Universidad Nacional de Trujillo
Escuela de Postgrado

Sección de Postgrado en Ciencias Físicas y Matemáticas

Recuperación de Información en Textos Hablados

Tesis presentada
a la
Escuela de Postgrado
de la
Universidad Nacional de Trujillo
para
La Obtención del Grado
de
Magister en Ciencia de la Computación

Autor: Jorge Luis Guevara Díaz

Trujillo-Perú, diciembre de 2010

Recuperación de Información en Textos hablados

Jurado:

- Ms. Carlos Castillo Diestra - Escuela de Informática-UNT.
- Ms. José Cruz Silva - Escuela de Informática-UNT.
- Ms. Edwin Mendoza Torres - Escuela de Matemática-UNT.

Dedicatoria

Para Christopher.

Agradecimientos

Un profundo agradecimiento a todas las personas que hicieron posible el desarrollo de esta investigación. En particular a mi madre Ester por su apoyo y motivación para el estudio constante, por sus palabras motivadoras, su buen ejemplo y su amor incondicional. Agradezco a mi hermana Yaqueline por ayudarme en los trámites necesarios cuando me encontraba lejos, sin los cuales hubiera sido imposible presentar esta tesis a tiempo.
Agradezco de manera especial a mi esposa Leissi por la revisión de la redacción del texto y todo el apoyo que me dio para culminar esta tesis.

Resumen

La recuperación de información de textos hablados es la transcripción de archivos con habla hacia una secuencia de palabras usando un computador. En este contexto, el término recuperación significa transcripción de los archivos de audio, información significa secuencia organizada de caracteres en palabras con algún significado, y textos hablados son grabaciones de audio con habla realizada por una persona en particular.

Este trabajo propone un algoritmo para la recuperación de información en textos hablados dependientes del hablante. Cada texto dependiente del hablante es generado de manera aleatoria por una gramática libre del contexto. Son utilizadas técnicas de diseño y de análisis de algoritmos, conceptos y algoritmos de procesamiento digital de señales y de reconocimiento de patrones. Este trabajo usa el enfoque de reconocimiento automático del habla continua dependiente del hablante, cuya finalidad es la transcripción del habla hacia una secuencia de caracteres usando un computador para un hablante en particular.

Posibles aplicaciones son aquellas que deseen realizar la recuperación de información de muchos archivos de audio de manera automática por un software de computador, donde cada archivo contenga el habla de una persona en particular y sea conocida la gramática que genera los textos a recuperar.

Palabras-clave: Reconocimiento de patrones, reconocimiento automático del habla dependiente del hablante, gramática libre del contexto.

Abstract

Information retrieval of spoken words is the transcription of speech audio files to word sequence using a computer algorithm. In this context, the word retrieval means transcription of audio files, information means organized sequence of characters in words with some meaning, and spoken words are speech audio recordings made by one person in particular.

This thesis propose an algorithm for information retrieval of speaker dependent spoken words. Each speaker dependent spoken words is randomly generated by a free-context grammar. Design and analysis of algorithms techniques, digital signal processing algorithms and pattern recognition algorithms are used. This thesis use the speaker dependent continuos speech recognition approach whose purpose is the transcription of speech audio files to character sequences using a computer algorithm by one person in particular.

Potential applications are those who wish to make automatic information retrieval from many audio files by computer software, where each file contains the speech of one person in particular and the grammar that generated of text to be retrieval is know.

Keywords: Pattern recognition, speaker dependent speech Recognition, free-context grammar.

Indice

Lista de Abreviaturas

DSP	procesamiento digital de señale.
STFT	transformada de corta de Fourier
DFT	transformada discreta de Fourier
FFT	transformada rápida de Fourier
DCT	transformada discreta del coseno
HMM	modelos Ocultos de Markov
GMM	modelos de mezclas de gausianas
kfold	validación cruzada k-fold
WER	tasa de error por palabras
IPA	alfabeto fonético internacional
MFCC	coeficientes ceptrales en frecuencia mel

Indice de figuras

Indice de tablas

Capítulo 1

Introducción

1.1. Consideraciones Preliminares

La recuperación de información en textos hablados *RITH* se define como la recuperación de información que está contenida dentro de archivos de audio, donde *información* es un conjunto organizado de datos con significado para un sujeto o sistema en particular. Entre las principales aplicaciones de RITH tenemos todas aquellas que requieran recuperar información de habla de muchos archivos de audio de manera automática como contestadoras telefónicas, grabaciones de conversaciones formales hechas por empresas de servicios, archivos de audios con información periodística, buscadores automáticos en archivos de audio, entre otros.

En la presente tesis limitamos la definición de recuperación de información de textos hablados hacia la transcripción de archivos con habla hacia una secuencia de palabras usando un computador. En este contexto el término recuperación significa transcripción de los archivos de audio, información significa secuencia organizada de caracteres en palabras con algún significado y textos hablados son grabaciones de audio con habla realizada por una persona en particular.

Trabajos previos muestran el uso de sistemas de reconocimiento automático del habla para recuperar ciertas palabras clave o ciertas frases de archivos de audio, éste tipo de enfoque de RITH se conoce en inglés como *keyword-spotting* (*KS*). Estos sistemas de reconocimiento automático del habla son entrenados solamente con las palabras deseadas. [Ros90], [Wil92]., [Ros91]. Diversos algoritmos de KS para sistemas independientes del hablante basados en modelos ocultos de Markov[Mar67], donde el usuario permite especificar las palabras clave de manera dinámica y entrenar el sistema han sido propuestos [WB92]. El reconocimiento del habla de una conversación informal es todavía un reto, diversos experimentos realizados en el Switchboard corpus [MNJ+94] tienen una tasa de error por palabras cercana del 70 % [JEC+95]. Transcripciones automáticas de noticias de radio y televisión por sistemas de reconocimiento automático del habla que manejen un un vocabulario grande de cientos de palabras son difíciles de crear por los frecuentes e imprevisibles cambios de voz que ocurren en el hablante, estilo de habla, condiciones de fondo entre otros [KJM+97]. Los modelos ocultos de Markov [Mar67] también han sido utilizado con éxito en sistemas KS [Ros90]. Enfoques que exploran el uso de la programación dinámica también han sido investigados [SC78]. Técnicas de recuperación de información y de reconocimiento automático del habla para RITH han sido abordadas con importantes resultados [KKB94]. Una recopilación de diversas técnicas y el estado de arte en sistemas RITH y de sistemas de recuperación de información en sistemas de audio musical también han sido abordadas [Foo99].

Los sistemas de reconocimiento automático del habla con un vocabulario grande se entrenan con relativamente pocos fonemas en vez de miles de palabras. Cada fonema es la combinacion de uno o más sonidos de una palabra. Estos sistemas trabajan con un diccionario de pronunciación basado en concatenación de fonemas, como ejemplo la palabra *uno* es pronunciada usando tres

fonemas: uh-n-oh (por convención la notación es la del sistema de notación fonética internacional *IPA*). Es necesario construir *modelos acústicos* de cada fonema a partir de cientos de fonemas de ejemplo, esto se logra usando técnicas de procesamiento digital señales para procesar la señal de habla, para luego crear modelos que representen estos fonemas, la técnica mas usada en esta etapa son los modelos ocultos de Markov [Mar67]. La distribución del lenguaje juega un rol importante en el diseño de sistemas de reconocimiento automático del habla, este conocimiento a priori ayuda a incrementar la tasa de reconocimientos, técnicamente a este modelo se le conoce como *modelo de lenguaje* y brinda información de la probabilidad de ocurrencia de las combinaciones de secuencias de palabras por ejemplo la probabilidad de ocurrencia de la secuencia de palabras: *"de el"* versus la probabilidad de ocurrencia de la secuencia de palabras *"el del"*.

Sistemas RITH usando reconocimiento automático del habla pueden lograr un poco más del *90 %* de precisión de palabras en un sistema con grabaciones hachas de manera cuidadosa, y un dominio limitado como el Wall Street Journal Corpus [PFF+94]. Sistemas similares logran un poco más del *60 %* de precisión de palabras en tareas del mundo real como conversaciones telefónicas [JFSJY96], [HW97], [JEC+95] [KJN+97]. Si bien los resultados para estas tareas no son muy buenos, el resultado de las transcripciones del reconocimiento automático del habla y el uso de algoritmos de recuperación de información puede mejorar los resultados.

En la presente tesis usamos el reconocimiento automático del habla continua dependiente del hablante, limitando así el problema a recuperar información del texto hablado de una persona en particular en ambientes controlados libres de ruido, en donde se tenga información a priori del lenguaje, el cual es modelado por una gramática libre del contexto.

1.2. Objetivos

El objetivo de este trabajo es proponer un algoritmo para recuperar información de textos hablados dependientes del hablante con información a priori del lenguaje ,haciendo uso de conceptos de reconocimiento automático del habla para palabras continuas. Diversos conceptos de procesamiento digital de señales, reconocimiento de patrones y fonética serán abordados para lograr este objetivo.

1.3. Contribuciones

Las principales contribuciones de la presente tesis son las siguientes:

- Se propuso el algoritmo de recuperación de información en textos hablados.
- Revisión bibliográfica presente en los Capítulos 2, 3 y 4.
- Se realizó una investigación sobre el proceso de extracción de características en reconocimiento automático de palabras aisladas [GD07], esta investigación empezó como parte de tesis de pregrado, fueron publicados artículos al respecto en [DÍA07] y en [DÍA05]
- Fue ministrada la disciplina de *Reconocimiento Automático del Habla* como parte del contenido del curso *Tópicos Especiales en Ciencia de la Computación* en la escuela académico profesional de Informática de la Universidad Nacional de Trujillo en los años 2007, 2008 y 2009.
- Se organizó el evento Speech Recognition: Algoritmos y aplicaciones donde se expuso el tema titulado "Técnicas de clustering para la inicialización de Modelos Ocultos de Markov para un sistema de reconocimiento automático del habla", como también los trabajos de investigación de los alumnos del curso de Tópicos Especiales en Ciencia de la Computación, disponibles en "http://eventos.seccperu.org/sraa2009/".

