7	100%

Figura 6.2: Resultado de detección de palabras clave.

Capítulo 7

Conclusiones

7.1. Conclusiones

En esta tesis se estudiaron los fundamentos del reconocimiento de voz y su aplicación a la detección de palabras clave para la realización de un sistema de reconocimiento de voz que permite la interpretación de comandos utilizados para un robot de servicio autónomo que debe realizar tareas en un ambiente habitacional.

Para esto se utilizó como base los Modelos Ocultos de Markov los cuales son un modelo matemático que tiene infinidad de maneras de utilizarse, por lo que se buscó la implementación que mostrara los mejores resultados. Los algoritmos se programaron en el lenguaje C Sharp ya que permite la comunicación con el dispositivo kinect que posee los micrófonos utilizados para implementar un filtro direccional, además de permitir la programación de los algoritmos de forma relativamente sencilla sin perder velocidad de proceso y permite la comunicación con los módulos adicionales del robot.

Con los parámetros utilizados los mejores resultados se obtuvieron al utilizar los coeficientes cepstrales junto con sus coeficientes delta cepstrales, en el reconocimiento de palabras aisladas se obtuvo una tasa mayor de reconocimiento usando Modelos Ocultos de Markov con funciones de probabilidad continua y en la detección de

palabras clave fue mejor utilizar cuantización vectorial. Es posible corregir o por lo menos detectar una parte de los errores por medio de aplicar métodos de estadística y probabilidad y también se puede aumentar el tamaño del vocabulario y de las muestras de entrenamiento.

7.2. Trabajo futuro

El siguiente paso del proyecto sería la expansión del vocabulario para poder reconocer un mayor número de frases; el uso de un mayor número de muestras de entrenamiento para hacerlo más robusto; así como incluir diferentes tipos de voz.

Un paso adicional es la integración con el robot y analizar la interacción con otros módulos; también se pueden utilizar técnicas más avanzadas de Inteligencia Artificial que utilice la información del contexto en que se encuentra el robot que mejoren la tasa de reconocimiento, se podrían hacer algunos pasos más flexibles y permitir que el robot pregunte al usuario cuando tenga la incertidumbre de si un comando fue interpretado correctamente para que éste lo corrobore o niegue, para corregir un mayor número de errores e inclusive reentrenar los métodos a lo largo del tiempo.

Utilizando gran parte del proyecto es posible utilizar otras técnicas para codificar la señal de voz y descubrir si alguna presenta buenos resultados, además de utilizar nuevas formas de cuantización vectorial o de entrenamiento de los modelos.

Por otro lado se pueden utilizar algunos de los algoritmos programados para aplicarlos a la identificación de personas por medio del habla, que al igual que el reconocimiento de voz, es un tema bastante complejo pero utiliza muchos de los conceptos vistos en este trabajo como: representación de la señal de voz, cuantización vectorial y Modelos Ocultos de Markov. Además se podría implementar para reconocer otro tipo sonidos como el toque de la puerta, el ladrido de un perro, un grito o el llanto de un bebe, lo que sería útil en un robot de servicio; pero necesitaríamos representar la señal de otro modo, por ejemplo la transformada de Fourier.

