Universidad Central de Venezuela
Facultad de Ciencias
Escuela de Computación
Centro de Ingeniería de Software y Sistemas
Laboratorio de Inteligencia Artificial

PROTOTIPO DE UN RECONOCEDOR DE VOZ PARA EL IDIOMA ESPAÑOL

Trabajo Especial de Grado
presentado ante la ilustre
Universidad Central de Venezuela
por el bachiller
David Castro
Para optar al título de
Licenciado en Computación

Tutores
Haydemar Nuñez
Esmeralda Ramos

Caracas, 2011

Universidad Central de Venezuela
Facultad de Ciencias
Escuela de Computación

ACTA DEL VEREDICTO

Quienes suscriben, miembros del jurado designado por el Consejo de la Escuela de Computación, para dictaminar sobre el Trabajo Especial de Grado titulado: "**Prototipo de un reconocedor de voz para el lenguaje español**" y presentado por el bachiller David Alejandro Castro Briceño, cédula de identidad V –17.423.089, para optar al título de Licenciado en Computación, dejan constancia de lo siguiente:

Leído como fue, dicho trabajo por cada uno de los miembros del jurado, se fijó el día 27 de septiembre de 2011 a las 10:00 a.m., para que su autor lo defendiera en forma pública, lo que hizo en el aula PB-III de la Escuela de Computación, mediante una presentación oral del contenido del Trabajo Especial de Grado, luego de lo cual respondió a las preguntas formuladas. Finalizada la defensa pública del Trabajo Especial de Grado, el jurado decidió **APROBARLO**.

En fe de lo cual se levanta la presente Acta, en la Ciudad Universitaria de Caracas a los veintisiete días del mes de septiembre del año dos mil once, dejándose también constancia de que actúo como Coordinadora del Jurado la Profesora Haydemar Núñez.

Jurado Principal

_____	_____
Profesora Haydemar Núñez	Profesora Esmeralda Ramos
(Tutora)	(Tutora)
_____	_____
Profesor Rhadamés Carmona	Profesor Robinson Rivas
(Jurado)	(Jurado)

PROTOTIPO DE UN RECONOCEDOR DE VOZ PARA EL IDIOMA ESPAÑOL

RESUMEN

Actualmente en el área de reconocimiento de voz se han producido grandes avances, que han permitido la reducción de la tasa de errores y la independencia del hablante. Un beneficio que presentan estas aplicaciones de reconocimiento se encuentra en el área de la telefonía, ya que permiten la interacción de una persona a través del teléfono usando comandos de voz para poder navegar a través de un sistema.
El problema que tienen estas aplicaciones, y más para el lenguaje español, es la poca disponibilidad de sistemas que hay en el mercado, donde la gran mayoría no están disponibles al público general. Otro punto en contra para el reconocimiento de voz en español es la falta de corpus de entrenamiento que permitan hacer sistemas lo suficientemente robustos a bajo costo.
Tomando en cuenta la situación actual del reconocimiento de voz para el español, en este trabajo se propone la construcción de un prototipo de reconocimiento de palabras aisladas, usando las bondades que pueden aportar los modelos ocultos de Markov, haciendo uso de plataformas open source tanto para el entrenamiento de los modelos, como para el reconocimiento de la señal de voz..
Entre los resultados obtenidos con los modelos construidos se logró un reconocimiento efectivo de 83.1% y 91.63%, en base a los criterios de evaluación definidos, con grabaciones de audio en situaciones reales. A su vez se logró la integración exitosa con una aplicación telefónica para pruebas en vivo. En conclusión, fue posible la construcción de los modelos, que luego de varios ajustes, obtuvieron buenos niveles de reconocimiento, junto con una integración exitosa en sistemas de telefonía para su uso en aplicaciones del mundo real.

Palabras claves:
Reconocimiento de voz, modelos ocultos de markov, aplicaciones en telefonía

Autor:

David Alejandro Castro Briceño

Tutores:

Prof. Haydemar Nuñez

Prof. Esmeralda Ramos

Fecha: Septiembre 2011

ÍNDICE

ÍNDICE DE FIGURAS

INTRODUCCIÓN

Entender el lenguaje hablado no es una tarea sencilla, lleva años poder desarrollarla y es aquella que nos permite comunicar nuestras ideas mediante sonidos con otras personas. El objetivo de este trabajo es intentar emular esta capacidad humana en un sistema computacional que permita realizar la transcripción de una señal de audio a texto.

El área de reconocimiento de voz da sus primeros pasos en los años 50, cuando se creó la primera máquina de reconocimiento aislado de dígitos, junto con trabajos para el reconocimiento de vocales y algunas consonantes, utilizando técnicas basadas en el reconocimiento de patrones. Durante los 70 se desarrolla el enfoque de los modelos ocultos de Markov hacia el reconocimiento de voz, lo que impulsó un cambio de paradigma en los años 80, haciendo uso de las técnicas probabilísticas (Pérez, 2006).

El uso de sistemas de reconocimiento de voz, particularmente para el idioma español, se ha visto limitado por la poca existencia de aplicaciones en el mercado, donde la gran mayoría no están disponibles al público general y son justamente estás las que tienen mejores prestaciones en aplicaciones del mundo real. Esto hace que se limite su uso a aplicaciones sencillas con bajo nivel de interacción, rendimiento y una tasa de error baja. Por ello, en este trabajo se propone la construcción de un prototipo de reconocedor de voz en el idioma español, enfocado al uso en aplicaciones telefónicas de atención al cliente, usando el proyecto *Sphinx* (http://cmusphinx.sourceforge.net/) de la *Carnegie Mellon University* para la extracción de características y construcción de modelos usando la técnica de modelos ocultos de Markov, que permita realizar la decodificación de una señal de audio a texto.

El trabajo se divide en dos Capítulos; el primero contiene una breve introducción al área del reconocimiento de voz, técnicas usadas, características que hay que tomar en cuenta al

momento de la construcción de un reconocedor, arquitectura general de un reconocedor de voz y los diferentes módulos que se necesitan para poder realizar la codificación de voz a texto.

En el segundo Capítulo se presenta el problema del reconocimiento de voz para el español, la solución propuesta, los objetivos planteados para poder dar solución al problema, la arquitectura de cada uno de los módulos que se van a utilizar para la extracción de las características de la señal, la construcción de los modelos acústicos, la decodificación de la señal y los resultados obtenidos luego de realizadas las pruebas a los modelos acústicos.

Por último, se presentan las conclusiones y una serie de recomendaciones para trabajos futuros.

CAPÍTULO I. MARCO TEÓRICO

1.1 Reconocimiento de Voz

El reconocimiento automático de voz se entiende como el proceso en el cual un sistema computacional recibe como entrada una señal de audio, que puede ser una frase o bien una palabra, de la cual se extrae la información acústica, y produce como salida un texto, que se corresponde con la señal original. Este proceso se puede ver como una tarea de reconocimiento de patrones, en donde se toma un patrón de entrada, que en este caso seria la señal de voz y se clasifica dentro de un conjunto establecido. La dificultad está en la variabilidad de las características propias del hablante, como son la edad, sexo, la velocidad de pronunciación, la forma en que se expresan y la región a la que pertenezca la persona, entre otros (Pérez, 2006).

Los sistemas de reconocimiento, dependiendo de sus características, se pueden clasificar en (Baghdasarya, 2010; Jayadev 2007):

- **Sistemas dependientes o independientes del hablante:** los sistemas dependientes son capaces de extraer información de un hablante en particular, cuyas características acústicas estén grabadas en una base de datos que permita realizar el reconocimiento de lo que dice la persona, en cambio los sistemas independientes están en la capacidad de extraer información sin tomar en cuenta las características acústicas de la persona, esto hace que sean más complejos a nivel computacional.
- **Sistemas de reconocimiento de palabras aisladas o reconocimiento continuo del habla:** para el reconocimiento de palabras aisladas, el hablante tiene que hacer pausas al decir alguna frase, de aproximadamente 200 milisegundos entre palabras, que permitan diferenciar el comienzo y el final de cada una. En el caso del reconocimiento continuo se reconocen secuencias de palabras donde no son necesarias las pausas, por lo que son sistemas de mayor complejidad, ya que se necesita diferenciar cada palabra en una señal

continua de audio.

- **Sistemas de reconocimiento de voz dependientes del tamaño del vocabulario:** pueden estar categorizados en vocabularios pequeños que tienen aproximadamente 100 palabras, vocabularios medianos con aproximadamente 1000 palabras y sistemas de vocabularios grandes con más de 10000 palabras. A medida que el vocabulario crece también aumentan los requerimientos de memoria y la complejidad computacional. Generalmente los sistemas con grandes vocabularios usan unidades de reconocimiento basados en sub-palabras, también conocidos como fonemas, que compensan la falta de datos para el entrenamiento de los modelos del sistema.

1.1.1 Aplicaciones

Los sistemas de reconocimiento de voz, dada su forma de interacción con el usuario, suelen tener las siguientes aplicaciones (Puertas, 2000):

- **Sistemas de dictado automático:** es uno de los usos más comunes para este tipo de tecnología, pueden ser sistemas independientes del hablante; sistemas adaptados al hablante, donde antes de utilizar la aplicación se le pide al usuario que haga un entrenamiento previo con algún texto para luego poder comenzar a utilizarlo; y sistemas que se van adaptando a medida que son utilizados. Ejemplos de estos sistemas son *ViaVoice* de IBM y *Dragon Sytems*.

- **Sistemas basados en comandos:** tienen como objetivo, dar órdenes concretas a un sistema, por lo general tienen un buen rendimiento por ser sistemas orientados a tareas específicas, por lo que no manejan un vocabulario muy grande.

- **Telefonía:** son sistemas que interactúan con el usuario a través de comandos de voz para navegar por un menú de opciones, son utilizados como una alternativas a los sistemas de tono o de entrada de dígitos por medio del teclado telefónico. Son en su mayoría utilizados por sistemas de atención al cliente.

- **Traducciones Automáticas:** son sistemas de traducción que permiten que el texto a traducir sea ingresado de forma hablada, lo que lo hace más práctico para dispositivos portables. Un ejemplo de esto, es el traductor de Google http://translate.google.com/ que permite la introducción del texto a traducir de forma hablada.

1.2 Arquitectura de un reconocedor de voz

Resolver el problema de reconocimiento de voz no es una tarea trivial y han surgido distintas metodologías para resolverlo, una de las más utilizadas y de la que se han obtenido mejores resultados consiste en que, a partir de una señal de entrada, se realice un proceso de codificación, usando como referencia un conjunto de modelos definidos en una fase previa de entrenamiento. Para esto se realiza un proceso de extracción de características, con el fin de obtener la mayor cantidad de información posible del habla y en donde se va a tratar de eliminar la información asociada al canal de comunicación. Este método consiste en dos procesos o fases. Una fase de entrenamiento, en la cual se generan los diferentes modelos de referencia, obtenidos a través de una base de datos de voces que representa un conjunto representativo de variaciones acústicas de la lengua, y una fase de reconocimiento, en el cual se usan los modelos de referencia y donde se elige la secuencia de palabras más cercana al modelo entrenado.

Para la construcción de un sistema de reconocimiento de voz, empleando la metodología anterior, se utilizan una serie de componentes: diccionario de pronunciación, modelo acústico y modelo de lenguaje; y de una serie de fases: extracción de características, entrenamiento de modelos y búsqueda o decodificación. Estos componentes y fases definen la arquitectura global de un sistema automatizado de reconocimiento de voz y cuya interacción se puede observar en la Figura 1, donde la “Señal de audio” y el “Patrón reconocido” corresponden a las entradas y salidas del sistema, respectivamente.

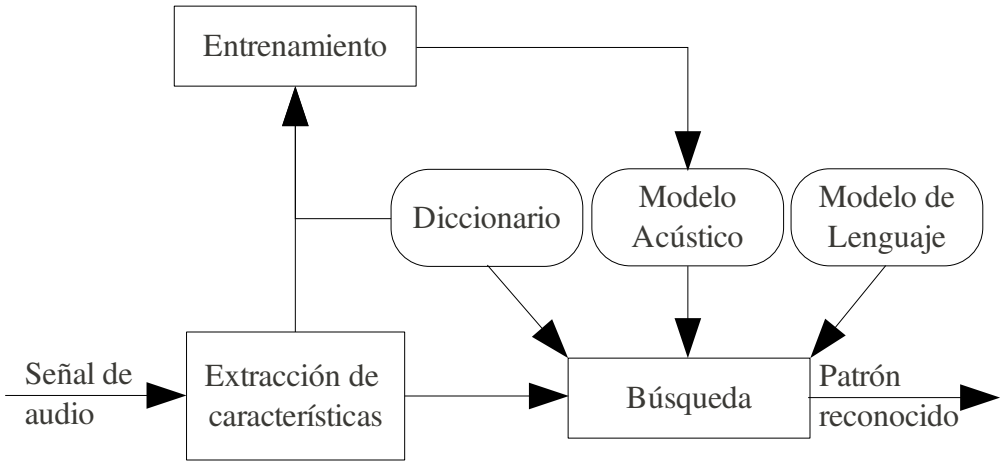

Figura 1: Componentes de un reconocedor de voz

1.2.1 Extracción de Características

En la fase de extracción se generan una serie de vectores, que contienen la información más relevante de la señal de voz. Para poder realizar esta extracción primero hay que entender cómo se produce la voz humana y cómo, a través de distintas técnicas, se pueden extraer las características más relevantes de una señal. Procesos que serán explicados a continuación.

1.2.1.1 Producción de la Voz

Para la producción del sonido, el aparato fonador compuesto por la lengua, mandíbula, labios, etc (Figura 2) modula la presión del aire que fluye a través de las cuerdas vocales para producir una secuencia reconocida de sonidos (Deller, Hansen, y Proakis 1993). Durante el habla, las cuerdas vocales permanecen en un movimiento continuo, abriéndose y cerrándose, lo que origina una mezcla de impulsos o características en la señal de voz. Estos impulsos son: (Plannerer, 2005)

- **Impulsos sonoros**: este tipo de impulsos se caracteriza en que las cuerdas vocales tienen un estado inicial, donde se encuentran cerradas, la presión de aire produce vibraciones

que ocasionan que las cuerdas vocales se abran y se cierren periódicamente, produciendo un flujo casi continuo de aire.