- Se realizaron diversas presentaciones de trabajos preliminares de la presente tesis en diversas conferencias, entre los cuales destacan el estudio de técnicas de clustering para el problema de inicialización de Modelos Ocultos de Markov [DÍA09a], se investigó e implementó una técnica eficiente para la construcción de Modelos Ocultos de Markov usando logaritmos de probabilidades [DÍA09b], se realizó un estudio comparativo sobre técnicas de extracción de características para un sistema de reconocimiento automático del habla [DÍA08a], entre otros [DÍA08b], [DÍA06].
- Fueron construidos dos softwares, uno llamado "Lorito"para analizar el proceso de selección de características, y otro de experimentación de la presente tesis para recuperación de información en textos hablados.

1.4. Organización de la Tesis

El Capitulo 2 presenta una revisión de conceptos de procesamiento digital de señales. son revisados los conceptos de señal digital, transformada de Fourier, transformada rápida de Fourier, ventaneamiento, filtros FIR, y la transformada discreta del coseno.

El Capitulo 3 presenta toda la teoría para la representación digital de una señal de habla, se describen conceptos como el modelo fuente-filtro de la producción de habla, la aplicación de la transformada corta de Fourier para analizar segmentos de habla, el procesamiento cepstral y la escala de frecuencias Mel, temas importantes que sirven de base para la fase de extracción de características del clasificador.

El Capítulo 4 presenta una revisión de la teoría del Reconocimiento Automático del Habla, analizando la dificultad existente debido a la variabilidad de la señal de habla. Se detalla el proceso de extracción de características usando los coeficientes cepstrales en escala Mel. Se presenta una revisión de la teoría de los Modelos Ocultos de Markov, el modelamiento de la distribución de probabilidades de éstos usando modelos de mezclas de gaussianas. Se presenta una discusión sobre los modelos usados en el reconocimiento automático del habla, así como una breve discusión sobre las limitaciones de estos modelos. Este capitulo también presenta un criterio para la medición de errores en el reconocimiento automático del habla, así como también una discusión sobre el lenguaje hablado. Se presenta además la teoría básica del modelado del lenguaje, así como algoritmos para la creación de fonemas dependientes del contexto

El Capitulo 5 presenta el enfoque de la investigación, la hipótesis, tipo de investigación, universo y muestra, instrumentos, procedimiento, métodos para la recolección de datos y análisis estadístico de los datos.

El Capítulo 6 presenta el algoritmo propuesto para la recuperación de información de textos hablados, se presentan la preparación de los datos, construcción de la gramática, extracción de características, construcción de los Modelos Ocultos de Markov y el proceso de decodificación del clasificador.

El Capítulo 7 presenta los resultados experimentales obtenidos en la presente tesis. En este capítulo se presenta la validación general del clasificador usando validación cruzada k-fold, también se realiza la prueba de significancia de t-student, y el análisis del tiempo de ejecución del algoritmo.

El Capitulo 8 presenta la discusión de resultados. El Capitulo 9 presenta las conclusiones. El Capitulo 10 presenta las recomendaciones y perspectivas futuras para la continuación de otros trabajos relacionados.

Se incluye también un apéndice con los archivos de configuración usados, y el código fuente del software construido.

Parte I

Marco Teórico

Capítulo 2

Procesamiento Digital de Señales

La presente sección revisa algunos conceptos importantes de procesamiento digital de señales, en particular se revisarán aquellos conceptos útiles para el procesamiento de señales de habla.

En la sección 2.1 se detallan algunos conceptos preliminares como definición de señal digital, algunas funciones básicas como función impulso unitario, función tren de impulsos así como también el proceso de muestreo y cuantificación de una señal analógica. En la sección 2.2 se estudia la transformada de Fourier en particular se revisan los conceptos de serie de Fourier, transformada de Fourier de funciones continuas de una variable, transformada de fourier de funciones discretas de una variable y la transformada discreta de Fourier de funciones de una variables . En la sección 2.3 se estudia el algoritmo de la transformada rápida de Fourier, también se realiza un análisis del tiempo de ejecución del algoritmo. En la sección 2.4 se describen el ventaneamiento de una señal, en particular se describen la función ventana rectangular y la función ventana generalizada Hamming. En la sección 2.5 se estudian los filtros de primer orden FIR, en particular el filtro de pre-énfasis y finalmente en la sección 2.6 se describe la transformada discreta del coseno.

2.1. Señal Digital

Una señal digital es una representación de una función analógica en una computadora. Para lograr esto es necesario dos procesos claves: el muestreo y la cuantificación de una señal analógica. A continuación describimos algunas funciones básicas que permitirán definir una señal digital

Función Impulso Unitario

La función *impulso unitario* denotada por $\delta(t)$ es definido por:

$$\delta(t) = \begin{cases} \infty & \text{si } t = 0 \\ 0 & \text{si } t \neq 0 \end{cases} \tag{2.1}$$

donde t es una variable continua. La función impulso unitario satisface la identidad:

$$\int_{-\infty}^{\infty} \delta(t)dt = 1 \tag{2.2}$$

y tiene la *propiedad de desplazamiento* con respecto a la integración:

$$\int_{-\infty}^{\infty} x(t)\delta(t - t_0)dt = x(t_0) \tag{2.3}$$

Función Impulso Discreto Unitario

Si n es una variable discreta, la función *impulso discreto unitario* (también conocida como *Kronecker delta*) está definida por:

$$\delta[n] = \begin{cases} 1 & \text{si } n = 0 \\ 0 & \text{si } n \neq 0 \end{cases} \tag{2.4}$$

la cual satisface la identidad:

$$\sum_{n=-\infty}^{\infty} \delta[n] = 1 \tag{2.5}$$

y con la *propiedad de desplazamiento*:

$$\sum_{n=-\infty}^{\infty} x[n]\delta[n - n_0] = x[n_0] \tag{2.6}$$

Función Tren de Impulsos

La función *tren de impulsos* está definida por:

$$s_{\Delta T}(t) = \sum_{n=-\infty}^{\infty} \delta[t - n\Delta T] \tag{2.7}$$

Las señales analógicas son representadas como funciones contínuas, éstas tienen que ser convertidas a una secuencia de valores discretos antes de que ellas sean procesadas por una computadora. Este proceso se llama *muestreo* o *discretización* de una señal. Formalmente este proceso puede ser denotado por la multiplicación de una función continua por un tren de impulsos, cuyo resultado es la función discreta.

Sea $x(t)$ una función contínua, entonces:

$$\begin{aligned} \overline{x}(t) &= x(t)s_{\Delta T}(t) && (2.8) \\ &= \sum_{n=-\infty}^{\infty} x(t)\delta(t - n\Delta T) && (2.9) \end{aligned}$$

donde $\overline{x}(t)$ denota la *función discreta*. El valor de cada muestra es obtenido por integración

$$\begin{aligned} x[n] &= \int_{-\infty}^{\infty} x(t)\delta(t - n\Delta T)dt \\ &= x(n\Delta T) && (2.10) \end{aligned}$$

Teniendo en cuenta el resultado anterior una *señal digital* $x[n]$ es definida como:

$$x[n] = x_a(n\Delta T) \tag{2.11}$$

donde x_a es la *señal analógica*, ΔT es el *periodo de muestreo* y n es un número natural.
La frecuencia de muestreo F_s está definida por $F_s = \frac{1}{\Delta T}$.

Cada valor de la muestra obtenida por el proceso de muestreo tiene que ser almacenado en la computadora, el proceso de representar estos valores se llama *cuantificación*.

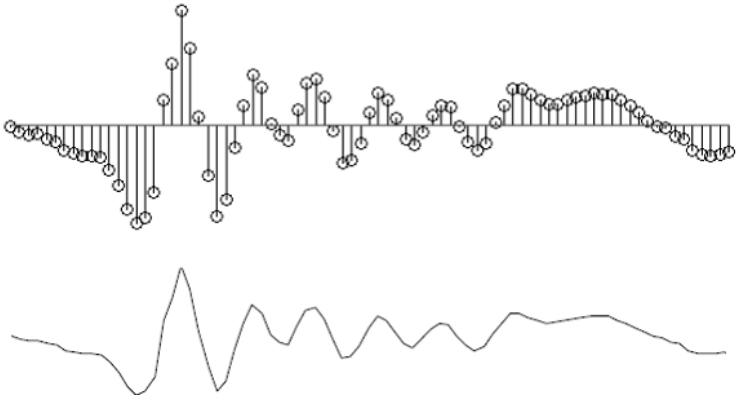


Figura 2.1: *Señal digital y su correpondiente señal analógica.*

2.2. Transformada de Fourier

2.2.1. Series de Fourier

Definición Una función continua $x(t)$, que es periódica con periodo T, puede ser expresada por la suma de senos y cosenos, multiplicados por los coeficientes apropiados. La *serie de Fourier* tiene la forma:

$$x(t) = \sum_{n=-\infty}^{\infty} C_n e^{j2\pi nt/T} \tag{2.12}$$

donde

$$C_n = \frac{1}{T}\int_{-T/2}^{T/2} x(t)e^{-j2\pi nt/T}dt \quad \text{para } n = 0, \pm 1, \pm 2, ... \tag{2.13}$$

2.2.2. Transformada Continua Fourier de Funciones de Una Variable

Definición Definimos la *transformada de Fourier de funciones de una variable continua* como:

$$X(\omega) = \int_{-\infty}^{\infty} x(t)e^{-j\omega t}dt \tag{2.14}$$

y la *transformada inversa de Fourier de funciones de una variable continua* como:

$$x(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} X(\omega)e^{-j\omega t}\delta\omega \tag{2.15}$$

donde $\omega = 2\pi f$. f indica la frecuencia.