Apéndice A

Código de algunas funciones y algoritmos en el lenguaje de programación C sharp

A.1. Entrenamiento Viterbi de un Modelo Oculto de Markov

```
public double LearnViterbi(int[][] observaciones, int iteraciones,
                                  double tolerancia,bool modo)
       {
           int N = observaciones.Length;
           double p=1;
           int [][] secuencias=new int[N][];

       double likelijud = 0,oldlilijud=double.MinValue;
       int currentiteracion=0;
       bool stop=false;
       do
```

```
        {
            double[,] mA = new double[States, States];
            double[,] mB = new double[States, Symbols];
            double[] vpi = new double[States];

            for (int i = 0; i < N; i++)
            {
                secuencias[i] = this.Decode(observaciones[i], out p);
//algoritmo viterbi
                Console.WriteLine();
                interpreta(secuencias[i]);

                if ((currentiteracion == 0) && (modo==false))
                {
                    for (int k = 0; k < observaciones[i].Length; k++)
                        secuencias[i][k] = k * States /
 observaciones[i].Length;
                }

                for (int j = 0; j < observaciones[i].Length - 1; j++)
                     mA[secuencias[i][j], secuencias[i][j + 1]] =
mA[secuencias[i][j], secuencias[i][j + 1]] + 1;

                for (int j = 0; j < observaciones[i].Length; j++)
                     mB[secuencias[i][j], observaciones[i][j]] =
mB[secuencias[i][j], observaciones[i][j]] + 1;

                vpi[secuencias[i][0]] = vpi[secuencias[i][0]] + 1;
                likelijud += p;// Math.Log(p);

            }
            Console.WriteLine("likelihood " + likelijud);
```

```
if (checkConvergence(oldlilijud, likelijud,
       currentiteracion, iteraciones, tolerancia))
{
    stop = true;
}

else
{
    for (int i = 0; i < States; i++)
    {
        double total = 0;
        for (int j = 0; j < States; j++)
            total = total + mA[i, j];
        for (int j = 0; j < States; j++)
        {
            if (mA[i, j] == 0)
            {
                mA[i, j] = 0.00001;
            }
            else
            {
                mA[i, j] = mA[i, j] / total;
            }

        }
        total = 0;
        for (int j = 0; j < Symbols; j++)
            total = total + mB[i, j];
        for (int j = 0; j < Symbols; j++)
        {
            if (mB[i, j] == 0)
```

```
                {
                    mB[i, j] = 0.00001;
                }
                else
                {
                    mB[i, j] = mB[i, j] / total;
                }
            }


            vpi[i] = vpi[i] / (double)N;
            if (vpi[i] == 0)
                vpi[i] = 0.00001;
        }

        oldlilijud = likelijud;
        likelijud = 0;

        A = mA;
        B = mB;
        pi = vpi;

        currentiteracion += 1;
    }
} while (!stop);
Console.WriteLine("modelo entrenado");

return likelijud;

}
```

A.2. Clasificador utilizando un grupo de Modelos Ocultos de Markov

```
 public int ComputeVit(int[] sequence, out double likelihood)
       {

           File.AppendAllText("Reporte.txt", "Clasifcador HMM "+
DateTime.Now);
           int label = 0;
           likelihood = double.MinValue;

           for (int i = 0; i < Nmod ; i++)
           {
               double p;
               int[] pat = new int[sequence.Length];
               pat = modelos[i].Decode(sequence, out p);

               string path = VoiceProcessor.stringea(pat);

               File.AppendAllText("Reporte.txt",p+" ");

               if (p > likelihood)
               {
                   label = i;
                   likelihood = p;

               }
           }


           File.AppendAllText("Reporte.txt", "Reconoci "+
nombres[label]+"\n");
```

```
            return label;
        }
```

A.3. Obtencion de coeficientes cepstrales y delta cepstrales

```
  public static double[][] cepstral(double[][] a, float[][] bloqcorr, int q)
        {
            double[][] cepstrum=new double[a.Length][];
            double[][] delta = new double[a.Length][];
            double[][] par = new double[a.Length][];

            double[] w=new double[q+1];
            for (int m = 0; m <= q; m++)
            {
                w[m] = 1 + (double)q / 2 * Math.Sin(Math.PI *
(double)m / (double)q);
            }

            for (int i = 0; i < a.Length; i++)
            {
                cepstrum[i] = LPC2Ceps(a[i], bloqcorr[i][0], q);
                cepstrum[i] = multiplicacion(cepstrum[i], w);
            }

            for (int i = 2; i < a.Length - 2; i++)
            {
                delta[i] = resta(cepstrum[i + 2], cepstrum[i - 2]);
                delta[i] = division(delta[i], 0.5);
```

```
            delta[i] =suma(delta[i], cepstrum[i+1]);
            delta[i] = resta(delta[i], cepstrum[i - 1]);
            delta[i] = division(delta[i], 2.7);
        }

        delta[a.Length - 1] = new double[delta[2].Length];
        delta[a.Length -2] = new double[delta[2].Length];
        delta[1] = new double[delta[2].Length];
        delta[0] = new double[delta[2].Length];

        delta[2].CopyTo(delta[1],0);
        delta[2].CopyTo(delta[0],0);
        delta[a.Length-3].CopyTo(delta[a.Length-2],0);
        delta[a.Length-3].CopyTo(delta[a.Length-1],0);

        par = cat(cepstrum, delta);
        return par;

    }
```

A.4. Transformacion de coeficientes LPC a cepstrales

```
public static double[] LPC2Ceps(double[] a,float r,int q)
    {
        double[] coef = new double[q + 1];
        int p = a.Length;
        coef[0] = r;
        for (int m = 1; m <= q; m++)
        {
            if (m <= p)
```

```
                    coef[m] = a[m-1];
                for (int k = 1; (k < m) && (k<=p); k++)
                {
                    coef[m] = coef[m] + (double)(m-k)/(double)m
* coef[m-k] * a[k-1];
                }
            }
            return coef;

        }
```

Apéndice B

Glosario

Cepstrum: Señal que se obtiene al obtener la transformada de Fourier del logaritmo natural del espectro de Fourier de potencia de una señal.