- **Impulsos no sonoros**: las cuerdas vocales permanecen abiertas y el aire pasa por el tracto vocal en forma turbulenta sin producir vibraciones en las cuerdas vocales, este tipo de impulso se caracteriza por tener frecuencias bajas y es considerado como ruido blanco.

En algunos tipos de sonidos estos dos impulsos se pueden presentar en combinación para producir distintos sonidos.

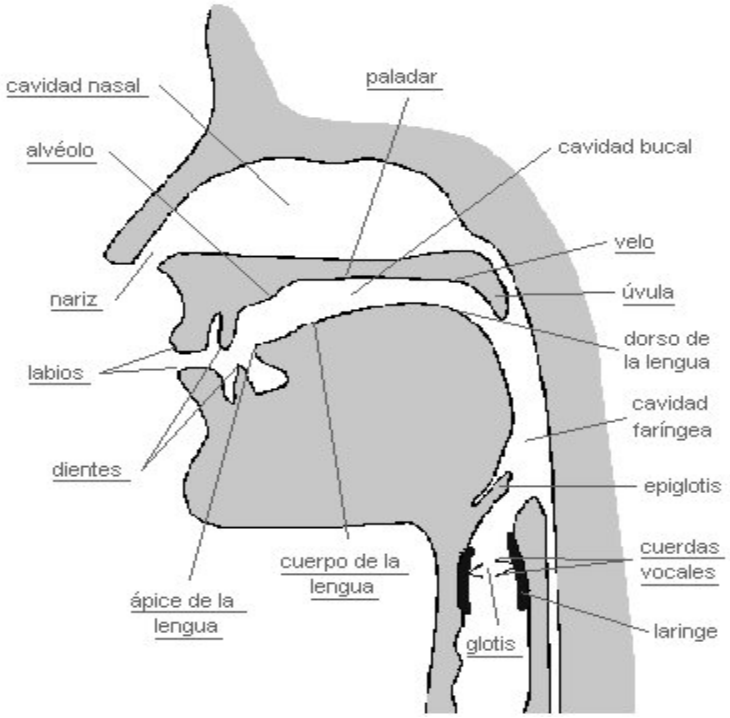

Figura 2: Aparato fonador

La forma espectral de una señal de voz está determinada por la forma del tracto vocal; éste se puede visualizar como una tubería, formada por la garganta, la lengua, dientes y labios,

los cambios en la forma del tracto vocal, producen variaciones en la forma de la señal de voz, que produce la articulación de distintos sonidos en el habla, estos cambios se ven reflejados como cambios en la resonancia de la frecuencia o picos en el espectro de la señal, también llamados formantes. Los formantes contienen la mayor información de la señal y están formados de impulsos correspondientes a las vibraciones producidas por las cuerdas vocales (Shaneh y Taheri, 2009).

1.2.1.2 Técnicas de extracción de características

Como se mencionó anteriormente, la información más importante de la voz está contenida en la forma en que varía la señal espectral, para poder reflejar estos cambios dinámicos se realizan una serie de procesos, descritos a continuación:

La señal pasa a través de un proceso de pre-énfasis, en el cual se compensa la pérdida que sufre la señal de voz en sus componentes de alta frecuencia, para esto se pasa la señal a través de un filtro paso alto que contrarresta la pendiente espectral negativa de la señal, atenuando las frecuencias más bajas.

Para reflejar los cambios dinámicos de la señal de voz se calcula una serie de parámetros o características, en intervalos cortos de tiempo, que por lo general se estima que sean de 10 milisegundos. La señal de voz se divide en bloques (Figura 3), que se obtienen, multiplicando la señal de voz $s(k)$ por una ventana $w(k)$, donde se obtiene un fragmento de la señal $v_m(k)$, para $k=m, m+1, \ldots, m+N-1$, donde N representa el tamaño de la ventana con valores entre los 16 y 25 milisegundos, con desplazamientos de 10 milisegundos para calcular los parámetros de la señal. La ventana más usada para la extracción de características es la ventana de Hamming, que se define como:

$$w(k)=\begin{cases} 0.54-0.46\cos\left(\frac{2\pi n}{N-1}\right) & n=0,1,\ldots,N-1 \\ 0 & sino \end{cases}$$

donde *m* se incrementa en intervalos de 10 milisegundos.

$$v_m(k)=\begin{cases} s(k)w(k-m) & k=m,m+1,\ldots,m+N-1 \\ 0 & sino \end{cases}$$

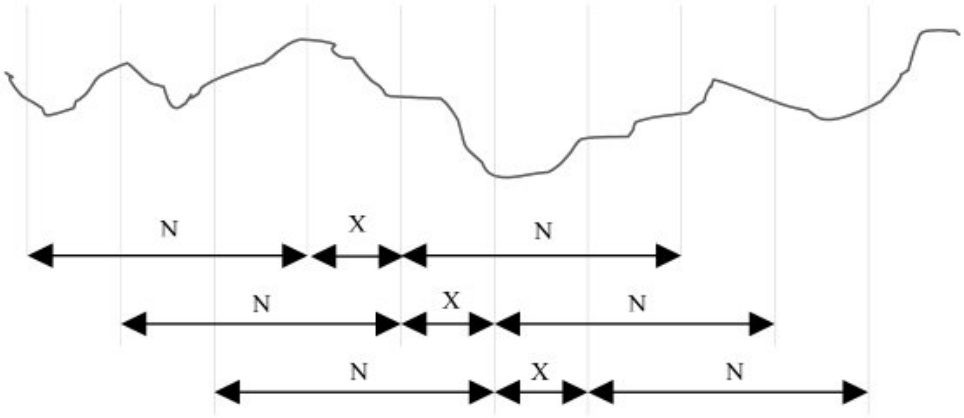

Figura 3: División de una señal en bloques:
Donde N es el tamaño de la ventana (16 o 25 milisegundos) y
X es un intervalo de tiempo de aproximadamente 10 milisegundos

Luego la señal de voz se codifica en un conjunto de parámetros, que van a representar la señal espectral, para esto se realiza un análisis de la señal usando técnicas de Codificación Linear Predictiva (LPC - Linear Predictive Coding) o de análisis cepstral, calculando Coeficientes Espectrales de Frecuencia Mel (MFCC - Mel Frecuency Cepstral Coefficients). Como se describe a continuación.

- **Codificación Linear Predictiva**

Viendo a la señal de voz como una serie de impulsos provenientes de la acción de las cuerdas vocales se tiene el siguiente modelo (Figura 4):

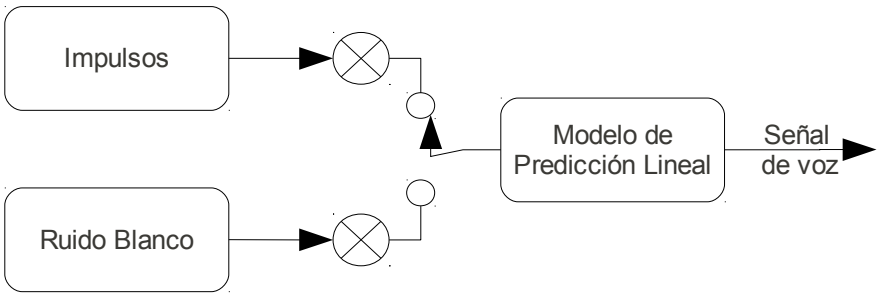

Figura 4: Modelo para el análisis de predicción lineal

En este modelo, la voz se modela como una serie de impulsos, provenientes de la acción de las cuerdas vocales que alterna con el ruido blanco, esta señal luego pasa al tracto vocal, en donde se origina la forma espectral de la señal. El resultado de este proceso origina una serie de coeficientes que contienen información de la señal de voz original.

- **Análisis Cepstral**

Para realizar el análisis Cepstral se utiliza el proceso mostrado en la Figura 5, en donde:

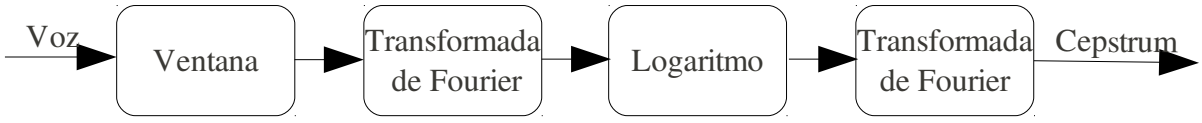

Figura 5: Modelo de análisis cepstral de la señal de voz

La transformada de Fourier proporciona una representación de la señal en términos de amplitud y fase dentro del dominio de la frecuencia. Para muestras de señales en unidades discretas de tiempo se utiliza la transformada discreta de Fourier (DFT - Discrete Fourier

Transformation).

$$\hat{V}(n)=\sum_{k=0}^{N-1} v(k)\mathrm{e}^{\frac{-J2\pi kn}{N}}, n=0,1,\ldots,N-1$$

Una implementación eficiente en términos computacionales es la Transformada Rápida de Fourier (FFT – Fast Fourier Transformation), mediante la cual se obtienen las mismas características espectrales de la DFT.

A través del cálculo del logaritmo de la señal, se obtiene una nueva señal en el dominio de la frecuencia, al realizar un nuevo cálculo de la transformada de Fourier se logra separar la información correspondiente al tracto vocal y a los impulsos provenientes de las cuerdas vocales, esta nueva señal es llamada Cepstrum (Childers, Skinner y Kemerait 1977).

Mejoras en el uso de este modelo se logran utilizando técnicas de filtrado con coeficientes espectrales de frecuencia Mel, siendo ésta, una de las técnicas más utilizadas, ya que proporciona una mejor representación de los cambios dinámicos de la señal de voz.

- **Coeficientes Espectrales de Frecuencia Mel**

Se basa en la premisa de que el sistema auditivo humano no sigue una escala lineal, sino que cada tono de frecuencia *f*, es mapeado dentro de una escala llamada escala Mel, la ventaja del uso de este tipo de frecuencias es que se obtienen grandes prestaciones en la captura de características fonéticas de la voz. Para este método se utiliza un banco de filtros triangulares espaciados uniformemente en la escala Mel, de la cual se obtienen bandas de energía, que pasaran a formar parte del cepstrum (Shaneh et al., 2009).

Para el caso de los coeficientes espectrales de Mel se pueden usar tanto la transformada de Fourier como la transformada del coseno (DCT), para el cálculo del cepstrum (Figura 6).

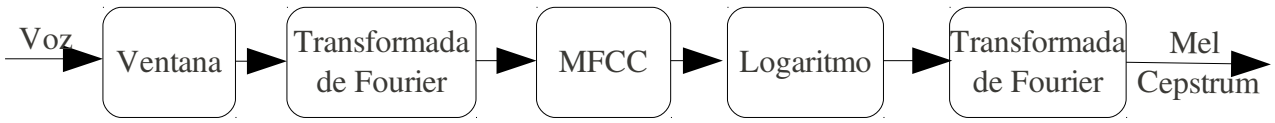

Figura 6: Analísis ceptral con frecuencias en la escala Mel

1.2.2 Entrenamiento

En la fase de entrenamiento se usan los vectores de características, resultado del proceso de extracción, que junto con un diccionario de pronunciación, transcripciones de los audios de entrenamiento y la definición de las unidades de reconocimiento, permiten la construcción de los modelos acústicos.

Para la fase de construcción de los modelos acústicos existen diversos enfoques, los cuales pueden usar técnicas probabilísticas, técnicas de reconocimiento de patrones o técnicas de inteligencia artificial. Una de las técnicas probabilísticas más utilizada son los modelos de Markov, por tener una estructura matemática bien definida que produce muy buenos resultados y es ésta técnica la que se va a describir a continuación, resaltando cuales son sus elementos, que procedimientos se utilizan para su construcción y que tipo de modelos permite construir.

1.2.2.1 Modelos ocultos de Markov

Un modelo oculto de Markov (HMM) es una máquina de estados finitos, donde las secuencias de estados quedan ocultas y las transiciones vienen dadas por una función de probabilidad del estado. Una topología de modelo oculto de Markov, ideal para modelar el reconocimiento de voz, ya que la secuencia de estados es incremental en el tiempo, es el modelo izquierda-derecha (*left-right*) o modelo de Bakis (1976) (Figura 7).

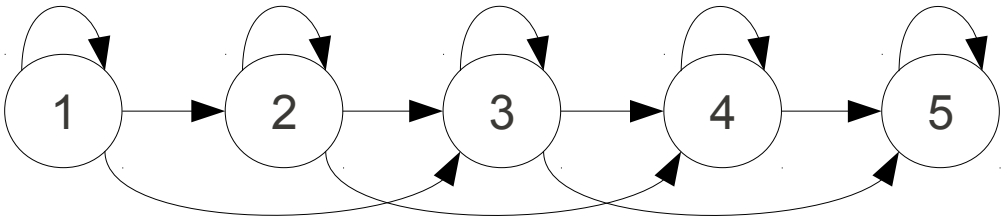

Figura 7: Modelo de Bakis con 5 estados

Los modelos ocultos de Markov pueden tener distintas topologías y distintos valores para las distribuciones de probabilidad entre observaciones o diferentes tipos de conexiones entre estados, pero todos tienen en común los siguientes elementos:

- Está formado por *N* estados.
- Existen *M* posibles símbolos para cada uno de los estados del modelo. Los símbolos se corresponde con las salidas que presenta el sistema, también llamados símbolos de observación.
- La salida final del modelo de Markov se conoce como secuencia de observaciones *O*.
- Existe una distribución de probabilidad para la transición entre estados.
- Existe una distribución de probabilidad para los símbolos de un estado.
- Existe una distribución de probabilidad propia del estado.

Con estos elementos se puede observar que un modelo oculto de Markov requiere de la especificación del número de estados, el número posible de símbolos, la definición de los símbolos de observación y de las tres probabilidades para los estados, símbolos y transiciones, que por conveniencia se denotan como λ, y son estas probabilidades las que definen el modelo.