2.2.3. Transformada de Fourier de Funciones Discretas

Definición La operación *convolución* está definida como:

$$x(t) * h(t) = \int_{-\infty}^{\infty} x(\tau)h(t-\tau)d\tau \tag{2.16}$$

Un resultado importante de la convolución es el *teorema de la convolución*

$$x(t) * h(t) \Leftrightarrow X(\omega)H(\omega) \tag{2.17}$$

y

$$x(t)h(t) \Leftrightarrow X(\omega) * H(\omega) \tag{2.18}$$

Usando el teorema de la convolución, la transformada de Fourier de una función discreta (Eq. 2.8) está dado por:

$$\overline{X}(\omega) = X(\omega) * S(\omega) \tag{2.19}$$

El tren de impulsos definido en (Eq. 2.7), es una función periódica con periodo igual a ΔT, entonces puede ser expresada mediante series de Fourier:

$$s_{\Delta T}(t) = \sum_{n=-\infty}^{\infty} C_n e^{j2\pi nt/\Delta T} \tag{2.20}$$

donde los coeficientes C_n son calculados como sigue

$$\begin{aligned} C_n &= \frac{1}{\Delta T}\int_{-\Delta T/2}^{\Delta T/2} s_{\Delta T}(t)e^{-j2\pi nt/\Delta T}dt \\ &= \frac{1}{\Delta T} \end{aligned} \tag{2.21}$$

Por tanto:

$$s_{\Delta T}(t) = \frac{1}{\Delta T}\sum_{n=-\infty}^{\infty} e^{j2\pi nt/\Delta T} \tag{2.22}$$

Luego la transformada de Fourier de un tren de impulsos está dado por:

$$S(\omega) = \frac{1}{\Delta T}\sum_{n=-\infty}^{\infty} \delta(k - \frac{n}{\Delta T}) \tag{2.23}$$

donde $\omega = 2\pi k$.

Este resultado muestra que la transformada de Fourier de un tren de impulsos con periodo ΔT también es un tren de impulsos con periodo $\frac{1}{\Delta T}$.

Figura 2.2: *Un tren de impulsos y su transformada de Fourier que también es un tren de impulsos.*

A continuación derivaremos la transformada de Fourier de funciones discretas.

$$\begin{aligned} \overline{X}(\omega) &= X(\omega) * S(\omega) \\ &= \int_{-\infty}^{\infty} X(\tau)S(k-\tau)dt \quad \text{usando (Eq. 2.16)} && (2.24)\\ &= \frac{1}{\Delta T}\int_{-\infty}^{\infty} X(\tau)\sum_{n=-\infty}^{\infty}\delta(k-\tau-\frac{n}{\Delta T})dt \quad \text{Reemplazando (Eq. 2.23)} && (2.25)\\ &= \frac{1}{\Delta T}\sum_{n=-\infty}^{\infty}\int_{-\infty}^{\infty} X(\tau)\delta(k-\tau-\frac{n}{\Delta T})dt && (2.26)\\ &= \frac{1}{\Delta T}\sum_{n=-\infty}^{\infty} X(k-\frac{n}{\Delta T})dt && (2.27)\\ & && (2.28) \end{aligned}$$

Donde el ultimo paso es debido a la propiedad de desplazamiento del tren de impulsos dado en (Eq. 2.3).

Este resultado también muestra que la transformada de Fourier $\overline{X}(\omega)$ de una función discreta $\overline{x}(t)$, es una *secuencia periódicamente infinita* de copias de $X(\omega)$, que es la transformada de Fourier de la función continua original. La separación entre las copias es determinado por el valor de $\frac{1}{\Delta T}$.

Si bien $\overline{x}(t)$ es una función discreta, su transformada de Fourier $\overline{X}(\omega)$ es continua, por que consiste de copias de $X(\omega)$ que es una función continua.

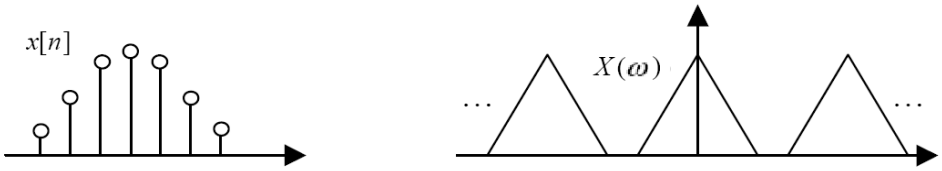


Figura 2.3: *Transformada de Fourier de una función discreta, $x[n]$ es una función discreta y $X(\omega)$ es continua y periódica.*

2.2.4. Transformada Discreta de Fourier de Funciones de Una Variable

Definición Definimos la *transformada discreta de Fourier* (DFT) como:

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{\frac{-j2\pi nk}{N}} = \sum_{n=0}^{N-1} x[n] W_N^{kn}, \quad k = 0, 1, ..., N-1 \tag{2.29}$$

donde:

$$W_N = e^{\frac{-j2\pi}{N}} \tag{2.30}$$

el factor W es llamado *factor mariposa.*

y la *transformada inversa discreta de Fourier* (IDFT) como:

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{\frac{j2\pi nk}{N}} = \sum_{k=0}^{N-1} X[k] W_N^{-kn} \quad n = 0, 1, ..., N-1 \tag{2.31}$$

La DFT es derivada de la siguiente manera:

Sean $\overline{X}(\omega)$ definida en (Eq. 2.24) la transformada de Fourier continua de una función discreta. Sea $\overline{x}(t)$. definida en (Eq. 2.8) una función discreta, entonces:

$$\begin{aligned}
\overline{X}(\omega) &= \int_{-\infty}^{\infty} \overline{x}(t) e^{-j\omega t} \delta t \\
&= \int_{-\infty}^{\infty} \sum_{n=-\infty}^{\infty} x(t) \delta[t - n\Delta T] e^{-j\omega t} dt \quad \text{(usando 2.7)} \\
&= \sum_{n=-\infty}^{\infty} \int_{-\infty}^{\infty} x(t) \delta[t - n\Delta T] e^{-j\omega t} dt \\
&= \sum_{n=-\infty}^{\infty} x[n] e^{-j2\pi ft} \quad \text{(usando 2.10)}
\end{aligned} \tag{2.32}$$

Si bien $x[n]$ es una función discreta, su transformada de Fourier $\overline{X}(\omega)$ es continua e infinitamente periódica, con periodo $\frac{1}{\Delta T}$ (Eq. 2.24). Para construir la DFT se necesita solo un periodo y muestrear este periodo.

Para discretizar f podemos tomar digamos N muestras igualmente espaciadas:

$$f = \frac{k}{N\Delta T} \quad \text{k=0,..N-1} \tag{2.33}$$

Finalmente la DFT es obtenida:

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{\frac{-j2\pi nk}{N}}, \quad k = 0, 1, ..., N-1 \tag{2.34}$$

2.3. Transformada Rápida de Fourier

El tiempo de ejecución de la DFT es $O(n^2)$. La *transformada rápida de Fourier* (FFT)[CT65] cuyo tiempo de ejecución es $O(n \lg n)$ permite un cálculo más eficiente de la DFT de una señal. la FFT se basa en el enfoque de diseño de algoritmos *divide y conquista*. la cual *divide* un problema en otros mas pequeños, soluciona (*conquista*) los subproblemas de manera recursiva y finalmente *combina* las soluciónes para obtener la solución original.

2.3.1. Algoritmo Radix-2 con Diezmado en Frecuencia y Reordenamiento en la Salida de Bits

A continuación se describe un algoritmo para el cálculo de la FFT llamado *Algoritmo Radix-2 con Diezmado en Frecuencia y reordenamiento en la salida de bits*[EA00].

Sea la DFT:

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{\frac{-j2\pi nk}{N}} = \sum_{n=0}^{N-1} x[n] W_N^{kn}, \quad k = 0, 1, ..., N-1 \tag{2.35}$$

donde:

$$W_N = e^{\frac{-j2\pi}{N}} \tag{2.36}$$

De 2.36 podemos establecer:

$$W_N^{\frac{N}{2}} = -1 \tag{2.37}$$

$$W_{\frac{N}{2}} = W_N^2 \tag{2.38}$$

$$W_N^N = 1 \tag{2.39}$$

donde N es una potencia de 2. (para que el enfoque *divide y conquista* pueda ser aplicado).

En la fase de *divide* el algoritmo divide el problema original en dos subproblemas de la forma:

$$\{X(2k) | k = 0, ..., \frac{N}{2} - 1\} \quad \text{(frecuencias diezmadas para índices pares)} \tag{2.40}$$

$$\{X(2k+1) | k = 0, ..., \frac{N}{2} - 1\} \quad \text{(frecuencias diezmadas para índices impares)} \tag{2.41}$$

La ecuación 2.35 puede ser escrita como: [†1]

$$X_k = \sum_{n=0}^{\frac{N}{2}-1} x_n W_N^{kn} + \sum_{n=\frac{N}{2}}^{N-1} x_n W_N^{kn} \tag{2.42}$$

$$X_k = \sum_{n=0}^{\frac{N}{2}-1} x_n W_N^{kn} + \sum_{n=0}^{\frac{N}{2}-1} x_{n+\frac{N}{2}} W_N^{k(n+\frac{N}{2})} \tag{2.43}$$

$$X_k = \sum_{n=0}^{\frac{N}{2}-1} (x_n + x_{n+\frac{N}{2}} W_N^{k\frac{N}{2}}) W_N^{kn}, \quad k = 0, 1, ..., N-1 \tag{2.44}$$

escogiendo los indices pares tenemos:

$$X_{2k} = \sum_{n=0}^{\frac{N}{2}-1} (x_n + x_{n+\frac{N}{2}} W_N^{kN}) W_N^{2kn} \tag{2.45}$$

$$X_{2k} = \sum_{n=0}^{\frac{N}{2}-1} (x_n + x_{n+\frac{N}{2}}) W_{\frac{N}{2}}^{kn}, \quad k = 0, 1, ..., \frac{N}{2} - 1 \tag{2.46}$$

definiendo $Y_k = X_{2k}$ y $y_n = x_n + x_{n+\frac{N}{2}}$, para $k = 0, 1, ..., \frac{N}{2} - 1$, tenemos la primera mitad del problema:

$$Y_k = \sum_{n=0}^{\frac{N}{2}-1} y_n W_{\frac{N}{2}}^{kn}, \quad k = 0, 1, ..., \frac{N}{2} - 1 \tag{2.47}$$

similarmente para los indices impares tenemos:

$$X_{2k+1} = \sum_{n=0}^{\frac{N}{2}-1} (x_n + x_{n+\frac{N}{2}} W_N^{(2k+1)\frac{N}{2}}) W_N^{(2k+1)n} \tag{2.48}$$

$$X_{2k+1} = \sum_{n=0}^{\frac{N}{2}-1} ((x_n - x_{n+\frac{N}{2}}) W_N^n) W_{\frac{N}{2}}^{kn}, \quad k = 0, 1, ..., \frac{N}{2} - 1 \tag{2.49}$$

definiendo $Z_k = X_{2k+1}$ y $z_n = (x_n - x_{n+\frac{N}{2}}) W_N^n$, para $k = 0, 1, ..., \frac{N}{2} - 1$, obtenemos la segunda mitad del problema:

$$Z_k = \sum_{n=0}^{\frac{N}{2}-1} z_n W_{\frac{N}{2}}^{kn}, \quad k = 0, 1, ..., \frac{N}{2} - 1 \tag{2.50}$$

El cálculo de y_n y de z_n es conocido en la literatura como: la mariposa Gentleman-Sande.