Codebook: Conjunto de códigos; en procesamiento de voz se refiere a un conjunto de vectores que representan una señal de voz y son los centroides resultantes de un proceso de cuantización vectorial.

Coeficientes cepstrales: Coeficientes basados en el cepstrum. que representan la señal de voz.

coeficientes delta-cepstrales Utilizando los coeficientes cepstrales, se obtiene la razón de cambio de éstos a lo largo del tiempo para incluir un análisis dinámico de la señal.

CV: Cuantización vectorial.

Cuantización vectorial: Proceso en el que se genera un conjunto de regiones de un espacio vectorial, a cada región le corresponde un centroide y se asigna un indice a cada vector según la región del espacio a la que pertenece.

Entrenamiento: Proceso en el cual un sistema computacional modela un patrón característico de cada clase diferente para después reconocer a que clase pertenece

una señal desconocida:

HMM: Modelo Oculto de Markov (Hidden Markov Model), es un modelo probabilístico que contiene cierta cantidad de estados y un grupo de funciones de probabilidad que describen la transición en estados y la probabilidad de observar determinados símbolos en cada estado.

LPC: Coeficientes de predicción lineal (Linear prediction coefficients), describen una señal de voz basados en la predicción lineal y el modelo de un filtro que caracteriza el tracto vocal de una persona.

MFCC: Coeficientes cepstrales en la escala mel (Mel-frequency cepstral coefficients); representan la señal de la voz basados en el cepstrum dispuesto sobre una escala logarítmica en el dominio de la frecuencia.

Predicción lineal: Teoría que señala que algunas señales pueden aproximarse como una combinación lineal de muestras anteriores de esa misma señal.

Variables utilizadas:

- **A:** Matriz de NxN donde a_{ij} representa la probabilidad de transitar de un estado i a un estado j en un Modelo Oculto de Markov.
- **B:** Matriz de NxM que representa una función de probabilidad discreta donde el elemento b_{ij} representa la probabilidad de observar el símbolo j en el estado i en un Modelo Oculto de Markov.
- λ**:** Representa un Modelo Oculto de Markov en particular.
- μ**:** Vector de medias de una función densidad de probabilidad gaussiana multidimensional.
- $N(o, \mu, \Sigma)$**:** Función densidad de probabilidad gaussiana multidimensional.
- **O:** Vector de observaciones en un Modelo Oculto de Markov.
- π**:** Vector que representa la probabilidad de comenzar en un estado determinado en un Modelo Oculto de Markov.

- **Q:** Vector que representa una secuencia de estados en un Modelo Oculto de Markov.
- **Σ:** Matriz de covarianzas de una función densidad de probabilidad gaussiana multidimensional.

Bibliografía

[1] Halima Bahi y Nadia Benati. A new keyword spotting approach. *IEEE*, 2009.

[2] Leonard E. Baum y J. A. Eagon. An inequality with applications to statistical estimation for probabilistic functions of markov processes and to a model for ecology. *Bulletin American Meteorological Society*, 1967.

[3] Leonard E. Baum y Ted Petrie. Statistical inference for probabilistic functions of finite state markov chain. *The Annals of Mathematical Statistics*, 1966.

[4] Leonard E. Baum, Ted Petrie, George Soules, y Norman Weiss. A maximization technique occurring in the statistical, analysis of of probabilistic functions of markov chains. *The Annals of Mathematical Statistics*, 1970.

[5] Jesús Savage Carmona. *A Hybrid System with Symbolic AI and Statistical Methods for Speech Recognition.* Tesis de Doctorado, Universidad de Washington, 1995.

[6] Heriberto Cuayáuitl y Ben Serridge. Out of vocabulary word modeling and rejection for spanish keyword spotting systems. *Proc. MICAI'02*, 2002.