- **Problemas básicos para la definición de un HMM**

  La correcta formulación de un modelo oculto de Markov en aplicaciones del mundo real,

se ve afectada por los siguientes problemas:

- Dada una secuencia de observaciones y un modelo λ ¿Cómo se puede calcular de manera eficiente la probabilidad de la secuencia de observaciones para el modelo $P=[O|\lambda]$?
- Dada una secuencia de observaciones O y un modelo λ ¿Cómo se escoge la correspondiente secuencia de estados que sea óptima?
- ¿Cómo se ajustan los parámetros del modelo λ que maximicen $P=[O|\lambda]$? En este problema se optimizan los parámetros de manera tal que describan de la mejor forma la secuencia de observaciones. Esta secuencia O se usa para el entrenamiento del modelo y forma parte de una de las tareas más importantes a la hora de construir un modelo oculto de Markov, ya que permite optimizar los parámetros de probabilidad que se adaptan de forma óptima al conjunto observado.

- **Tipos de modelos ocultos de Markov (Puertas, 2000)**

Para realizar el proceso de clasificación es necesario asignar una distribución de probabilidad a cada uno de los símbolos de observación de un estado, la forma en que se asigne este valor de probabilidad da origen a diferentes tipos de modelos ocultos de Markov. Estos modelos pueden ser de tipo discretos, continuos y semi-continuos, descritos a continuación.

- **Discretos:** se utiliza un proceso de clasificación para los vectores de características, un ejemplo es el método de cuantificación vectorial (*Vector Quantization* – VQ) (Linde, Buzo y Gray 1980; Rogers, 1989), que retorna el conjunto de observaciones como vectores de símbolos con M elementos diferentes, llamados *codebook*.
- **Continuos:** las distribuciones son densidades de probabilidad de espacios de observación continua, esta distribución es la mezcla de un conjunto de funciones de tipo gaussiano, para la cual se define la matriz de medias y la de covarianzas. Para estimar estos parámetros se necesitan grandes cantidades de datos para el entrenamiento,

obteniendo así una gran cantidad de vectores de entrenamiento, que aumentan de forma lineal con el número de modelos a entrenar.

- **Semi-Continuos:** surgen como una alternativa ante la necesidad de entrenar con bases de datos limitadas, con pocas horas de audio. Estos modelos, al igual que los modelos continuos, usan funciones de densidades de probabilidad gaussiana, pero tienen como diferencia que las funciones base son comunes entre todos los modelos, como es el caso de los modelos discretos que usan un *codebook* común. Un modelo semi-continuo se define con los pesos asociados a cada una de las funciones base.

1.2.2.2 Modelo Acústico

El modelo acústico cumple la función de filtro que contiene las distintas variaciones de una lengua, de acuerdo a una unidad de reconocimiento previamente definida (palabras, fonemas, etc). Dependiendo de las técnicas y del alcance del modelo y de una señal acústica de entrada se obtiene la hipótesis de lo que ha dicho el hablante.

El modelo acústico captura las características acústicas de una señal de entrada, de ésta se obtienen un conjunto de vectores llamados vectores característicos o vectores de observación, estos vectores son comparados con un conjunto de patrones que representan símbolos de un alfabeto y retorna aquellos símbolos que más se parecen. Este es el proceso usado en los modelos ocultos de Markov (Pérez, 2006).

Los modelos acústicos contienen:

- Un análisis acústico, el cual comprende la representación de una señal acústica en un conjunto de vectores característicos, para este análisis se puede hacer uso de distintas técnicas de filtrado como pueden ser LPC o MFCC
- Los modelo acústicos para cada una de las unidades de reconocimiento utilizadas

Uno de los problemas principales en la construcción de modelos acústicos es la definición de las unidades de representación del habla. Para esto, se pueden usar principalmente dos enfoques, los cuales van a depender del alcance del sistema (Ravishankar, 1996).

Para el caso de sistemas con vocabularios pequeños que tenga decenas de palabras, es posible construir modelos para cada una de las palabras del vocabulario. El problema con este enfoque es que se vuelve insostenible a medida que crece el vocabulario, ya que se vuelve sumamente difícil la obtención de datos de entrenamiento lo suficientemente representativos para la construcción de los modelos individuales.

El otro enfoque es representar las unidades acústicas no en función de las palabras sino de sub-palabras y construir los modelo acústicos para cada una de estas unidades, esto permite que una palabra se pueda representar en términos de los modelos de sub-palabras, estas unidades son conocidas como fonemas y son las más utilizadas como unidad de reconocimiento; para el español existen aproximadamente 22 fonemas (Figura 8). La cantidad de fonemas puede variar dependiendo del lenguaje o de la región a la que pertenezcan una serie de hablantes, así hablen el mismo idioma. Un ejemplo de esto se ve en el uso de la “z” entre los españoles y los Latinoamericanos.

Un fonema modela la posición de los articuladores de la boca y del canal nasal para la producción de un sonido en particular, estos articuladores se mueven ligeramente entre diferentes sonidos, por lo que cada fonema se va ha ver influenciado por sus vecinos, sobre todo en la transición de un fonema a otro.

Fonemas	Ejemplo
/a/	mal
/b/	bueno, vino
/d/	día, cada
/e/	vez
/f/	fuerte, gafas
/g/	grande, tengo
/i/	piso
/j/	Gente, jardín, girar
/k/	cola, poco, que
/l/	lado, hablar
/m/	mano, come
/n/	no, hermano
/o/	oso
/p/	peso, sopa
/r/	pero, fruta
/s/	cena, zapato, sol
/t/	tomar, dato
/u/	luz
/ch/	chico, leche
/ll/	llamar, yo
/ñ/	niño, señor
/rr/	rico, perro

Figura 8: Fonemas del español

Esto no trae problemas en vocabularios pequeños formados por palabras diferenciables acústica y fonéticamente, pero trae inconvenientes con vocabularios muy grandes en donde la tasa de error tiende a ser mayor, dada la complejidad misma del habla, para esto la mayoría de los sistemas usan, adicionalmente de los fonemas, tri-fonemas para solucionar este problema.

Los tri-fonemas son básicamente fonemas observados con respecto a su predecesor y

sucesor, el problema que tiene este enfoque es que para un lenguaje como el español que puede tener aproximadamente 22 fonemas, existirían un total de 22^3 = 10648 tri-fonemas, de los cuales sólo una pequeña fracción son observados en el lenguaje, para esto cada modelo se coloca en un cluster o en clases de equivalencia llamada senones (Ravishankar, 1996).

1.2.3 Búsqueda

El módulo de búsqueda o reconocimiento es el encargado de, dado un vector de observaciones, dar como salida el patrón que reconoció, este patrón, bien puede ser una letra, una palabra o un grupo de palabras.

Para poder determinar correctamente cuál es la frase reconocida, el módulo de reconocimiento hace uso del modelo acústico, para la correcta correspondencia del conjunto de vectores de observación con una de las unidades de reconocimiento, junto con el diccionario y el modelo de lenguaje, descrito a continuación.

1.2.3.1 Modelo del Lenguaje

Un modelo de lenguaje restringe el número de palabras que se puedan producir en el reconocimiento, estas restricciones pueden ser de carácter sintáctico o semántico y van a permitir reducir el espacio de búsqueda (Macias, 1998).

Para reconocedores de voz con vocabularios pequeños se utilizan modelos de gramáticas restringidas, en las cuales el hablante sólo dice aquellas palabras que están dentro del contexto definido, este tipo de gramáticas son las más utilizadas para sistemas de reconocimiento en el ámbito de la telefonía, pero no son la mejor solución para sistemas de reconocimiento robustos donde la entrada es libre, como seria el caso de un reconocimiento continuo (Ravishankar, 1996).

Para los reconocedores con grandes vocabularios, se requiere del uso de modelos de lenguaje o gramáticas que establezcan los límites de lo que está permitido reconocer, lo que permite seleccionar la secuencia de palabras más adecuada o que mejor se ajuste a las hipótesis que se producen en el proceso de reconocimiento. En ausencia de límites entre palabras se producen un número mayor de hipótesis que no se adecuan al contexto en que se emplea el reconocedor.

Para tareas simples como puede ser: reconocer un número reducido y restringido de frases, se pueden usar reglas o gramáticas libres de contexto como un tipo de representación compacta o limitada, pero es un método poco práctico para tareas que tienen vocabularios muy grandes, otra opción es construir modelos de lenguaje en clases, como sería agrupar meses, ciudades, números, etc, pero no sigue siendo la mejor opción para vocabularios muy grandes o de reconocimiento continuo.

Para los vocabularios muy grandes se utilizan modelos probabilísticos, modelos de N-gramas, los cuales consisten en combinaciones de N palabras. Entre los más utilizados están los modelos de bigramas y trigramas que consisten en pares y tripletas de palabras respectivamente, con una probabilidad de ocurrencia. Estos modelos se pueden construir de forma automática con un texto de entrenamiento.

La limitante en el uso de los bigramas y trigramas está en la creación de combinaciones completas de palabras, ya que lo que se representa con estos modelos es sólo un grupo o conjunto de palabras dependientes del corpus de entrenamiento, por lo tanto se trabaja con un número finito de combinaciones, por esto pueden llegar a existir miles o millones de combinaciones para tareas que requieran de grandes vocabularios, en donde se necesita grandes requerimientos de memoria.

1.2.4 Diccionario

En el diccionario de pronunciación se especifica la secuencia de sonidos, representada a través de símbolos, que componen una palabra, estos diccionarios se construyen a partir de grandes corpus de textos, ya sea de manera automática o de manera manual, el método manual trae como ventaja la definición de una misma palabra con distintos símbolos, dependiendo de las características fonéticas del lenguaje o de los hablantes, el problema que pueden tener los diccionarios y los modelos de lenguaje muy grandes, es que no establecen restricciones claras en el reconocedor.

Una vez vista cada una de las técnicas empleadas en la construcción de un reconocedor de voz se puede decir que, para la construcción de un sistema de reconocimiento de voz será necesario: un modelo acústico, un modelo de lenguaje, la definición de las unidades de reconocimiento y del diccionario de pronunciación, como elementos principales. Para la construcción del modelo acústico, se utilizan una serie de audios, también conocidos como corpus de voces, que permiten entrenar el sistema. Estos audios pasan por una primera fase, donde se extraen las características más relevantes de cada una de las señales, ya sea usando técnicas de codificación linear predictiva o coeficientes espectrales de frecuencia Mel, estas características luego son procesadas para generar los modelos acústicos, para cada una de las unidades de reconocimiento, usando modelos ocultos de Markov, y por último, se construye el modelo de lenguaje que va a establecer el contexto de reconocimiento. Durante el proceso de decodificación se procesará una señal de audio de entrada a través de los modelos acústicos y de lenguaje generados y se seleccionará aquella hipótesis que presente una mayor probabilidad o un mayor puntaje. Esta hipótesis vendría siendo la transcripción devuelta por el sistema, la cual, se debe corresponder con la señal de audio original.

CAPÍTULO II. MARCO APLICATIVO

2.1 Planteamiento del problema

Actualmente en el área de reconocimiento de voz existen grandes avances que han permitido la reducción de la tasa de errores y la independencia del hablante y de las características del mismo. Esto permite su aplicación dentro del área de interacción humano-computador, a través de un manejo mucho más intuitivo y natural del que se puede lograr con el uso del *mouse* o el teclado, donde el uso de la voz es esencial para la interacción. Un ejemplo claro de esto son los celulares que están presentando este tipo de características para hacer más fácil y rápido su manejo. Otro beneficio que presentan las aplicaciones de reconocimiento se encuentra en el área de la telefonía, ya que permiten la interacción de una persona a través del teléfono usando comandos de voz para poder navegar a través de un sistema, acompañado en ocasiones por sistemas de reconocimiento de dígitos (DTMF – Dual-Tone Multi-Frecuency) para facilitar la interacción con el usuario.

El problema que tienen estas aplicaciones, y más para el idioma español, es la poca existencia de sistemas que hay en el mercado, donde la gran mayoría no están disponibles al público general y son justamente estos los que tienen mejores prestaciones en aplicaciones del mundo real. Otro punto en contra de los avances para el reconocimiento de voz en español, es la falta de corpus de entrenamiento que permitan hacer sistemas lo suficientemente robustos a bajo costo.

Por esto resultaría de gran utilidad una aplicación de reconocimiento de voz de código libre que otorgue buenas prestaciones en aplicaciones del mundo real, específicamente en el mundo de la telefonía.

2.2 Solución Propuesta

Tomando en cuenta la situación actual en el ámbito del reconocimiento de voz para el español, se propone la construcción de un prototipo de reconocedor de palabras aisladas, usando las bondades que pueden aportar los modelos ocultos de Markov, junto con corpus de audios que se puedan encontrar en Internet sin costo alguno y haciendo uso de plataformas *open source* tanto para el entrenamiento de los modelos como para el reconocimiento de la señal de voz.

Para este sistema, dado que la cantidad de corpus de audios que se puedan encontrar en Internet es limitada, se utilizará el enfoque de los modelos semi-continuos de Markov que garantizan que con pocas horas de grabación se puedan obtener buenos resultados.

Este proyecto será elaborado con el aporte de algunos corpus de audio por parte de Merlin Telecom (http://www.merlin-telecom.com/) y de bases de datos de audio disponibles en la Web. Se va a realizar con miras hacia el uso en aplicaciones telefónicas, por lo que se tiene que tomar en consideración que los modelos serán construidos con audios de 8000 Hz que es la frecuencia usada en el canal telefónico.

Se usará el proyecto *Sphinx* de la *Carnegie Mellon University* (CMU) para la construcción de sistemas de reconocimiento, por estar considerado como uno de los mejores paquetes de software para el desarrollo de aplicaciones de reconocimiento de voz, que continua en desarrollo para la mejora de los algoritmos usados durante los entrenamientos y para el reconocimiento. Esto garantiza que la plataforma tiene una estructura bien definida, en constante desarrollo, sobre la que se obtendrán buenos resultados. Entre los paquetes a usar están:

- **Sphinx Font-End:** para la extracción de características

- **Sphinx-Train:** para el entrenamiento de los modelos ocultos de Markov
- **PocketSphinx:** para la decodificación de la señal de audio

2.3 Objetivos

2.3.1 Objetivo General

Construir un sistema de reconocimiento de voz de palabras aisladas para su uso en sistemas de telefonía, usando técnicas de modelos ocultos de Markov para el entrenamiento de los modelos.