Este procedimiento obtiene una salida mezclada de valores de frecuencia, es necesario un procedimiento adicional para ordenar estos valores, el tiempo de ejecución de dicho procedimiento es lineal.

El ordenamiento de estos valores se hace teniendo en cuenta lo siguiente: las posiciones ordenadas de los valores de salida serán obtenidos al hacer un operación de *inversión de bits* a las posiciones del vector de salida.

[1†] usaremos X_k por $X(K)$ y x_n por $x(n)$

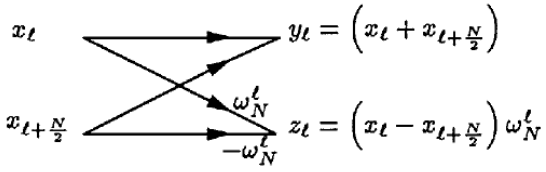


Figura 2.4: *La mariposa Gentleman-Sande.*

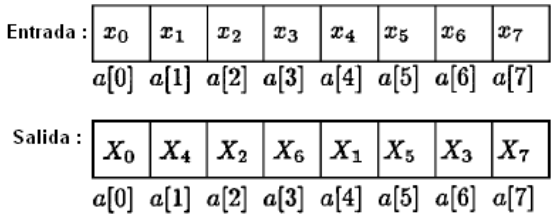


Figura 2.5: *Operación de inversión de bits X.*

Si la notación binaria de un número A es la siguiente:

$$A = abcd...yz \tag{2.51}$$

donde *abcde...yz* son valores que corresponden a los números 0 o 1, su *inversión de bits* sera:

$$A = zyxw....a \tag{2.52}$$

2.3.2. Análisis del Tiempo de Ejecución de la Transformada Rápida de Fourier

La Transformada Rápida de Fourier tiene la siguiente ecuación de recurrencia:

$$T[n] = \begin{cases} 2T(n/2) + \Theta(n) & \text{si } 2^n \geq 2 \\ 0 & n = 1 \end{cases} \tag{2.53}$$

Donde el problema original es dividido en 2 subproblemas, y ambos subproblemas son considerados en el proceso de cálculo. El tiempo de división es lineal $\Theta(n)$, pues solo basta con recorrer toda la entrada y asignar cada elemento a uno de los conjuntos pares o impares. El algoritmo descrito para calcular FFT no usa la fase de *combina*, del enfoque *divide y conquista.*

Usaremos el Teorema Master para solucionar dicha ecuación de recurrencia.

Teorema 2.3.1 (Teorema Master) *Sea $a \geq 1$ y $b > 1$ constantes, Sea $f(n)$ una función, y sea $T(n)$ definida para enteros no negativos por la recurrencia*

$$T(n) = aT(n/b) + f(n) \tag{2.54}$$

donde n/b significa $\lfloor n/b \rfloor$ ó $\lceil n/b \rceil$ entonces $T(n)$ puede ser acotada asintóticamente como sigue:

1. *Si $f(n) = O(n^{log_b a} - \epsilon)$, para alguna constante $\epsilon > 0$, entonces $T(n) = \Theta(n^{log_b a})$*
2. *Si $f(n) = \Theta(n^{log_b a})$, entonces $T(n) = \Theta(n^{log_b a} \lg n)$*
3. *Si $f(n) = \Omega(n^{log_b a} + \epsilon)$, para alguna constante $\epsilon > 0$, y si $f(n/b) \leq cf(n)$ para alguna constante $c < 1$ y todos los valores suficientemente grandes de n, entonces $T(n) = \Theta(f(n))$*

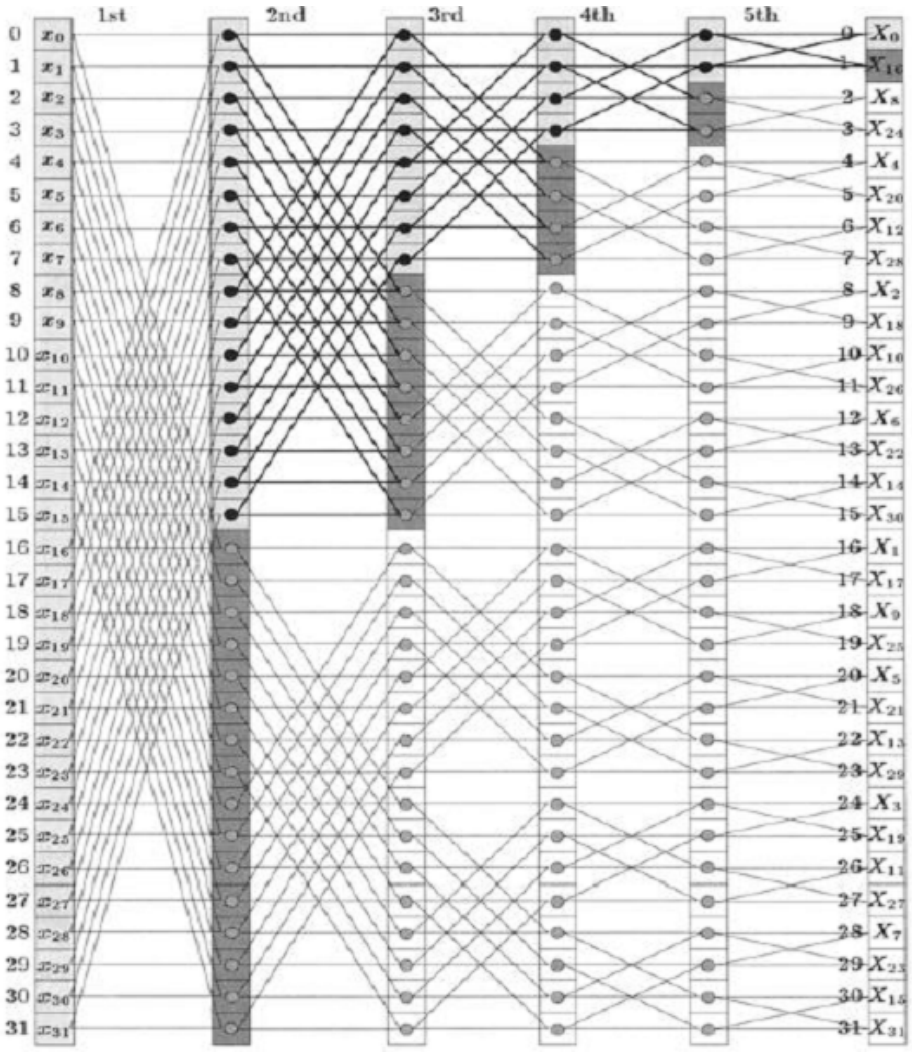


Figura 2.6: *Algoritmo Radix 2 . Fuente: [EA00]*

Usando el Teorema Master tenemos:

$$T(n) = \Theta(n^{\log_b a} \lg \lg n) \tag{2.55}$$

reemplazando los valores para $a = 2$, $b = 2$, $f(n) = \Theta(n)$

La Transformada Rápida de Fourier tiene una complejidad de $\Theta(n \log n)$

2.4. Ventaneamiento

Las funciones ventana son señales concentradas en un lapso de tiempo, y sirven para el análisis de cierta región en particular. A continuación describiremos las siguientes: rectangular, Hanning y Hamming.

2.4.1. Ventana Rectangular

Definición La ventana rectangular es definida como:

$$h_\pi = \delta[n] - \delta[n - N] \tag{2.56}$$

donde la función δ es la función definida en la ecuación 2.5, y N es el tamaño de la ventana.

La transformada de Fourier de la ventana rectangular es:

$$\begin{aligned} H_\pi(\omega) &= \frac{1-e^{-j\omega N}}{1-e^{-j\omega}} \\ &= \frac{(e^{j\omega N/2}-e^{-j\omega N/2})e^{-j\omega N/2}}{(e^{j\omega/2}-e^{-j\omega/2})e^{-j\omega}} \\ &= \frac{\sin \omega N/2}{\sin \omega/2} e^{j\omega(N-1)/2} \end{aligned} \tag{2.57}$$

(2.58)

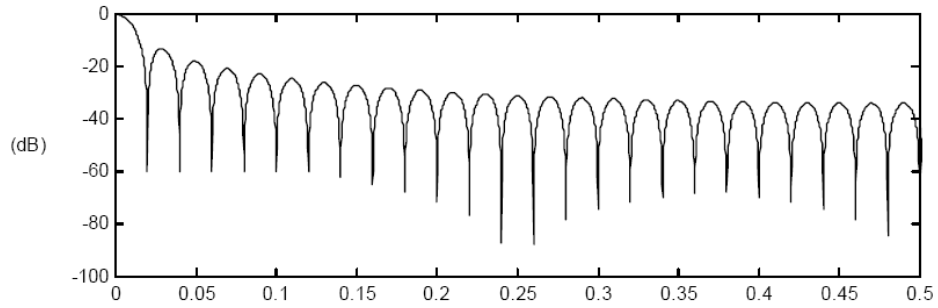


Figura 2.7: *Ventana rectangular con N=50 en el dominio de la frecuencia. Magnitud en decibelios [HAH01]* †

† El valor de energía E es expresado en decibelios como $\overline{E} = 10 \log_{10} E$

2.4.2. Ventana Generalizada Hamming

Definición La forma generalizada de la ventana Hamming es definida como:

$$h_h[n] = \begin{cases} (1-\alpha) - \alpha \cos(\frac{2\pi n}{N}) & \text{si } 0 \leq n < N \\ 0 & \text{otra manera} \end{cases} \tag{2.59}$$

Esta función es expresada es expresada en términos de la ventana rectangular como sigue:

$$h_h[n] = (1-\alpha)h_\pi[n] - \alpha h_\pi[n] \cos(2\pi n/N) \tag{2.60}$$

La transformada de Fourier es la siguiente:

$$H_h(\omega) = (1-\alpha)H_\pi(\omega) - (\alpha/2)H_\pi(\omega - 2\pi/N) - (\alpha/2)H_\pi(\omega + 2\pi/N \tag{2.61}$$

Cuando $\alpha = 0{,}5$ la función ventana se denomina ventana Hanning y cuando $\alpha = 0{,}46$ la función se denomina ventana Hamming.