[7] John R. Deller, John H.L. Hansen, y John G. Proakis. *Discrete-Time Processing of Speech Signals.* IEEE Press Editorial, 1993.

[8] James Durbin. The fitting of time-series models. *Review of the International Statistical Institute*, 1960.

[9] G. David Forney. The viterbi algorithm. *Proc. IEEE*, 1973.

[10] Christian Gargour, Marcel Gabrea, Venkatanarayana Ramachandran, y Jean-Marc Lina. A short introduction to wavelets and their applications. *IEEE CIRCUITS AND SYSTEMS MAGAZINE*, 2009.

[11] Alex Graves, Abdel rahman Mohamed, y Geoffrey Hinton. Speech recognition with recurrent neural networks. *ICASSP*, 2013.

[12] Simon Haykin. *Neural Networks A comprehensive Foundation*. Prentice Hall, 1999.

[13] Xuedong Huang, Acero A, y Hon H. *Spoken Language Processing-A Guide to theory, Algorithms, and System Development*. Prentice Hall, 2001.

[14] Xuedong Huang y Li Deng. *Handbook of Natural Language Processing*. 2009.

[15] Mahdi Jamaati, Hossein Marvi, y Milad Lankarany. Vowels recognition using mellin transform and plp-based feature extraction. *Acoustics-08 Paris*, 2008.

[16] Biing H. Juang, Lawrence R. Rabiner, y Jay G. Wilpon. On the use of bandpass liftering in speech recognition. *IEEE Trans. Acoustic Speech Signal Processing*, 1987.

[17] Ingrid Kirschning. *Automatic Speech Recognition with the Parallel Cascade neural Network*. Tesis de Doctorado, Universidad de Tokushima, 1998.

[18] Su Jan Leow, Tze Siong Lau, Alvina Goh, Han Meng Peh, Teck Khim Ng, Sabato Marco Siniscalchi, y Chi-Hui Lee. A new confidence measure combining hidden markov models and artificial neural networks of phonemes for effective keyword spotting. *IEEE*, 2012.

[19] Norman Levinson. The wiener rms (root mean square) error criterion in filter design and prediction. *Journal of Mathematics and Physics vol 25*, 1947.

[20] Yoseph Linde, Andres Buzo, y Robert M. Gray. An algorithm for vector quantizer design. *IEEE Trans. on Communications*, 1980.

[21] Eduardo Lleida, José B. Marino, Josep Salavedra, Antonio Bonafonte, Enrique Monte, y A. Martinez. Out of vocabulary word modeling and rejection for keyword spotting. *EUROSPEECH'93*, 1993.

[22] Stuart P. Lloyd. Least squares quantization in pcm's. *IEEE Trans. on Communications*, 1957.

[23] John Makhoul. *Linear prediction: A tutorial review*, 1976.

[24] Lawrence Rabiner y Biing-Hwang Juang. *Fundamentals of Speech Recognition*. Prentice Hall International Inc., 1993.

[25] Lawrence R. Rabiner. *A tutorial on Hidden Markov Models and selected applications in speech recognition*, 1989.

[26] Lawrence R. Rabiner y Ronald W. Schafer. *Digital Processing of Speech Signals*. Prentice Hall, 1978.

[27] J.R. Rohlicek, P. Jeanrenaud, K. Ng, H. Gish, B. Musicus, y M. Siu. Phonetic training and language modeling for word spotting. *ICASSP*, págs. II 459–II 463, 1993.

[28] Paul Taylor. *Text to Speech Synthesis*. Cambridge University, 2009.

[29] Joe Tebelskis. *Speech Recognition using Neural Networks*. Tesis de Doctorado, Universidad Carnegie Mellon, 1995.

[30] Javier Tejedor y José Colás. Spanish keyword spotting system based on filler models, pseudo n-gram language model and a confidence measure. En *IV Jornadas en Tecnología del Habla*. 2006.

[31] Y. Tokhura. A weighted cepstral distance measure for speech recognition. *IEEE Trans. Acoustic Speech Signal Processing*, 1987.

[32] Andrew J. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE trans. Informat. Theory*, 1967.

[33] Jay G. Wilpon, Lawrence R. Rabiner, Chin hui Lee, y E. R. Goldman. Automatic recognition of keywords in unconstrained speech using hidden markov models. *IEEE*, 38, 1990.