2.3.2 Objetivos Específicos

- Recolección y transcripción del corpus de voces para la fase de entrenamiento
- Definir las unidades fonéticas que se van a usar para el entrenamiento de los modelos
- Construir el diccionario de pronunciación
- Realizar la conversión del corpus de audio
- Extraer los coeficientes espectrales Mel para la posterior construcción de los vectores de características, utilizando el paquete Sphinx Front-End
- Construir los modelos acústicos usando el paquete Sphinx-Train, que utiliza técnicas de modelos ocultos de Markov
- Construir los modelos de lenguaje para las pruebas del sistema
- Realizar las pruebas para verificar la efectividad del sistema utilizando el paquete PocketSphinx, y realizar los ajustes necesarios para mejorar el reconocimiento
- Realizar integración para el uso de los modelos en una aplicación de telefonía

2.4 Desarrollo del sistema de reconocimiento

El desarrollo del sistema de reconocimiento de voz, se fundamenta en la arquitectura mostrada en la Figura 9, donde se observan las entradas y salidas de cada uno de los módulos utilizados.

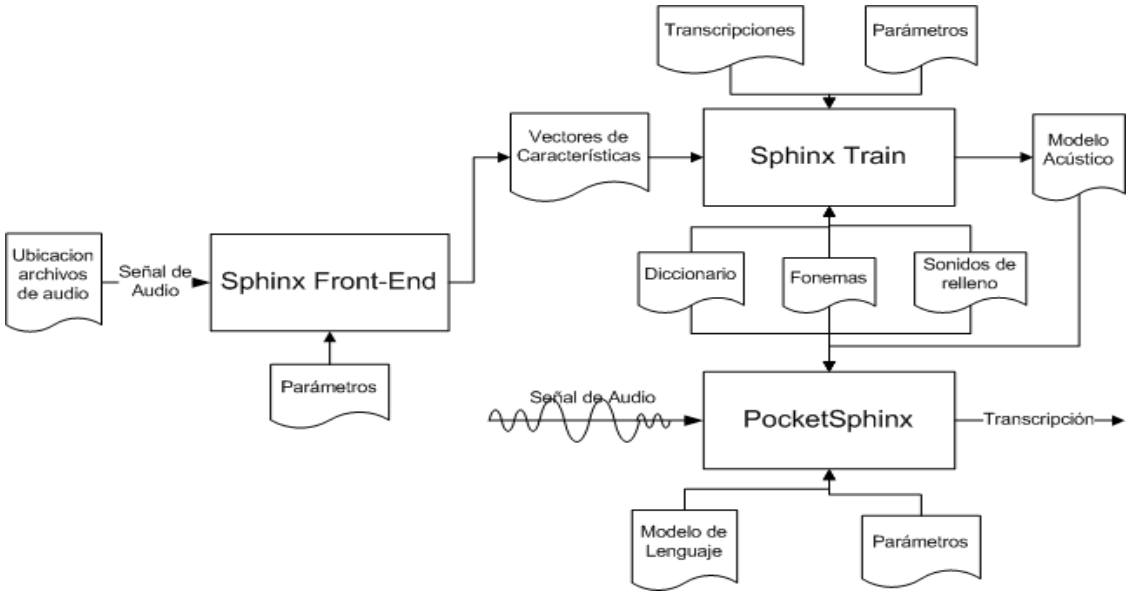

Figura 9: Arquitectura de un reconocedor de voz usando los módulos de *Sphinx*

A continuación se presentan cada una de las etapas de desarrollo, considerando los siguientes aspectos:

- Recolección del corpus de voces
- Extracción de características
- Entrenamiento
- Decodificación

2.4.1 Recolección del corpus de voces

Para la recolección del corpus de voces usado para el entrenamiento del sistema, se utilizaron repositorios de audios disponibles a través de Internet, junto con una muestra provista por Merlin Telecom.

El total de horas del corpus de voces es de 12 horas y 26 minutos, para una colección de 6936 muestras de audio, distribuidas de la siguiente manera:

- 4583 archivos, con un total de 9 horas del sitio http://www.voxforge.org/es/listen
- 1469 archivos, con un total de 1 hora y 37 minutos del sitio http://lab.chass.utoronto.ca/rescentre/spanish/
- 884 archivos, con un total de 1 hora y 49 minutos de Merlin Telecom

Para hacer uso de estos audios en los entrenamientos fue necesaria una reconversión, para cada uno, a una frecuencia de 8000 Hz, con una compresión en formato PCM a 16 bits, usando un solo canal (mono), características con las que se van a entrenar los modelos acústicos.

2.4.2 Extracción de Características

Para la extracción de características se usó el módulo de Sphinx Front-End, el cual realiza la conversión de los audios de entrada a archivos que contienen la información cepstral de cada uno, usando el formato utilizado por el módulo de entrenamiento Sphinx-Train.

En este módulo se realizan las siguientes tareas (Figura 10) para obtener los vectores de características:

- Realizar el proceso de pre-énfasis a la señal de entrada de audio
- Aplicar la ventana de Hamming a cada uno de los fragmentos de la señal de audio
- Extraer los Power Spectrum de cada una de las ventanas de Hamming
- Extraer los Mel Spectrum, multiplicando los power spectrum por un filtro en la escala Mel
- Extraer los Mel Cepstrum, que son el resultado de aplicar la transformada discreta de

Fourier al logaritmo de los Mel spectrum

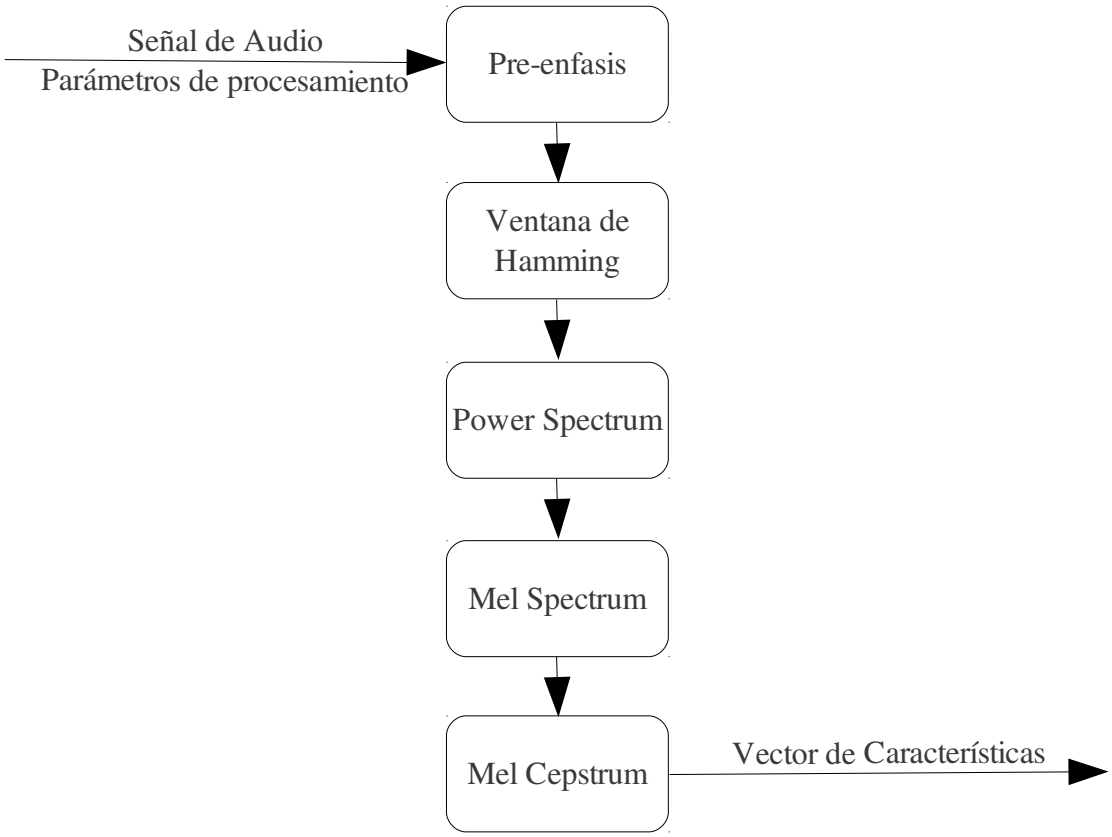

Figura 10: Diagrama de flujo del Sphinx FrontEnd

Adicionalmente, este módulo requiere como entrada la ubicación de cada uno de los archivos de audios, dado un directorio base. La arquitectura global de este módulo es la que se puede observar en la Figura 11.

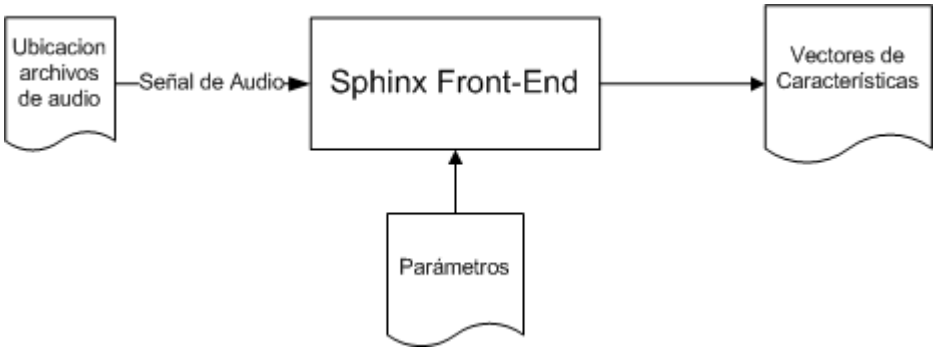

Figura 11: Arquitectura global de Sphinx Front-End

Durante este proceso se obtienen una serie de archivos, uno por muestra de audio, con las características de cada señal de audio, los cuales van a ser utilizados posteriormente para la fase de entrenamiento.

2.4.3 Entrenamiento

Durante la fase de entrenamiento se utilizó el módulo de Sphinx-Train, el cual permite construir los modelos ocultos de Markov. Para esta fase se construyeron cuatro modelos acústicos con las siguientes premisas:

- Un modelo acústico semi-continuo usando 256 muestras para la transformada de Fourier
- Un modelo acústico semi-continuo usando 512 muestras para la transformada de Fourier
- Un modelo acústico continuo usando 256 muestras para la transformada de Fourier
- Un modelo acústico continuo usando 512 muestras para la transformada de Fourier
- Los modelos semi-continuos fueron entrenados con 1000 senones
- Los modelos continuos fueron entrenados con 3000 senones
- Cada modelo acústico va a tener en común el tamaño de la ventana de Hamming, con un valor de 256 milisegundos y la frecuencia con la que se entrenan los modelos que es de 8000 Hz para cada muestra de audio.

La construcción de cuatro modelos fue debido a que no se tenia disponible un modelo de control para las pruebas, por lo tanto, se tomó la decisión de construir cuatro modelos, con características diferentes y luego de realizadas las pruebas individuales, y en base a los resultados obtenidos, determinar cuál es el mejor modelo acústico, en base a las tasas de reconocimiento y de error.

Durante el proceso de extracción de características se realizaron cuatro iteraciones,

usando para cada una el mismo corpus de audio, pero variando el tamaño de las muestras para la transformada de Fourier, esto para generar los vectores de acuerdo a las características de los modelos que se plantearon obtener para realizar las pruebas individuales.

El valor del número de senones fue determinado por las mismas recomendaciones de la aplicación y por la siguiente tabla

Hora de data para entrenamiento	**Número de senones**
1–3	500–1000
4–6	1000–2500
6–8	2500–4000
8–10	4000–5000
10–30	5000–5500
30–60	5500–6000
60–100	6000–8000
Mayor a 100	8000

Figura 12: Número de senones por horas de audio

Este módulo presenta la arquitectura mostrada en la Figura 13, cuyas entradas son:

- Archivo de transcripciones
- Sonidos de relleno
- Fonemas
- Diccionario de pronunciación
- Ubicación de los archivos con las características por señal de audio
- Parámetros del modelo, este contiene los parámetros con los que se realizó la extracción de características y parámetros adicionales para el entrenamiento, como lo son: número de senones, tipo de modelo (continuo, semi-continuo) número de estados para el modelo de Markov, entre otros.

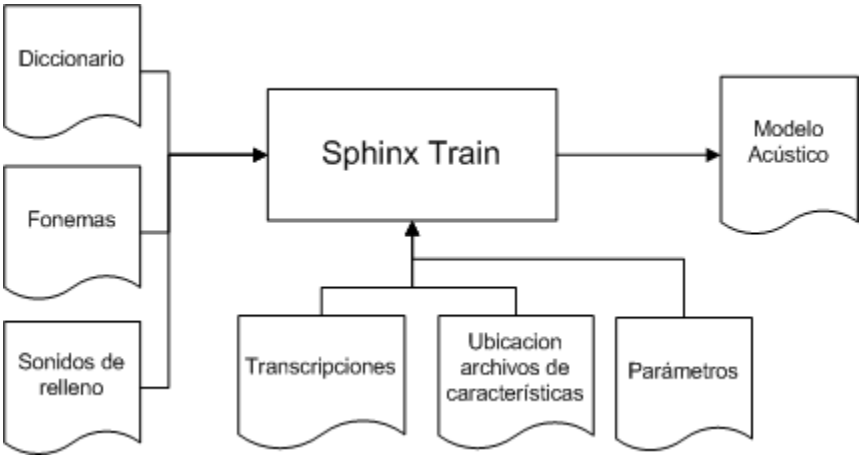

Figura 13: Arquitectura global de Sphinx-Train

Cada una de las entradas del módulo se construyo siguiendo el formato requerido por el mismo. Las entradas más importantes, como son: el archivo de transcripción, los sonidos de relleno, los fonemas y el diccionario de pronunciación, van a ser descritos a continuación, junto con un ejemplo de la definición de cada uno.