Analizando el comportamiento en el dominio de Fourier ambas ventanas, podemos notar que el ancho del lóbulo principal de las ventanas Hamming y Hanning es dos veces que de la ventana rectangular. el lóbulo secundario es 31 dB debajo del lóbulo principal para la ventana Hanning y de 44 dB para la ventana Hamming. la atenuación de la ventana Hanning decae rápidamente con la frecuencia, pero no es el caso de la ventana Hamming, cuya atenuación es constante es aproximadamente constante para todas las frecuencias.

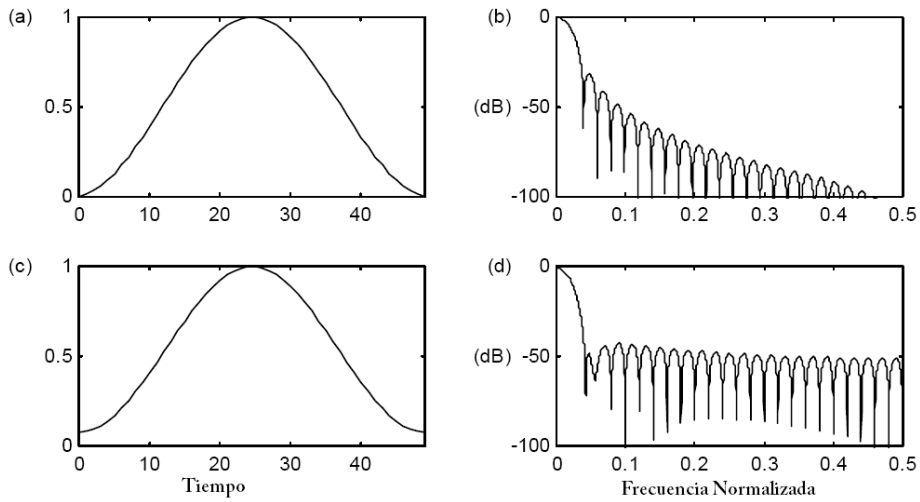


Figura 2.8: *a)Ventana Hanning. b)Magnitud de la respuesta en frecuencia en decibelios; c)Ventana Hamming. d)Magnitud de la respuesta en frecuencia en decibelios para N=50 [HAH01].*

2.5. Filtros FIR

Los *filtros de respuesta finita al impulso* FIR, son un tipo de filtros digitales. Estos filtros tienen la formulación matemática.

$$y[n] = \sum_{r=0}^{M} b_r x[n-r] \tag{2.62}$$

2.5.1. Filtros FIR de Primer Orden

Es un caso especial de los filtros FIR.

Definición Un filtro FIR de primer orden es definido como:

$$y[n] = x[n] + \alpha x[n-1] \tag{2.63}$$

Analizando la magnitud y fase de la respuesta en frecuencia tenemos:

$$|H(\omega)|^2 = |1 + \alpha(\cos\omega - j\sin\omega)|^2 \tag{2.64}$$

$$\theta(\omega) = -\arctan(\frac{\alpha\sin\omega}{1+\alpha\cos\omega}) \tag{2.65}$$

Si $\alpha > 0$ el filtro FIR de primer orden se comporta como *filtro paso bajo*, Si $\alpha < 0$ el filtro FIR de primer orden se comporta como un *filtro paso alto*. Este tipo de filtro es también llamado *filtro pre-énfasis*.

2.6. Transformada Discreta del Coseno

La Transformada Discreta del Coseno (DTC) es ampliamente usada en el procesamiento del habla, ésta tiene varias definiciones; la DTC-II $C[k]$ de una señal real $x[n]$ está definida:

$$c(m) = \sum_{n=0}^{N-1} x[n]\cos(\pi k(\frac{n+\frac{1}{2}}{N})), \quad 0 \le k < N \tag{2.66}$$

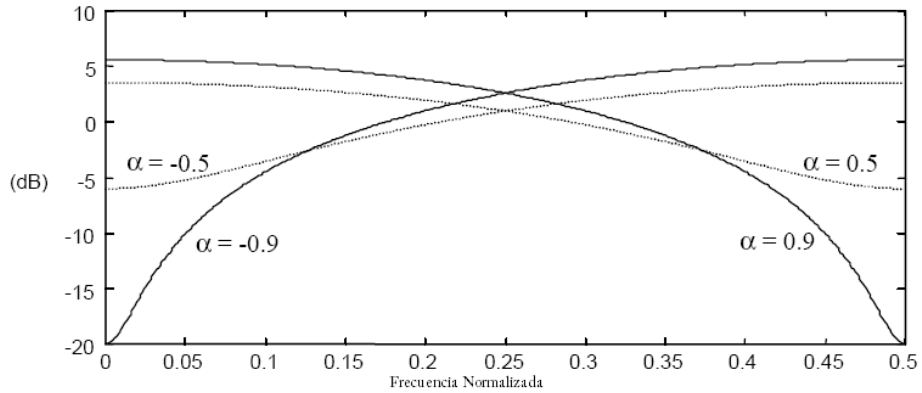


Figura 2.9: *Respuesta en frecuencia de los filtros de primer orden para varios valores de* α *[HAH01].*

y su inversa definida por:

$$x[n] = \frac{1}{N}\{C[0] + 2\sum_{k=1}^{N-1} C[k]\cos(\pi k(\frac{n+\frac{1}{2}}{N}))\} \tag{2.67}$$

La DTC-II puede ser derivada de la Transformada Discreta de Fourier, la DTC-II es más usada, pues tiene su energía compacta esto quiere decir que concentra su energía en los componentes bajos de frecuencia. Esta propiedad permite aproximar una señal con pocos coeficientes [OS09].

Capítulo 3

Representación Digital de la Señal de Habla

A continuación se describen representationes digitales de la señal de habla, útiles para el reconocimiento continuo del habla.

La sección 3.1 describe el modelo fuente-filtro de la producción del habla. La sección 3.2 describe la transformada corta de Fourier como técnica de análisis en señales digitales de habla. La sección 3.3 describe el concepto de cepstrum de una señal digital y finalmente la seccion 3.4 describe una escala perceptual de frecuencias llamada escala en frecuencia Mel.

3.1. Modelo Fuente-Filtro de la Producción de Habla

El modelo Fuente-Filtro es un modelo matemático de la producción del habla, la fuente representa el aire que fluye a través de las cuerdas vocales y el filtro representa las resonancias temporales del tracto vocal.

Matemáticamente está definido como:

$$x[n] = e[n] * h[n] \tag{3.1}$$

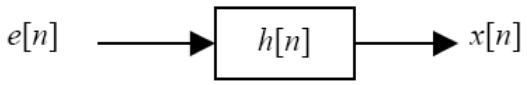


Figura 3.1: *Modelo Fuente-Filtro de las señales de habla [HAH01].*

donde $x[n]$ es la señal de habla, $e[n]$ es la fuente o *excitación* y $h[n]$ es el filtro.

3.2. Transformada Corta de Fourier

Descompone la señal en una serie de frames [†1] y analiza los frames independientemente.

Dada una señal $x[n]$ se define una señal corta en el tiempo $x_s[n]$ de un frame s como sigue:

$$x_s[n] = x[n]w_s[n] \tag{3.2}$$

[1†] Se conoce como frame a una pequeña región de análisis en una señal.

que es el producto de $x[n]$ por una función ventana $w_s[n]$, luego podremos hacer que la función tenga valores constantes para todos los frames:

$$w_s[n] = w[s - n] \tag{3.3}$$

donde $w[n] = 0$ para $|n| > N/2$, donde N es el tamaño del frame (también constante). Usualmente se usa un tamaño de ventana de $20ms$ a $30ms$.

Finalmente se tiene que la Transformada Corta de Fourier para un frame s está definida como:

$$X_s(\omega) = \sum_{n=-\infty}^{\infty} x_s[n]e^{-jw} = \sum_{n=-\infty}^{\infty} w[s-n]x[n]e^{-jw} \tag{3.4}$$

3.3. Procesamiento Cepstral

Es una transformación homomórfica [†1] [Opp65] definida por:

$$\hat{x} = D(x[n]) \tag{3.5}$$

que transforma una convolución

$$x[n] = e[n] * h[n] \tag{3.6}$$

en una suma:

$$\hat{x}[n] = \hat{e}[n] + \hat{h}[n] \tag{3.7}$$

Se define *cepstrum* [OS09] a una transformación homomórfica que permite separar la fuente $e[n]$ del filtro $h[n]$. Es posible encontrar un valor N, partiendo de la suposición de el cesptrum del filtro $\hat{h}[n] \approx 0$ para $n \geq N$ y el el cepstrum de la excitación $\hat{e}[n] \approx 0$ para $n < N$. Con dicha suposición es posible recuperar de manera aproximada $\hat{h}[n]$ y $\hat{e}[n]$ de $\hat{x}[n]$. por filtrado homomórfico.

El cepstrum $D[]$ de una señal digital $x[n]$ esta definido:

$$\hat{x}[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} \ln|X(e^{jw})|e^{jwn} \tag{3.8}$$

El análisis de $\hat{x}$ nos permitirá conocer información del tracto vocal que se encuentra en la parte baja del cepstrum representada por la fuente $e[n]$ y la información del tracto volcal representada por el filtro $h[n]$ contenida en la parte alta del cepstrum, luego se puede separar fácilmente $e[n]$ de $h[n]$ asumiendo el valor N antes mencionado y haciendo la operación inversa $D[]^{-1}$ del cepstrum.

3.4. Escala en Frecuencia Mel

La escala en frecuencia Mel [†2] es una escala perceptual [SV40]. Esta escala es linear debajo de $1KHz$ y logarítmica por encima de este valor. El número de muestras debajo de $1Khz$ es igual al número de muestras por encima de dicho valor. La escala en frecuencia mel se construyó de manera experimental, los sujetos que participaron en el experimento fueron solicitados de dividir los rangos de frecuencias dada en cuatro intervalos perceptuales iguales.

La escala en frecuencia Mel está definida:

[1†] Alan V. Oppenheim definió en su tésis de Phd, una clase de sistemas no lineales llamados sistémas homomórficos. Muchas clases de sistemas no lineales pueden ser interpretados como transformaciones lineales entre espacios vectoriales bajo apropiadas definiciones para operaciones entre vectores de entrada y salida. Esta clase de sistema es llamado sistema homomórfico.

[2†] El nombre de Mel proviene de la palabra inglesa "Melody".

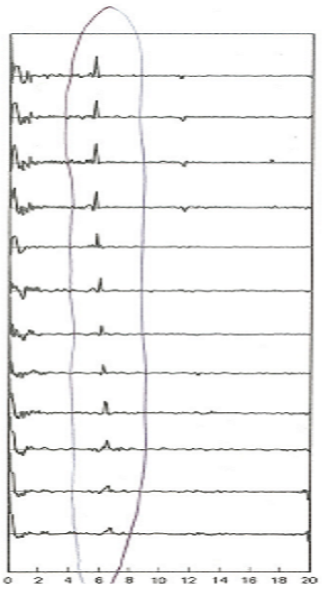


Figura 3.2: *Coeficientes Cepstrales de la señal de habla, la parte baja corresponde al tracto vocal, la parte alta corresponde a la información provenientes de las cuerdas vocales. Fuente: Oppenheim*

$$\beta(f) = 1125 \ln(1 + \frac{f}{700}) \tag{3.9}$$

y su inversa β^{-1} esta dada por:

$$\beta^{-1}[b] = 700(\exp(\frac{b}{1125}) - 1) \tag{3.10}$$

El análisis en frecuencias en escala Mel es ampliamente usado en RAH [HAH01].

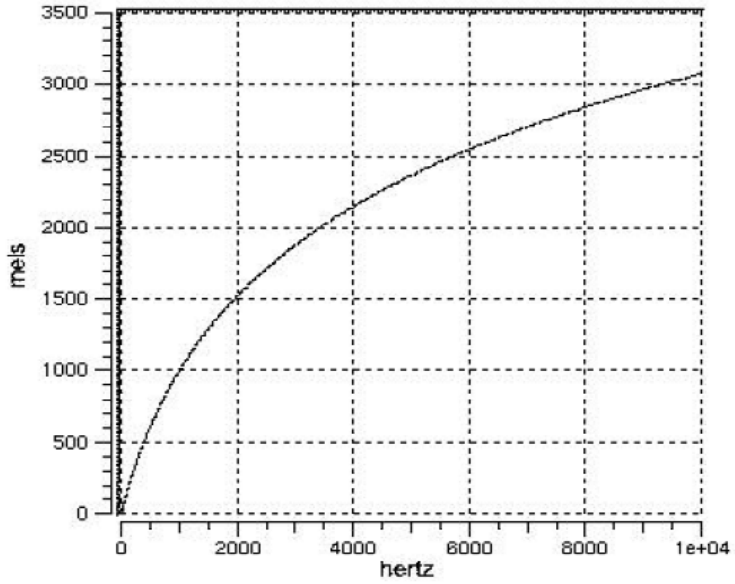


Figura 3.3: *Escala en Frecuencia Mel.*

Capítulo 4

Reconocimiento Automático del Habla

A continuación se describe la tecnología del RAH y los algoritmos involucrados.

La sección 4.1 introduce la definición del reconocimiento automático del habla, también muestra por que es un problema complejo de abordar debido a la variabilidad de la señal de habla. La sección 4.2 describe el algoritmo de extracción de características llamada coeficientes cepstrales en frecuencia Mel ampliamente usada en los sistemas RAH, en esta sección se aplican los conceptos vistos de procesamiento digital de señales y de representación digital de una señal de habla vistos anteriormente. La sección 4.3 describe de manera amplia y detallada los modelos ocultos de Markov, discute los criterios de implementación de estos y los tipos de modelos ocultos de Markov usados en RAH. Finalmente la sección 4.4 describe como evaluar el desempeño de un sistema RAH.

La sección 4.5 describe el proceso del habla como sonido, el proceso de producción y percepción del habla y los conceptos de fonema y fono.

4.1. Introducción

El Reconocimiento Automático del Habla RAH [†1] también conocido como reconocimiento automático de voz es una área de la inteligencia artificial que tiene como fin el convertir una señal acústica de habla a texto usando una computadora.

El habla es representada mediante una secuencia de vectores llamados *observación acústica* que son el resultado del proceso de extracción de características de una señal acústica digitalizada.

Definición *(Observación Acústica)* Llamamos Observación acústica O a la siguiente secuencia de vectores:

$$O = \vec{o_1}, \vec{o_2}, ..., \vec{o_T} \tag{4.1}$$

resultado del proceso de extracción de características. donde $\vec{o_t}$, es el vector de habla observado en el tiempo t.

Dada observación acústica el objetivo del RAH es encontrar la correspondiente **secuencia de palabras** $W = w_1, w_2, ..., w_m$

Definición *(Reconocimiento Automático del Habla)* El reconocimiento Automático del habla (RAH) es formulado usando el teorema de Bayes como sigue:

$$\mathbf{W} = argmax_w P(\mathbf{W}|\mathbf{O}) = argmax_w \frac{P(\mathbf{W})P(\mathbf{O}|\mathbf{W})}{P(\mathbf{O})} \tag{4.2}$$

[1†] del inglés *speech recognition*

Como la maximización de (Eq. 4.2) es realizada manteniendo la observación acústica O fija, la (Eq. 4.2) puede reformularse como sigue:

Definición

$$\mathbf{W} = argmax_w P(\mathbf{W}|\mathbf{O}) = argmax_w P(\mathbf{W})P(\mathbf{O}|\mathbf{W}) \tag{4.3}$$

El término P($\mathbf{O}|\mathbf{W}$) es conocido como *modelo acústico*, El término $P(\mathbf{W})$ es conocido como *modelado del lenguaje*.

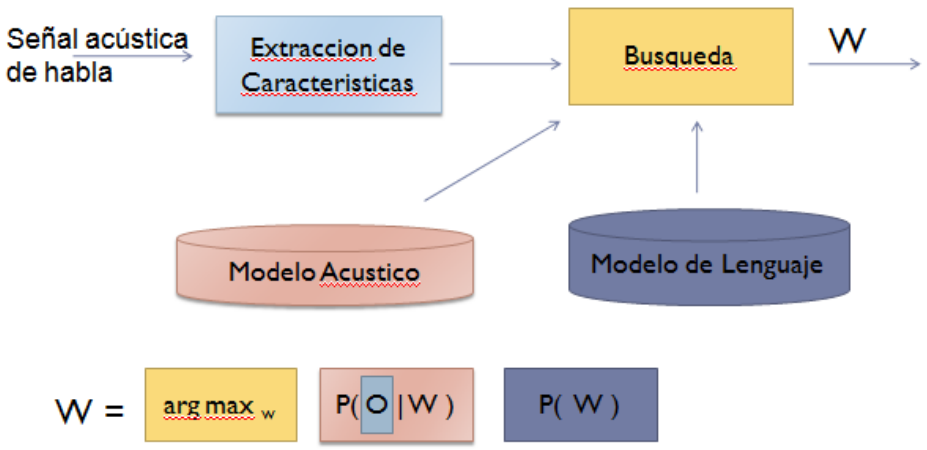


Figura 4.1: *Arquitectura del Reconocimiento Automático del Habla*

4.1.1. Variabilidad de la Señal de Habla

Los mejores sistemas de RAH aún no tienen el mismo desempeño que los humanos, actualmente es posible construir sistemas RAH robustos para usuarios en particular, pero debido a la variabilidad de la señal de habla aún no es posible tener sistemas RAH que funcionen para cualquier hablante, en cualquier idioma, en cualquier tópico. A continuación describiremos los principales criterios de variabilidad de la señal de habla que hacen que RAH sea complejo.

Variabilidad de Contexto

El lenguaje hablado requiere que la gente conozca el significado de las palabras y el contexto en las que ellas aparecen. Palabras con diferentes significados tienen la misma realización fonética. Esto es más frecuente en el idioma ingles. Por ejemplo la siguiente sentencia en el idioma inglés:

- Mr. *Wright* should write to Ms. Wright right away about his Ford or four door Honda.

Las palabras *Wright write* y *right* tienen la misma pronunciación así como también *Ford or four door* no son solamente fonéticamente idénticas si no semánticamente diferentes.

La variabilidad de contexto también se da a novel fonético. por ejemplo el fonema /ee/ de las palabras en inglés peat y wheel dependen de su contexto izquiero y derecho. Aún se tiene más dificultad cuando se tiene una conversación fluida en la cual muchos fonemas no son completamente pronunciados.

Variabilidad de Estilo

El estado de ánimo, la velocidad de la forma de hablar, etc afecta el grado de desempeño de un sistema RAH

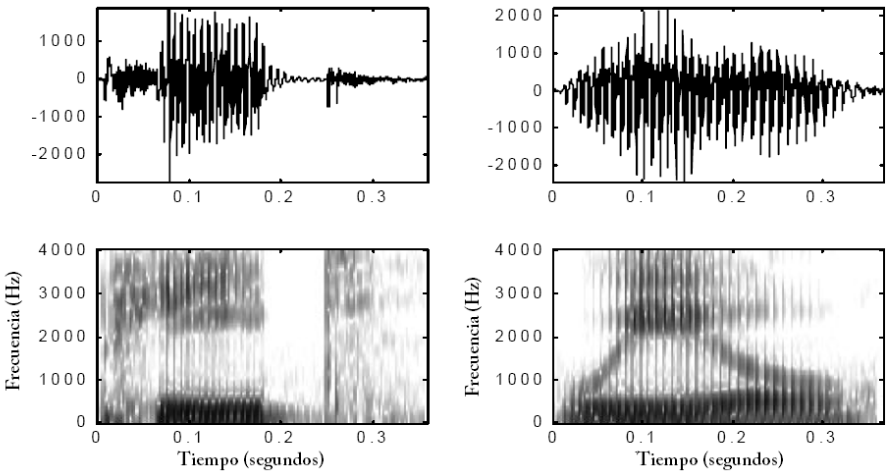


Figura 4.2: *Formas de onda y espectogramas de las palabras* peat *y* wheel *donde el fonema /ee/ es ilustrado en dos diferentes contextos*

Variabilidad de Hablante

Cada hablante es diferente de otro hablante, el habla que genera un hablante refleja la forma física de su tracto vocal, como también características como edad, sexo, dialecto, salud, educación y estilo personal.