- **Archivo de transcripciones**

Con ayuda del corpus de voces se realizaron las transcripciones para las muestras de audio del sitio http://lab.chass.utoronto.ca/rescentre/spanish/ y para las muestras de Merlin Telecom, ya que son las únicas muestras del corpus de los cuales no se tenían transcripciones, para el caso de el corpus de http://www.voxforge.org/es/listen, cada uno de los audios viene acompañado de su correspondiente transcripción. Estás transcripciones luego se pasaron al formato necesario por Sphinx-Train el cual es el siguiente:

```
<s> texto ++sonidos de relleno++ texto </s> (nombre del archivo de audio)
```

Las etiquetas <s> y </s> indican el inicio y fin respectivamente de la transcripción, el nombre del archivo por convención no contiene la extensión del mismo.

```
<s> DOS SALONES AGRANDADOS POR LA ESCASEZ DE SU ALTURA ERAN EL
CAMPO VISUAL DE OJEDA </s> (es-0011)
<s> EN EL PRIMERO DONDE ESTABA EL MEZCLABASE A LA BLANCURA
UNIFORME DE LA DECORACION EL VERDE CHAROLADO </s> (es-0012)
<s> LAS PALMERAS DE INVERNACULO EL VERDE PICTORICO DE LOS
ENREJADOS </s> (es-0013)
<s> PUEDE GRABAR SU QUEJA DESPUES DEL TONO ++BIP++ </s>
(QUEJA)
```

- **Sonidos de relleno**

Los sonidos de relleno son usados para la definición de ruido que pueda existir dentro de la señal de audio y que no sean propias de ésta.

El archivo utilizado para la definición de los sonidos de relleno es el siguiente:

```
</s>            SIL
<s>             SIL
<sil>           SIL
++BIP++         +BIP+
++LALALALA++    +LALALALA+
++NOISE++       +NOISE+
```

Donde <s>, </s> y <sil> siempre están presentes ya que representan el inicio, el fin y los silencios dentro de la transcripción, cualquier otra etiqueta que se agregue es definida de acuerdo a los ruidos presentes en el corpus de voces.

- **Fonemas**

Para el caso de las unidades fonéticas, luego de una revisión de distintos sitios y documentación (http://liceu.uab.es/~joaquim/publicacions/SAMPA_Spanish_93.pdf) se usaron los siguientes fonemas, incluyendo los sonidos de relleno:

```
+BIP+
+LALALALA+
+NOISE+
SIL
A
B
CH
D
E
F
G
GN
I
J
K
L
LL
M
N
O
P
R
RR
S
T
U
V
X
Y
```

- **Diccionario**

El diccionario de pronunciación utilizado fue tomado del siguiente sitio

http://www.speech.cs.cmu.edu/sphinx/models/hub4spanish_itesm/proyectos/h4.dict y fue usado como base, teniendo que agregar todas aquellas palabras que no se encontraran definidas, junto con su definición fonética correspondiente.

El diccionario de pronunciación tiene el siguiente formato y lo recomendable es que el mismo esté ordenado de forma ascendente.

```
ACLARADOS          A K L A R A D O S
ACLARAMOS          A K L A R A M O S
ACLARAN            A K L A R A N
ACLARANDO          A K L A R A N D O
ACLARAR            A K L A R A R
ACLARARLE          A K L A R A R L E
```

Una vez definidos el diccionario, el archivo de transcripciones, los sonidos de relleno, los fonemas y obtenido el corpus de voces, se procedió a realizar el entrenamiento de los distintos modelos con Sphinx-Train.

El módulo de Sphinx-Train realiza las siguientes tareas (Figura 14) para la definición de los modelos acústicos:

- Clasificación de los vectores de características usando la técnica de Cuantificación Vectorial
- Entrenamiento de los modelos independientes del contexto para cada uno de los fonemas usados como unidades de reconocimiento
- Entrenamiento de los modelos dependientes del contexto
- Creación de los árboles de decisión
- Entrenamiento de los modelos dependientes del contexto

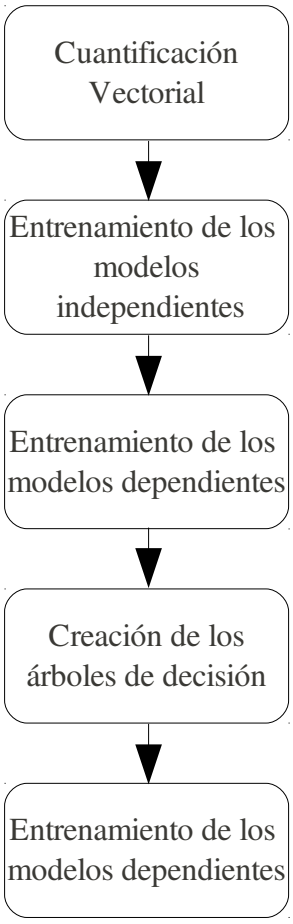

Figura 14: Diagrama de flujo de Sphinx-Train

2.4.4 Decodificación

El proceso de decodificación se realizó utilizando el módulo de PocketSphinx, el cual realiza la codificación de la señal de audio de entrada y la evalúa con los modelos acústicos, creados en la fase de entrenamiento, para obtener un resultado, que en este caso sería una transcripción del audio de entrada.

PocketSphinx presenta la arquitectura global de la Figura 15, donde las entradas del sistema vienen dadas por:

- Diccionario de pronunciación

- Fonemas
- Sonidos de relleno
- Modelo de lenguaje
- Modelo acústico
- Parámetros adicionales

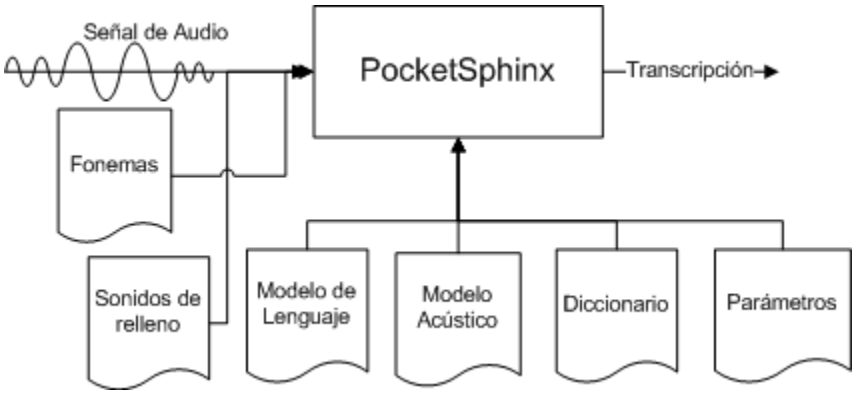

Figura 15: Arquitectura global de PocketSphinx

El formato usado para el archivo con la ubicación de los audios es el mostrado a continuación:

```
test_sphinx/muestras_txn_100/ASRGfLtDB
test_sphinx/muestras_txn_100/ASRhiKFiG
test_sphinx/muestras_txn_100/ASRJZolAN
test_sphinx/muestras_txn_100/ASRfVyKsA
test_sphinx/muestras_txn_100/ASRQ3bOqf
test_sphinx/muestras_txn_100/ASRbGYvKB
test_sphinx/muestras_txn_100/ASR6pIdgF
```

Al igual que en el archivo de transcripciones se omite la extensión del archivo de audio

El formato de salida de este módulo luego de haber procesado la señal es el siguiente:

```
bloquear tarjeta de (test_sphinx/muestras_txn_100/ASRGfLtDB -21893)
saldos (test_sphinx/muestras_txn_100/ASRhiKFiG -10393)
protector tarjeta de debito (test_sphinx/muestras_txn_100/ASRJZolAN
-10781)
 (test_sphinx/muestras_txn_100/ASRfVyKsA -10781)
 (test_sphinx/muestras_txn_100/ASRQ3bOqf -10781)
 (test_sphinx/muestras_txn_100/ASRbGYvKB -10781)
 (test_sphinx/muestras_txn_100/ASR6pIdgF -10781)
```

- **Modelo de Lenguaje**

Para la decodificación se definieron dos modelos de lenguaje usando el formato JSGF (http://java.sun.com/products/java-media/speech/forDevelopers/JSGF/JSGF.html), la primera gramática, denominada txn_100 esta definida como:

```
#JSGF V1.0;
grammar txn_100;
public <Menu> = <saldo> | <reclamos> | <protector> |
<actualizar> | <informacion> | <bloqueo>;
<saldo> = consultar mi saldo | consultar el saldo | saldo |
saldos | consultar saldo | consulta de saldo;
<reclamos> = reclamo | reclamos | realizar reclamo | realizar
un reclamo;
<protector> = protector de tarjeta de debito | protector de
tarjeta | protector | protector tarjeta de debito;
<actualizar> = actualizar telefono celular | actualizacion de
telefono celular | actualizacion | actualizar | actualizar
telefono | actualizacion de telefono | actualizar mi telefono
| actualizar mi telefono celular | actualizar mi celular;
<informacion> = informacion | informacion de productos o
servicios | informacion de productos y servicios | informacion
de productos | informacion de producto;
```

```
<bloqueo> = bloquear tarjeta de debito | bloquear tarjeta
debito;
```

y la segunda gramática denominada confirmenu como:

```
#JSGF V1.0;
grammar ConfirMenu;
public <ConfirMenu> = <si> | <no>;
<si> = si;
<no> = no;
```

Los resultados obtenidos en el módulo de decodificación serán presentados a continuación, junto con los criterios utilizados para la evaluación de los modelos.

2.5 Experimento y Resultados

El proceso de decodificación utilizado durante las pruebas se realizó en lotes, donde se pasa como parámetro un archivo que contiene la ubicación de los audios que se van a evaluar y se genera un archivo con las transcripciones para cada audio. Este archivo luego fue evaluado de acuerdo a ciertos criterios definidos, para obtener los resultados que serán presentados más adelante.

Para la evaluación de cada uno de los modelos se definieron:

- Criterios de evaluación
- Ajuste de parámetros
- Corpus de entrenamiento

- **Criterios de evaluación**

Se utilizaron los siguientes criterios de evaluación para los modelos:

- Evaluación de los resultados obtenidos por intención del hablante, comparando la

transcripción resultante con lo que está queriendo decir la persona

- Evaluación por número de palabras reconocidas correctamente

Cada criterio de evaluación, se dividió en dos fases:

- Una primera fase en donde se evalúo cada modelo, usando el módulo de PocketSphinx, sin hacer ninguna modificación a los parámetros que vienen por defecto, para así tener una base de cómo es el reconocimiento para cada modelo generado.
- Una segunda fase en donde se hicieron las modificaciones individuales de los parámetros, de forma tal, que se ajusten mejor a cada uno de los modelos generados.

- **Ajuste de parámetros**

El ajuste de los parámetros, para cada modelo, se realizó por ensayo y error, haciendo énfasis principalmente en los valores de:

- beam: delimitador para el número de HMM activos durante el proceso de búsqueda
- pbeam: delimitador para el número de fonemas que se van a mantener activos durante el proceso de búsqueda
- wbeam: delimitador para el número de palabras que se van a mantener activas durante el proceso de búsqueda
- pip: valor de penalidad por insertar un fonema
- wip: valor de penalidad para insertar una palabra
- topn: número máximo de gaussianos para la calificación
- lw: peso probabilístico para el modelo de lenguaje

y evaluando en cada cambio las disminución de la tasa de error, hasta que no se observaran mejoras en los valores de reconocimiento de palabras.

Los siguientes son los parámetros utilizados para cada una de las fases:

- Fase 1: los cuatro modelos utilizaron los mismos parámetros

Parámetros	Valores
beam	1e-48
pbeam	1e-48
wbeam	7e-29
pip	1
wip	0,65
topn	4
lw	6,5

- Fase 2: ajustes de parámetros por modelo

	semi_256	semi_512	cont_256	cont_512
beam	1e-80	1e-60	0	1e-80
pbeam	1e-62	1e-40	0	1e-40
wbeam	7e-29	1e-40	7e-29	1e-70
pip	0,6	0,6	0,1	0,1
wip	0,65	0,4	0,65	0,65
topn	1	1	7	7
lw	10	10	7	7

- **Corpus de entrenamiento**

El corpus utilizado para las pruebas fue provisto por Merlin Telecom y cuenta con un total de 263 audios. Éste corpus pertenece al contexto de una aplicación telefónica, donde 157 de los audios corresponden a opciones de un menú de servicios bancarios, al cual acepta las opciones definidas en la gramática txn_100. Los 106 audios restantes son respuestas de si y no que utilizan la gramática confirmenu. El corpus también incluye silencios (audios vacíos), ruido de fondo y palabras fuera de contexto.

Las características del corpus son las siguientes:

- 8000 Hz de frecuencia
- No tienen cabecera (sin formato)
- Un solo canal (mono)
- Compresión en formato PCM

Los resultados obtenidos para cada una de las fases de evaluación serán presentados a continuación, separados de acuerdo a los criterios de evaluación, haciendo las comparaciones entre cada fase, y el resultado del muestreo individual para cada fase, separado de acuerdo a los dos criterios de evaluación utilizados.

2.5.1 Resultados por intención del hablante

Por notación los modelos acústicos se van a denotar de la siguiente manera:

- semi_256: Modelo semi-continuo, 256 muestras para la transformada de Fourier
- semi_512: Modelo semi-continuo, 512 muestras para la transformada de Fourier
- cont_256: Modelo continuo, 256 muestras para la transformada de Fourier
- cont_512: Modelo continuo, 512 muestras para la transformada de Fourier

Para la evaluación de los resultados por intención del hablante se utilizaron los siguientes criterios:

- Palabras correctas: número de frases correctas, de acuerdo a la intención del hablante; número de frases que no fueron reconocidas porque no existen dentro de la gramática; número de silencios o ruidos detectado correctamente (sin reconocimiento)
- Errores: número de frases que no se corresponden con la intención del hablante; ruido, silencio o palabras que no pertenezcan a la gramática y que devuelven algún reconocimiento

- %Correcto: $\frac{número\ correctas * 100}{número\ correctas + número\ errores}$
- %Error: $\frac{número\ errores * 100}{número\ correctas + número\ errores}$

En las siguientes tablas, se observarán los resultados obtenidos en la primera fase de las pruebas para evaluar la calidad de los modelos sin modificaciones en los parámetros, de cada una de las gramáticas, y resultados generales.

Tabla 1: Resultados para la gramática txn_100:

Modelo	Correctas	Errores	%Correcto	%Error
semi_256	116	41	73,89%	26,11%
semi_512	110	47	70,06%	29,94%
cont_256	118	39	75,16%	24,84%
cont_512	117	40	74,52%	25,48%

Tabla 2: Resultados para la gramática confirmenu

Modelo	Correctas	Errores	%Correcto	%Error
semi_256	87	19	82,08%	17,92%
semi_512	86	20	81,13%	18,87%
cont_256	67	39	63,21%	36,79%
cont_512	66	40	62,26%	37,74%

Tabla 3: Resultados totales para ambas gramáticas

Modelo	Correctas	Errores	%Correcto	%Error
semi_256	203	60	77,19%	22,81%
semi_512	196	67	74,52%	25,48%
cont_256	185	78	70,34%	29,66%
cont_512	183	80	69,58%	30,42%

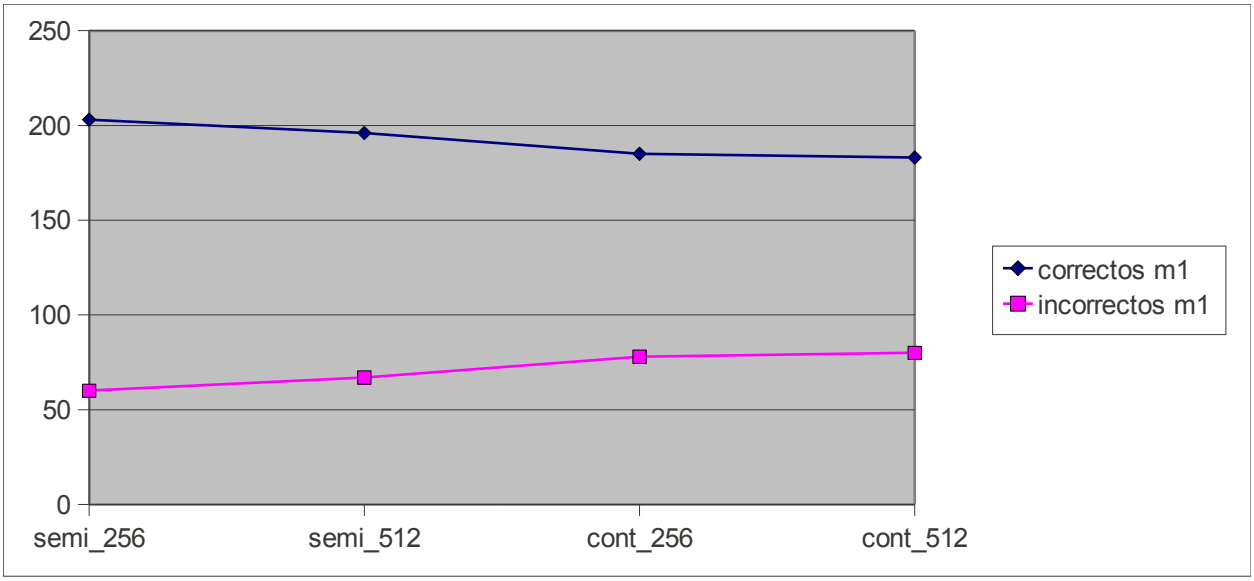

Figura 16: Comportamiento de los modelos con respecto al número de reconocimientos correctos e incorrectos, para la primera fase

Como se puede observar en la Figura 16 y según los resultados obtenidos luego de las pruebas de la primera fase, la tasa de error se encuentra entre el 22 y el 30 por ciento, donde los mejores resultados, con ambas gramáticas, son las obtenidas por el modelo semi_256, el cual presenta un porcentaje de reconocimiento correcto, por intención, del 77,19% con un error del 22,81%.

En las siguientes tablas, se observarán los resultados de la segunda fase de las pruebas para evaluar la calidad de los modelos luego de los ajustes individuales de parámetros.

Tabla 4: Resultados para la gramática txn_100

Modelo	Correctas	Errores	%Correcto	%Error
semi_256	142	15	90,45%	9,55%
semi_512	136	21	86,62%	13,38%
cont_256	143	14	91,08%	8,92%
cont_512	139	18	88,54%	11,46%

Tabla 5: Resultados para la gramática confirmenu

Modelo	Correctas	Errores	%Correcto	%Error
semi_256	99	7	93,40%	6,60%
semi_512	100	6	94,34%	5,66%
cont_256	71	35	66,98%	33,02%
cont_512	83	23	78,30%	21,70%

Tabla 6: Resultados totales para ambas gramáticas

Modelo	Correctas	Errores	%Correcto	%Error
semi_256	241	22	91,63%	8,37%
semi_512	236	27	89,73%	10,27%
cont_256	214	49	81,37%	18,63%
cont_512	222	41	84,41%	15,59%

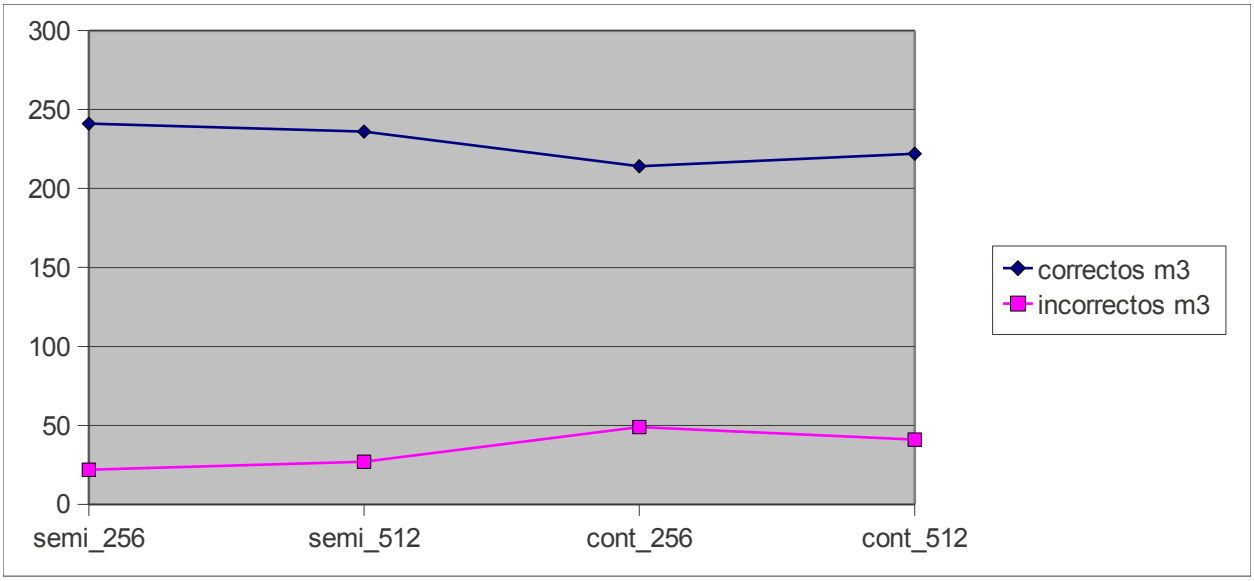

Figura 17: Comportamiento de los modelos con respecto al número de reconocimientos correctos e incorrectos, para la segunda fase

Los resultados para la segunda fase tienen una tasa de error entre el 8% y el 16%, donde, al igual que en la primera fase, el modelo que mejores resultados obtuvo es el semi_256 con un porcentaje de reconocimiento, por intención, del 91,63% y un porcentaje de error del 8,37%

Resultados globales de las dos fases:

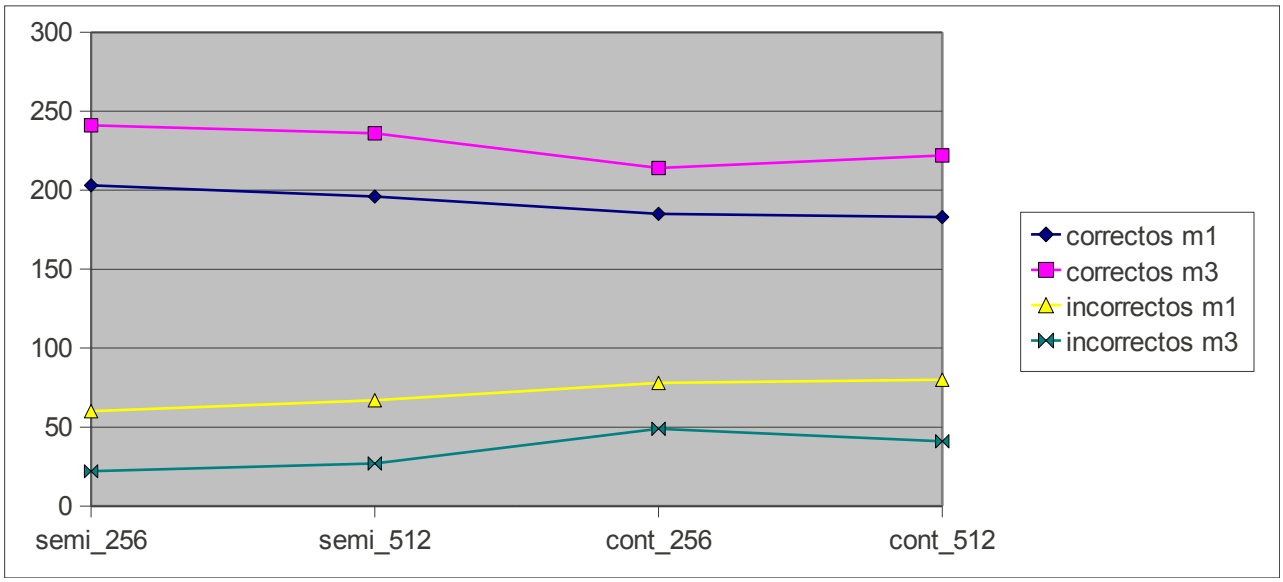

Figura 18: Resultados obtenidos en ambas fases con respecto al número de reconocimientos correctos e incorrectos

Comparando los resultados de ambas fases (Figura 18) se puede observar que el reconocimiento de los modelos, luego del ajuste de parámetros, mejoró para el mejor de los casos de 74,52% de frases, reconocidas por intención, a un 89,73%. Para el caso concreto del modelo que obtuvo los mejores resultados (semi_256) se puede observar una mejora en el reconocimiento de 91,63% de frases reconocidas con una tasa de error del 8,37%.

2.5.2 Resultados por número de palabras reconocidas correctamente

Para la evaluación de los resultados por número de palabras correctamente reconocidas se utilizaron los siguientes criterios:

- Reconocidas: número total de palabras reconocidas
- Correctas: número total de palabras reconocidas correctamente, esto incluye su ubicación dentro de la oración. El número de palabras correctas también es igual al número total de palabras menos el número de errores
- Errores: número de palabras mal reconocidas; número de palabras reconocidas cuando no debería existir reconocimiento; número de palabras omitidas en el reconocimiento
- %Reconocimiento: $\frac{palabras\ reconocidas * 100}{total\ de\ palabras}$
- %Error: $\frac{número\ errores * 100}{total\ de\ palabras}$
- %Efectividad: $\frac{palabras\ correctas * 100}{total\ de\ palabras}$

El número total de palabras para el corpus de prueba de 263 audios es de 420. Distribuidos en 318 palabras para el corpus evaluado con la gramática txn_100 y de 102 palabras para el corpus de la gramática confirmenu

En las siguientes tablas, se observarán los resultados obtenidos en la primera fase de las pruebas para evaluar la calidad de los modelos sin modificaciones en los parámetros, de cada una de las gramáticas, y resultados generales.

Tabla 7: Resultados para la gramática txn_100

Modelo	Reconocidas	Correctas	Errores	%Reconocimiento	%Error	%Efectividad
semi_256	224	190	128	70,44%	40,25%	59,75%
semi_512	204	180	138	64,15%	43,40%	56,60%
cont_256	218	177	141	68,55%	44,34%	55,66%
cont_512	222	189	129	69,81%	40,57%	59,43%

Tabla 8: Resultados para la gramática confirmenu

Modelo	Reconocidas	Correctas	Errores	%Reconocimiento	%Error	%Efectividad
semi_256	87	82	20	85,29%	19,61%	80,39%
semi_512	86	81	21	84,31%	20,59%	79,41%
cont_256	67	62	40	65,69%	39,22%	60,78%
cont_512	66	61	41	64,71%	40,20%	59,80%

Tabla 9: Resultados totales para ambas gramáticas

Modelo	Reconocidas	Correctas	Errores	%Reconocimiento	%Error	%Efectividad
semi_256	311	272	148	74,05%	35,24%	64,76%
semi_512	290	261	159	69,05%	37,86%	62,14%
cont_256	285	239	181	67,86%	43,10%	56,90%
cont_512	288	250	170	68,57%	40,48%	59,52%

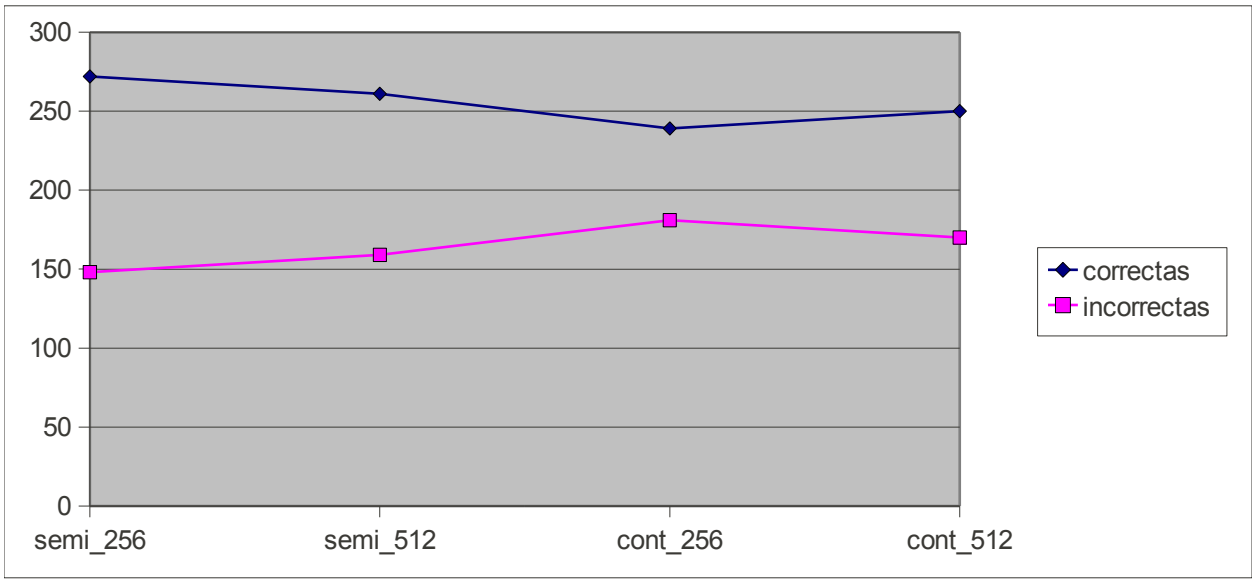

Figura 19: Comportamiento de los modelos con respecto al número de reconocimientos correctos e incorrectos, para la primera fase

Según los resultados obtenidos luego de las pruebas de la primera fase, la tasa de error se encuentra entre el 35 y el 41 por ciento, donde los mejores resultados, con ambas gramáticas, son las obtenidas por el modelo semi_256, el cual presenta un porcentaje de efectividad, para el reconocimiento de palabras correctas, de un 64,76% con una tasa de error del 35.24%.