Variabilidad de Entorno

El entorno y el sonido producido por el influye en el desempeño de un sistema RAH, como el sonido exterior, características del micrófono, etc.

4.2. Extracción de Características con Coeficientes Cepstrales en Frecuencia Mel.

Características son medidas o propiedades usadas para clasificar los objetos. Los tipos o categorías en los cuales los objetos son clasificados son llamados classes [GJJ96]. El término objeto puede referirse a una imagen digital, archivos de audio, modelos geométricos de estrucuturas 3D, modelos de datos, etc.

Extracción de características es el proceso de determinar las características de un objeto que luego podrán ser usadas en un sistema de reconocimiento de patrones.

Objetos parecidos tienen características parecidas, por otra parte objetos muy diferentes tienen características diferentes. El área que estudia el proceso de extracción de características y clasificar los objetos de acuerdo a sus características se denomina Reconocimiento de Patrones.

En RAH se necesitan características que sean invariantes a la traslación en el tiempo, a los cambios en la amplitud, y que no sean sensitivos a la duración de la palabra.

Existen varias técnicas para extraer características, los Coeficientes Cepstrales en Frecuencia Mel (MFCC) han probado tener un mejor desempeño y son usados en la presente investigación. A continuación se describe el proceso de construcción de los MFCC.

Los MFCC [DM90] es una técnica que se basa en características perceptuales y en la producción de habla humana.

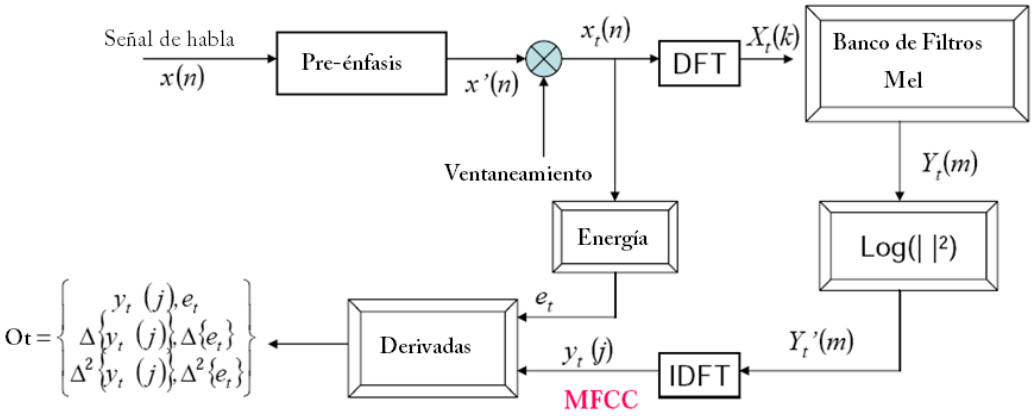


Figura 4.3: *Proceso de construcción de los MFCC.*

El proceso de construcción de los MFCC involucra los siguientes pasos:

- Digitalización y cuantificación de la señal.
- Pre-énfasis en altas frecuencias.
- Ventaneamiento.
- Análisis con Transformada de Fourier.
- Banco del Filtros Mel.
- Análisis Cepstral.
- Coeficientes Cepstrales Dinámicos.

A continuación describiremos este proceso.

4.2.1. Digitalización y Cuantificación de la Señal de Habla

El primer paso de convertir la señal analógica de habla a una señal digital. este proceso tiene dos partes: *muestreo* y *cuantificación*. La señal es muestreada midiendo la amplitud en un tiempo en particular. la frecuencia de muestreo F_s es el número de muestras tomadas por segundo.

La máxima frecuencia presente en la señal para una frecuencia de muestreo se llama *Frecuencia Nyquist* que es igual a

$$F_s = \text{máxima frecuencia presente en la señal} \tag{4.4}$$

Mucha información del habla se encuentra bajo los 10000Hz. entonces para capturar de manera segura todas las frecuencias se necesitará una frecuencia de muestreo de 20000Hz. De manera general el RAH usa como frecuencia de muestreo 8000Hz ó 16000Hz.

El proceso de cuantificación se refiere al proceso de representar cada muestra como un número entero. Si la representación es hecha con 8 bits, cada muestra tendrá valores de -128 a 127, si la representación es hecha con 16 bits, cada muestra tendrá valores entre -32768 a 32767.

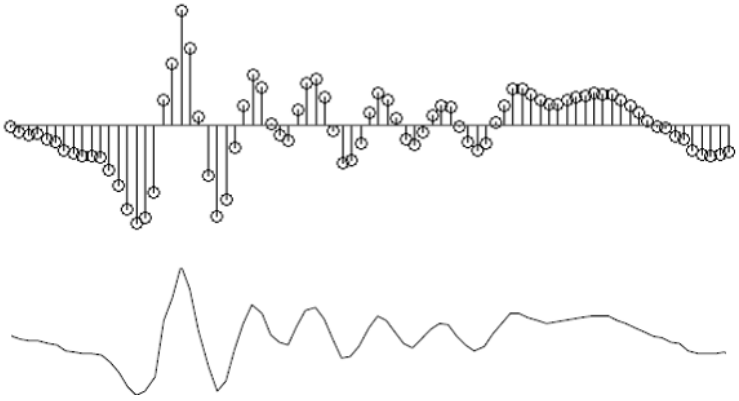


Figura 4.4: *Muestreo de una señal de habla.*

El proceso final es representado como sigue:

$$x[n] = x_a(n\Delta T) \tag{4.5}$$

donde $\Delta T = \frac{1}{F_s}$.

ALGORITMO DIGITALIZACIÓN-CUANTIFICACIÓN(T, x_a)

```
1  ▷ Digitalización y cuantificación de la señal.
2  for n ← 0 to N − 1
3        do
4           x[n] ← x_a(nT)
```

Análisis del Tiempo de Ejecución

El tiempo de ejecución es lineal y por tanto es $\Theta(n)$

4.2.2. Pre-énfasis de la Señal Digital de Habla

Luego que la señal analógica de voz es discretizada. Un filtro FIR de primer orden o filtro de pre-énfasis en altas frecuencias es aplicado (Eq. 4.6), la justificación de esto es que el espectro del habla es caracterizado por una tendencia de descenso en las altas frecuencias. mediante la cual las frecuencias altas del espectro son atenuadas en aproximadamente $6Db$. Esto se debe al hecho de que ocurre un solapamiento de $-12Db$ del espectro del tracto vocal con $6Db$ dado por la radiación de los labios.

Cuando se aplica el filtro pre-enfasis en altas frecuencias se compensa esta diferencia.

$$y[n] = x[n] + \alpha x[n-1] \tag{4.6}$$

donde $x[n]$ corresponde a la señal discreta de habla. el valor de α [†1] es un valor configurable.

Una consecuencia directa de usar este tipo de filtro es incrementar la tasa de reconocimientos [JM09].

ALGORITMO PRE-ÉNFASIS(α)

```
1  for n ← 1 to N − 1
2        do
3           y[n] ← x[n] + αx[n − 1] para n = 1, ..., N − 1
```

[1†] El valor de α generalmente se establece a 0.95 para que el filtro FIR de primer orden se comporte como un filtro de énfasis en altas frecuencias

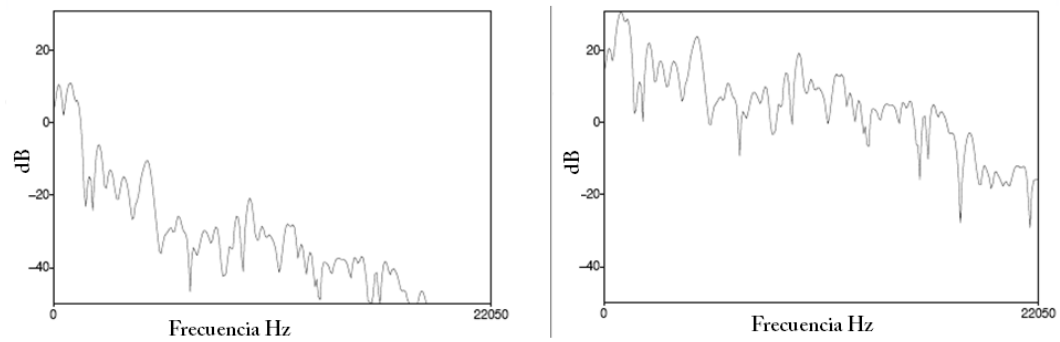


Figura 4.5: *Resultado de aplicar un filtro pre-énfasis en altas frecuencias para la vocal [aa].*

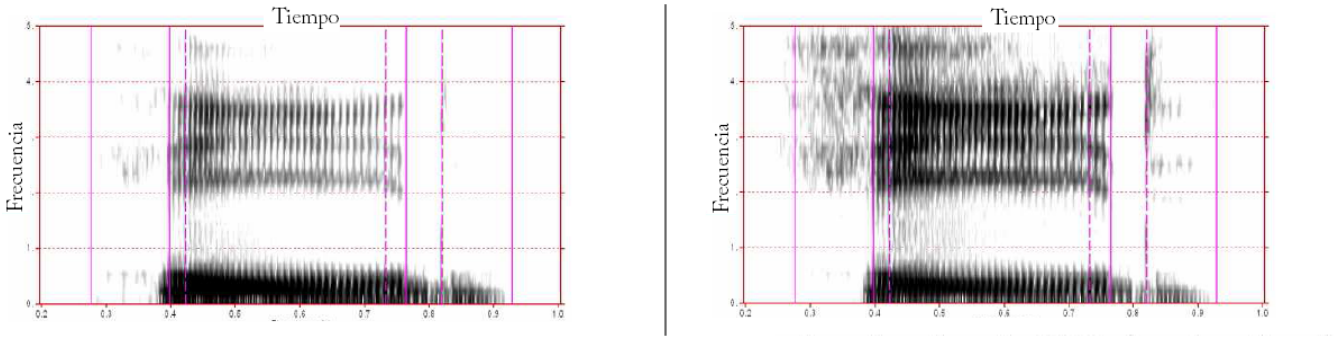


Figura 4.6: *Espectograma antes y despues de aplicar un filtro pre-énfasis en altas frecuencias para la vocal [aa].*

Análisis del Tiempo de Ejecución

El tiempo de ejecución es lineal y por tanto es $\Theta(n)$

4.2.3. Ventaneamiento de la Señal Digital de Habla

La forma tracto vocal generalmente cambia cuando una persona habla, pero tiende a ser constante en intervalos cortos entre 20ms y 25ms. El ventaneamiento es un análisis que se hace a la señal de habla usando ventanas para extraer las características espectrales, dado que el habla es una señal no estacionaria, es decir que sus propiedades estadísticas no son constantes en el tiempo. Dentro de cada ventana se hace la suposición de que la señal es estacionaria esto es de que las propiedades estadísticas son constantes dentro de esta región.