En las siguientes tablas, se observarán los resultados de la segunda fase de las pruebas para evaluar la calidad de los modelos luego de los ajustes individuales de parámetros.

Tabla 10: Resultados para la gramática txn_100

Modelo	Reconocidas	Correctas	Errores	%Reconocimiento	%Error	%Efectividad
semi_256	267	255	63	83,96%	19,81%	80,19%
semi_512	241	232	86	75,79%	27,04%	72,96%
cont_256	255	248	70	80,19%	22,01%	77,99%
cont_512	250	240	78	78,62%	24,53%	75,47%

Tabla 11: Resultados para la gramática confirmenu

Modelo	Reconocidas	Correctas	Errores	%Reconocimiento	%Error	%Efectividad
semi_256	99	94	8	97,06%	7,84%	92,16%
semi_512	97	95	7	95,10%	6,86%	93,14%
cont_256	71	66	36	69,61%	35,29%	64,71%
cont_512	82	78	24	80,39%	23,53%	76,47%

Tabla 12: Resultados totales para ambas gramáticas

Modelo	Reconocidas	Correctas	Errores	%Reconocimiento	%Error	%Efectividad
semi_256	366	349	71	87,14%	16,90%	83,10%
semi_512	338	327	93	80,48%	22,14%	77,86%
cont_256	326	314	106	77,62%	25,24%	74,76%
cont_512	332	318	102	79,05%	24,29%	75,71%

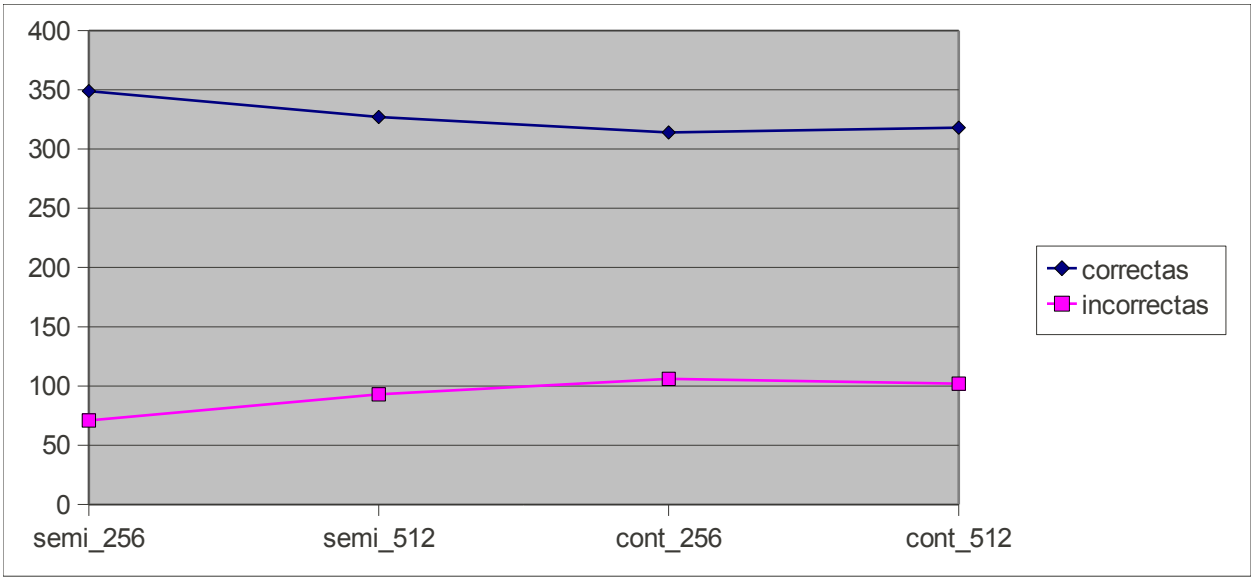

Figura 20: Comportamiento de los modelos con respecto al número de reconocimientos correctos e incorrectos, para la segunda fase

Los resultados para la segunda fase tienen una tasa de error entre el 16 y el 25 por ciento, donde, el modelo que mejores resultados obtuvo es el semi_256 con un porcentaje de efectividad, para el reconocimiento de palabras, del 83,10% y un porcentaje de error del 16,90%.

Resultados globales de las dos fases:

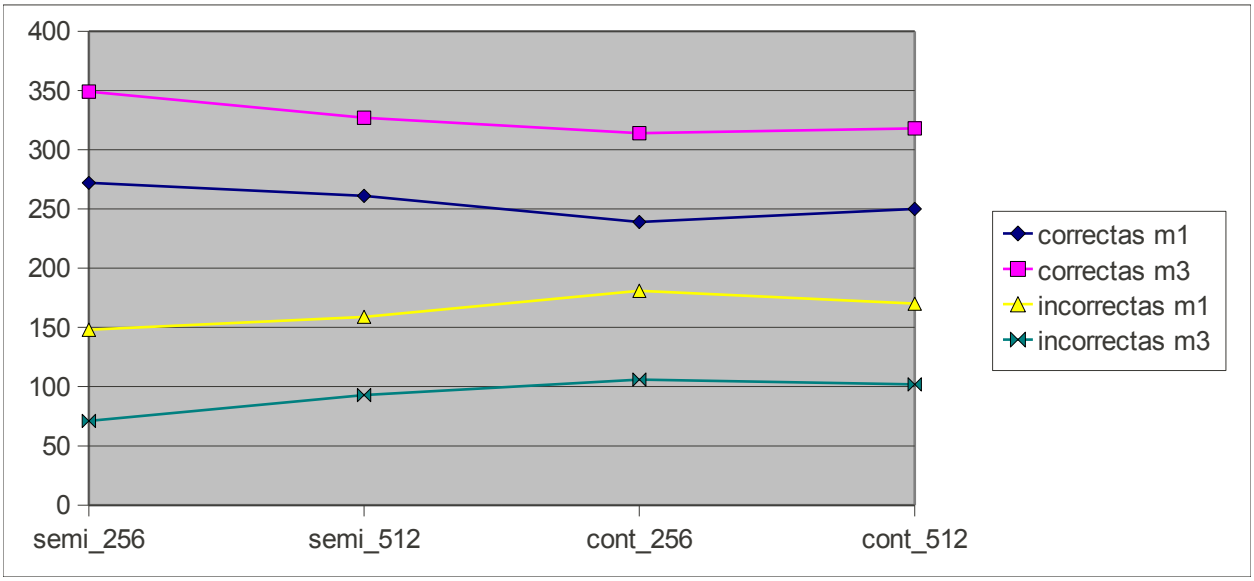

Figura 21: Resultados obtenidos en ambas fases con
respecto al número de reconocimientos correctos e incorrectos

Comparando los resultados de ambas fases (Figura 21) se puede observar que el reconocimiento de los modelos mejoró luego del ajuste de parámetros, donde el modelo semi_256, que proporciona los mejores resultados, tiene una tasa de reconocimiento efectivo del 83,10% con una tasa de error de 16,90% por palabras.

De los resultados obtenidos se puede observar que luego de los ajustes, los modelos generados tienen una buena tasa de reconocimiento y una tasa de error aceptable, donde los modelos semi-continuos fueron los que ofrecieron mejores resultados, a diferencia de los modelos continuos. Entre los modelos semi-continuo destaca particularmente el modelo con 256 muestras para la transformada de Fourier, del que se obtuvieron las mejores tasas de reconocimiento para los dos criterios de evaluación utilizados.

Uno de los principales factores que afectó la tasa de reconocimiento efectivo se encuentra en el origen del corpus de audios de prueba, el cual contiene grabaciones saturadas, conversaciones, palabras fuera del contexto del modelo de lenguaje, ruido de fondo, entre otros, que a diferencia de los audios utilizados para el entrenamiento de los modelos acústicos, no contenían ninguna de esas características, en su gran mayoría eran grabaciones hechas en ambientes sin ruidos de fondo o grabaciones hechas en estudio.

2.6 Desarrollo de prototipo de aplicación

Para la construcción del prototipo de reconocimiento de voz se definieron dos módulos, como se puede observar en la Figura 22.

- Aplicación
- Sistema de reconocimiento automático de voz

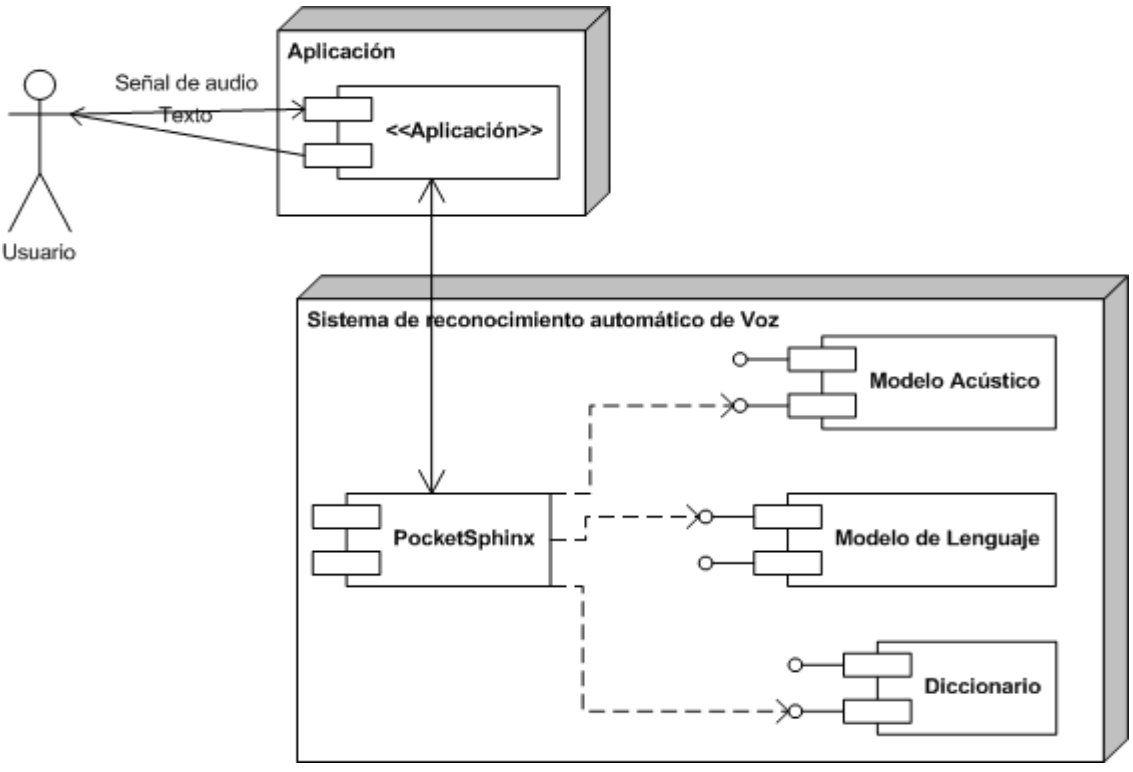

Figura 22: Arquitectura del prototipo de se sistema de reconocimiento

El modulo de aplicación, para este caso en particular, y como el uso de los modelos acústicos está enfocado al uso en sistemas de telefonía, está constituido por una plataforma de software libre, conocida como Asterisk, que cumple las funciones de una central telefónica (PBX). En ésta se desarrolló una aplicación de IVR (Respuesta de Voz Interactiva) que hiciese uso del motor de reconocimiento de Sphinx (PocketSphinx) para la interacción con el usuario.

Las funciones del módulo de aplicación son las siguientes:

- Atender llamadas telefónicas usando los protocolos IAX, propio de asterisk, o SIP , estos se encargan del manejo de sesiones con elementos multimedia, que en este caso seria la voz
- Navegar a través del flujo de la aplicación IVR
- Iniciar el motor de reconocimiento y configurar las gramáticas que va a utilizar
- Enviar la señal de audio al motor de reconocimiento
- Procesar la respuesta de acuerdo al flujo de la aplicación IVR

El sistema de reconocimiento automático de voz, esta formado por PocketSphinx el cual se va a ejecutar como un servicio, que al recibir la petición de audio, por parte de Asterisk y haciendo uso de los modelos acústicos, del modelo de lenguaje y del diccionario, va a retornar la hipótesis que más se ajuste a la señal de audio recibida.

Las funciones del sistema de reconocimiento automático de voz son las siguientes:

- Recibir peticiones de audio
- Procesar la señal de audio, haciendo uso del modelo acústico, modelo de lenguaje y diccionario
- Retornar la mejor de las hipótesis

A continuación se van a describir las tareas principales en el flujo de la aplicación IVR, donde se podrá observar, con la ayuda de algunas trazas, como es la interacción con el usuario y con el servicio de reconocimiento.

En primera instancia se realiza una llamada a la aplicación utilizando el protocolo SIP o IAX con la ayuda de un SoftPhone. Ésta recibe la solicitud y comienza el flujo de la llamada

```
Accepting UNAUTHENTICATED call from ###:
Launching 'Answer'
Executing [s@phonetest:1] Answer("IAX2/###", "") in new
stack
```

con el comando *SpeechCreate* se levanta la información relacionada al puerto y dirección IP del sistema de reconocimiento de voz que se va a utilizar a lo largo de la aplicación. Estos parámetros se encuentran almacenados en un archivo de configuración,

```
Executing [s@phonetest:2] SpeechCreate("IAX2/###",
"Sphinx") in new stack
```

luego se le presenta al usuario un menú con las distintas opciones que va a poder seleccionar mediante comandos de voz. Para presentar las opciones se utilizo un sintetizador de voz (TTS – Text To Speech) conocido como Festival, en el cual se pasa una cadena de texto que luego va a ser reproducida por Asterisk.

```
Executing [s@phonetest:4] Festival("IAX2/###", "Por favor
diga si desea") in new stack
Executing [s@phonetest:5] Festival("IAX2/###", "Consultar
saldo") in new stack
Executing [s@phonetest:6] Festival("IAX2/###",
"Actualizar numero de telefono") in new stack
Executing [s@phonetest:7] Festival("IAX2/###", "Protector
```

```
de tarjeta de debito") in new stack
Executing [s@phonetest:8] Festival("IAX2/###", "Bloquear
tarjeta de debito") in new stack
Executing [s@phonetest:9] Festival("IAX2/###",
"Reclamos") in new stack
Executing [s@phonetest:10] Festival("IAX2/###", "o
Informacion de productos y servicios") in new stack
```

Con el comando *SpeechActivateGrammar* se activa la gramática utilizada por el reconocedor de voz, en este caso se utiliza la gramática txn_100.

```
Executing [s@phonetest:11]
SpeechActivateGrammar("IAX2/###", "txn_100") in new stack
```

Al ejecutar *SpeechStart* se le indica al reconocedor de voz que ya puede empezar a escuchar e ir retornando las hipótesis.

```
Executing [s@phonetest:12] SpeechStart("IAX2/###", "") in
new stack
```

SpeechBackground reproduce un texto y luego espera un tiempo (en este caso 10 segundos máximo) para grabar la entrada de datos. Esta entrada es enviada al sistema de reconocimiento de voz.

```
Executing [s@phonetest:13] SpeechBackground("IAX2/###",
"silence/5.gsm|10") in new stack
```

Durante este tiempo de grabación se van produciendo una seria de hipótesis en el sistema de reconocimiento de voz, dependiendo de lo que este diciendo el hablante, para este caso la entrada fue “información de productos y servicios”. Las siguientes son las trazas de reconocimiento, donde se obtuvo como hipótesis “informacion de productos o servicios”

```
Got hyp: 00105 -003173932 'informacion'
```

```
Got hyp: 00106 -003204454 'informacion de'
....
Got hyp: 00200 -006007058 'informacion de productos'
...
Got hyp: 00397 -011935873 'informacion de productos o
servicios'
Finalizing and getting end hypothesis.
```

Ya terminado el proceso de reconocimiento se desactiva la gramática y se guarda la hipótesis retornada dentro de una variable local de Asterisk

```
Executing                                [s@phonetest:14]
SpeechDeactivateGrammar("IAX2/###",  "txn_100")  in  new
stack
Function result is 'informacion de productos o servicios'
```

con ayuda del TTS se sintetiza un audio con la hipótesis obtenida durante el proceso de reconocimiento de voz.

```
Executing  [s@phonetest:16]  Festival("IAX2/###",  "Usted
dijo informacion de productos o servicios") in new stack
```

Luego el IVR, haciendo uso de la gramática confirmenu, le pregunta al usuario si desea continuar o no con las pruebas de reconocimiento. Las trazas obtenidas no se van a mostrar ya que son repeticiones de los procesos observados anteriormente.

- **Aspectos Técnicos**

La plataforma para la construcción del prototipo cumple con las siguientes especificaciones:

- **Sistema Operativo:** La aplicación fue desarrollada bajo Linux, distribución Ubuntu

10.04 LTS

- **PBX:** Asterisk 1.4.36. Software encargado de procesar las llamadas, realizando las funciones de una central telefónica
- **Motor de reconocimiento:** PocketSphinx-0.6. Paquete de software para procesar y decodificar una señal de audio
- **Motor de TTS:** Festival 1.96. Herramienta de software libre capaz de sintetizar un texto a voz
- **SoftPhone:** Software que permite hacer la emulación de un teléfono desde una computadora, con la posibilidad de hacer llamadas usando los protocolos SIP o IAX. Entre los softphone utilizados están:
  - Sjphone 1.60.299, con soporte para llamadas sobre el protocolo SIP
  - Zoiper 2.11, con soporte para llamadas sobre el protocolo SIP y IAX

CONCLUSIONES

Luego de realizado el desarrollo del reconocedor de voz, junto con todas las actividades que implican su construcción, y en vista a los resultados obtenidos durante la fase de pruebas con los distintos modelos acústicos generados, se puede decir que los objetivos planteados inicialmente para el trabajo especial de grado se cumplieron satisfactoriamente.

Entre estas actividades se realizó satisfactoriamente la recolección del corpus de entrenamiento, logrando obtener más de doce horas de grabación que permitieron la construcción de los modelos acústicos, junto con esta recolección se procedió a realizar la transcripción, para aquellos casos en que fuese necesario, pudiendo a lo largo de este proceso detectar los sonidos que no pertenecen al idioma, en su mayoría ruido durante las grabaciones, que serían definidos como sonidos de relleno.

Se realizó la conversión del corpus de audio a 8000 Hz y se definieron los parámetros para el proceso de extracción de los coeficientes espectrales de Mel, con los que se construyeron los vectores de características utilizando las librerías provistas por Sphinx Fron-End.

Una vez que definidos y construidos los parámetros de entrada para el paquete Sphinx-Train, se logró la construcción satisfactoria de los cuatro modelos acústicos y se procedió a realizar las pruebas de reconocimiento con los criterios definidos, para determinar la efectividad de cada modelo. Para estas pruebas se construyeron los modelos de lenguaje enfocados a su uso en sistemas de atención telefónica, como lo es la prestación de servicios bancarios.

Durante la primera fase de pruebas, con los parámetros por defecto de PocketSphinx, no se obtuvieron buenos resultados, como se pudo constatar en la alta tasa de error, un ejemplo de esto se puede observar en los resultados para el reconocimiento por palabras, obteniendo una tasa de error del 43,10%. Luego de los ajustes individuales durante la segunda fase se obtuvieron mejores resultados, como se puede observar en el mayor número de reconocimiento, por intención y por palabras, con respecto a los resultados de la primera fase y una reducción significativa de la tasa de error.

Por último se logró satisfactoriamente la integración con Asterisk, para que utilizara como motor de reconocimiento PocketSphinx, con el modelo acústico del cual se obtuvieron mejores resultados durante la fase de experimentación y haciendo uso de los modelos de lenguaje construidos previamente. Para esto se construyó una aplicación sencilla de telefonía, donde hay un menú de opciones que el usuario debe seleccionar, luego la aplicación le indica cual al usuario cuál fue la opción seleccionada y le pregunta si quiere repetir o no la prueba de reconocimiento.

Finalmente, se puede decir que este trabajo, luego de una ardua investigación de las distintas técnicas utilizadas para la construcción de reconocedores de voz y para el análisis de señales, requiere de grandes conocimiento en el área de matemáticas, estadísticas y probabilidad, que es la base fundamental para la construcción de modelos de Markov, al igual que conocimiento en el área de análisis de señales de audio y de los canales de comunicación, lo cual no hace una tarea para nada trivial el desarrollo de un reconocedor de voz y en donde luego se necesitó de varias horas de entrenamiento, ajustes de parámetros y pruebas para garantizar buenos resultados y que a lo largo de este trabajo permitieron cumplir todos los objetivos, obteniendo resultados finales bastante satisfactorios.

Como recomendaciones para trabajos futuros se pueden hacer mejoras en los siguientes

puntos:

- Automatizar los procesos de prueba para la búsqueda de los mejores parámetros de un modelo acústico dado
- Automatizar el proceso de construcción del diccionario de pronunciación mediante el uso de reglas sintácticas
- Aumentar el corpus de voces actual para realizar nuevos entrenamientos a los modelos acústicos, de manera tal que disminuya la tasa de error
- Automatizar el proceso de transcripción al formato utilizado por Sphinx-Train
- Realizar pruebas de integración en otros sistemas, como los teléfonos inteligentes

REFERENCIAS

BAGHDASARYA, A. (2010). Automatic Phoneme Recognition with Segmental Hidden Markov Models. [Consultado: 07 de Abril del 2010]. Disponible en http://scholar.lib.vt.edu/theses/available/etd-02082010-174617/unrestricted/Baghdasaryan_AG_T_2010.pdf

BAKIS, R. (1976). Continuous speech recognition via centisecond acoustic states. *en 91st Meeting of the Acoustical Society of America*

CHAN, A., GOUVEA, E., SINGH, R., RAVISHANKAR, M., ROSENFELD, R., SUN, Y., HUGGINS-DAINES, D. y SELTZER, M. (2007). The Hieroglyphs: Building Speech Applications Using CMU Sphinx and Related Resources. [Consultado: 07 de Enero del 2010]. Disponible en http://speech.tifr.res.in/tutorials/sphinxDocChan070111.pdf

CHILDERS, D., SKINNER, S. y KEMERAIT, R. (1977). The Cepstrum: A Guide to Processing. Proc. *IEEE, vol 65 No. 10, pp. 1428-1443.*

CHOMSKY, N. y MILLER, G. (1958). Finite state languages. *Information and Control, vol. 1, pp. 91-112.*

DELLER, J., HANSEN, J. y PROAKIS, J. (1993). Discrete-Time Processing of Speech Signals. *Macmillan Publishing Company*

HAGAN, M., DEMUTH, H. y BEALE, M. (1996). Neural Network Design. *Thompson Publishing*

HAYKIN, S. (1999). Neural Networks A Comprehensive Foundation. *Printece Hall*.

HUANG, X., ALLEVA, F., HON, H., HWANG, M. y ROSENFELD, R. (1992). The SPHINX-II Speech Recognition System: An Overview. *Carnegie Mellon University.* [Consultado: 25 de Marzo del 2010]. Disponible en http://www.aclweb.org/anthology/H/H93/H93-1016.pdf

JAYADEV, V. (2007) Hardware Software Codesign of a large vocabulary continuous speech recognition system. [Consultado: 27 de Abril del 2010]. Disponible en http://repository.lib.ncsu.edu/ir/bitstream/1840.16/156/1/etd.pdf

LINDE, J., BUZO, A. y GRAY, R. (1980). An Algorithm for Vector Quantizer Design. *IEEE Transactions on Communications, Vol. COM-28, No. 1, pp. 84-95*

MACIAS, J. (2001). Arquitecturas y métodos en sistemas de reconocimiento automático de habla de gran vocabulario. *Universidad Politécnica de Madrid.* [Consultado: 27 de Abril del 2010]. Disponible en http://www-gth.die.upm.es/~macias/tesis.pdf

MARTINEZ, F., PORTALE, G., KLEIN, H. y OLMOS, O. Reconocimiento de voz, apuntes de cátedra para Introducción a la Inteligencia Artificial. *Universidad Tecnológica Nacional.* [Consultado: 07 de Abril del 2010]. Disponible en http://www.secyt.frba.utn.edu.ar/gia/IA1_IntroReconocimientoVoz.pdf

PECH, J. (2006). Desarrollo de un sistema de reconocimiento de voz para el control de dispositivos utilizando mixturas gaussianas. *Instituto Politécnico Nacional.* [Consultado: 07 de Abril del 2010]. Disponible en http://itzamna.bnct.ipn.mx:8080/dspace/bitstream/123456789/1427/1/1000_2006_CIC_MAESTRIA_pech_carmona_jaimehumberto.pdf

PÉREZ, E. (2006). Construcción de un reconocedor de voz utilizando Sphinx y el corpus DIMEx100. *Universidad Nacional Autónoma de México.* [Consultado: 10 de Marzo del 2010]. Disponible en http://leibniz.iimas.unam.mx/~luis/DIME/publicaciones/tesis/Tesis-Paty.pdf

PLANNERER, B. (2005), An Introduction to Speech Recognition. *Bernd Plannerer*. [Consultado: 25 de Marzo del 2010]. Disponible en http://www.speech-recognition.de/pdf/introSR.pdf

PUERTAS, J. (2000), Robustez en reconocimiento fonético de voz para aplicaciones telefónicas. *Universidad Politécnica de Madrid.* [Consultado: 28 de Abril del 2010]. Disponible en http://oa.upm.es/657/1/JOSE_IGNACIO_PUERTAS_TERA.pdf

RABINER, L. (1989). A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *Proceedings of The IEEE, Vol. 77, NO. 2, pp. 257-286.*

RAVISHANKAR, M. (1996). Efficient Algorithms for Speech Recognition. *Carnegie Mellon University.* [Consultado: 23 de Marzo del 2010]. Disponible en http://www.cs.cmu.edu/~rkm/th/th.pdf

ROGERS, F. (1989). On the application of vector quantization to speaker-independent isolated word recognition. *Simon Fraser University.* [Consultado: 06 de Junio del 2010]. Disponible en http://ir.lib.sfu.ca/bitstream/1892/8240/1/b17878883.pdf

SHANEH, M y TAHERI, A. (2009). Voice Command Recognition System Based on MFCC and VQ Algorithms. *World Academy of Science, Engineering and Technology 57, pp. 534-538*

SHANNON, C. (1948). A mathematical theory of communication. *Bell System Technical Journal. Vol 27, pp. 379-423, 623-656*

WALEED, H y KASABOV, N. (1999). The Concepts of Hidden Markov Model in Speech Recognition. *University of Otago.* [Consultado: 06 de Abril del 2010]. Disponible en http://www.aut.ac.nz/resources/research/research_institutes/kedri/downloads/pdf/waleed-kas-9909.pdf

Grammar Format Especification. (1998). Especificación del formato de gramáticas de java. [Consultado: 23 de Mayo del 2010]. Disponible en http://java.sun.com/products/java-media/speech/forDevelopers/JSGF/JSGF.html

SAMPA Spanish. (1993). Alfabeto fonético SAMPA. [Consultado: 28 de Abril de 2010]. Disponible en http://liceu.uab.es/~joaquim/publicacions/SAMPA_Spanish_93.pdf