El proceso de ventaneamiento está caracterizado por tres parámetros:

- *Ancho*, es el ancho de la ventana de análisis en milisegundos. Denotaremos al ancho como $\| \; frame \; \|$ que indica el tamaño del frame a ser analizado por la ventana
- *Desplazamiento*, es la separación entre dos ventanas sucesivas. Denotaremos al desplazamiento como *desp*
- *Forma*, es la forma de cada ventana. Por ejemplo rectangular o hamming

Se usa generalmente una ventana Hamming, ya que la ventana rectangular causa problemas cuando se realiza un análisis en el dominio de las frecuencias con Fourier debido a que corta abruptamente la señal.

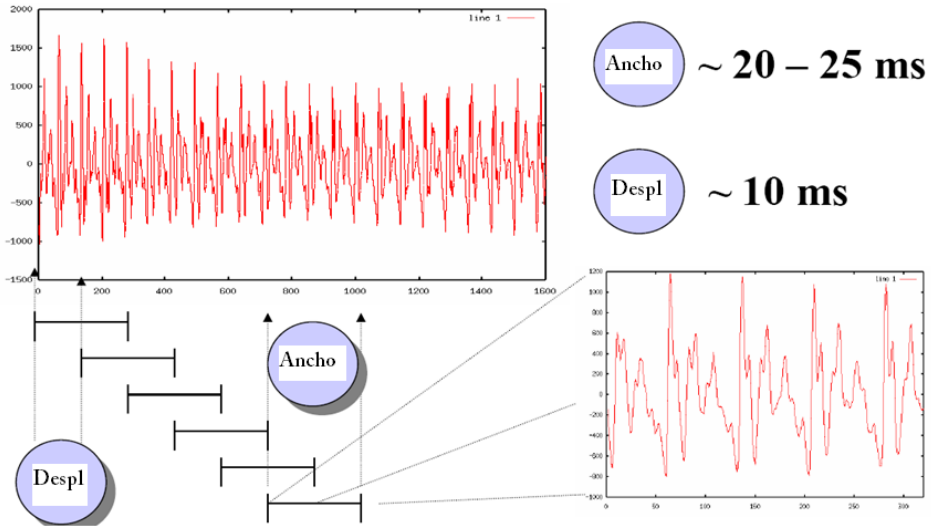


Figura 4.7: *Proceso de ventaneamiento.*

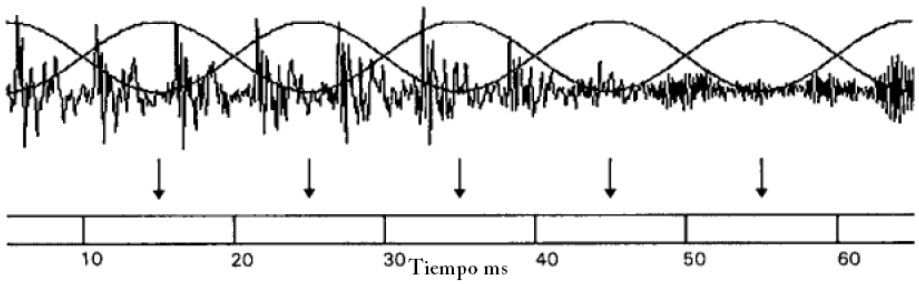


Figura 4.8: *Análisis de una señal de habla con ventanas. donde el ancho = 20 ms, desplazamiento = 10ms, y forma = ventana Hamming.*

El proceso es descrito como sigue

$$x_s[n] = y[n]h_{h_s}[n] \tag{4.7}$$

donde $h_{h_s}[n]$ es una ventana Hamming , que tiene valores constantes para todos los frames s. A seguir el algoritmo para calcular la ventana Hamming.

ALGORITMO VENTANA-HAMMING ($\| \ frame \ \|$)

1 ▷ Ventana Hamming para un frame de tamaño $\| \ frame \ \|$.
2 **for** $i \leftarrow 1$ **to** $\| \ frame \ \| -1$
3 **do**
4 $h_h[i] = (1 - \alpha) - 0{,}46 \cos(\frac{2\pi i}{\|frame\|})$

Análisis del Tiempo de Ejecución

El tiempo de ejecución es lineal y por tanto es $\Theta(n)$

4.2.4. Análisis con Transformada Corta de Fourier

El siguiente paso es calcular la DFT de cada ventana $x_m[n]$, esto es realizado usando el algoritmo FFT (Sección 2.3). Este tipo de análisis se llama Transformada corta de Fourier. (Sección 3.2).

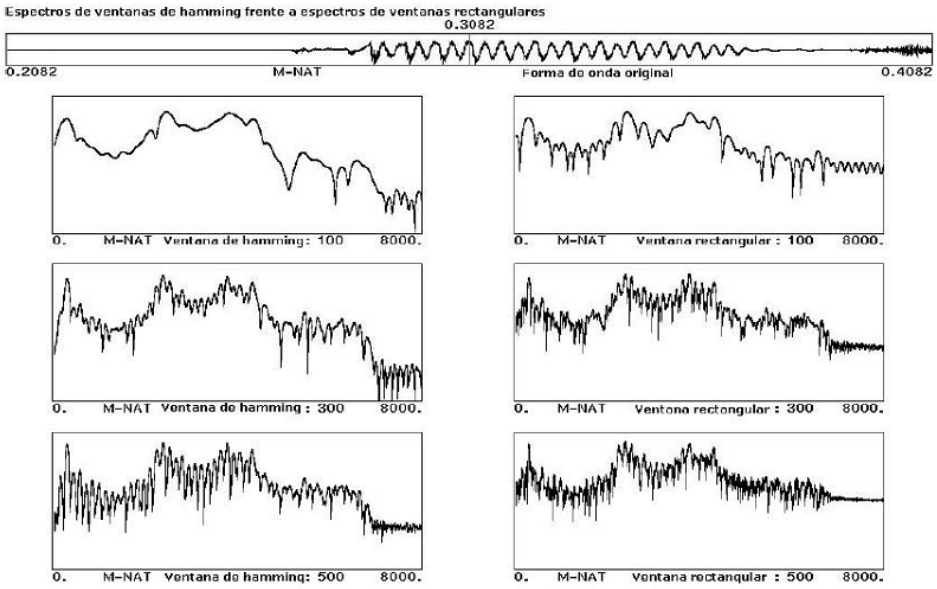


Figura 4.9: *Comparación en el dominio de la frecuencia entre la Ventana Rectangular y la Ventana Hamming [HAH01].*

El proceso es descrito como sigue:

$$X_s(\omega) = \sum_{n=1}^{\infty} x_s[n]e^{-jw} = \sum_{n=-\infty}^{\infty} h_h[s-n]x[n]e^{-jw} \tag{4.8}$$

donde $X_s(e^{jw})$ es la DFT de cada frame producido por la operación de ventaneamiento.

4.2.5. Banco de Filtros en Escala Mel

Dada una Transformada Discreta de Fourier de una señal de entrada:

$$X[k] = \sum_{n=0}^{N-1} x[n]e^{\frac{-j2\pi nk}{N}}, \quad k = 0, 1, ..., N-1 \tag{4.9}$$

Se define un banco de filtros M, con $(m = 1, 2, ..., M)$ donde el filtro m es un filtro triangular dado por:

$$H_m[k] = \begin{cases} 0 & \text{si } k < f(m-1) \\ \frac{k-f(m-1)}{f(m)-f(m-1)} & f(m-1) \leq k \leq f(m) \\ \frac{k-f(m-1)}{f(m+1)-f(m)} & f(m) \leq k \leq f(m+1) \\ 0 & k > f(m+1) \end{cases} \tag{4.10}$$

Estos filtros calculan el promedio del espectro alrededor de cada frecuencia central.

Definimos f_l como la frecuencia más alta y f_h como la frecuencia más baja del banco de filtros en Hz, Fs es la frecuencia de Muestreo en Hz, M el número de filtros y N el tamaño de la Transformada Rápida de Fourier. Los puntos límite $f(m)$ son uniformemente espaciados en la escala Mel:

$$f(m) = \frac{N}{F_s}\beta^{-1}(\beta(f_1) + m\frac{\beta(f_h) - \beta(f_1)}{M+1}) \tag{4.11}$$

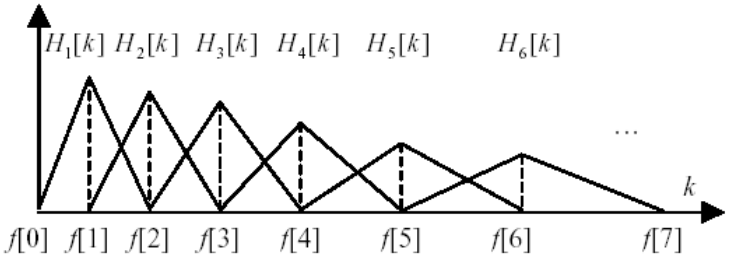


Figura 4.10: *Banco de filtros en escala mel .*

donde la escala Mel β esta dada por:

$$\beta(f) = 1125 \ln(1 + \frac{f}{700}) \tag{4.12}$$

y su inversa β^{-1} esta dada por:

$$\beta^{-1}[b] = 700(\exp(\frac{b}{1125}) - 1) \tag{4.13}$$

Entonces finalmente se computa el logaritmo de la energía de cada filtro:

$$S(m) = \ln(\sum_{k=0}^{N-1} |X_a(k)|^2 H_m(k)), \quad 0 < m < M \tag{4.14}$$

4.2.6. Análisis Cepstral

El Cepstrum en Frecuencia Mel es la Transformada Discreta del Coseno de las salidas de los M filtros:

$$c(m) = \sum_{m=0}^{M-1} S(m) \cos(\pi n(\frac{m - \frac{1}{2}}{M})) \tag{4.15}$$

donde M varía para diferentes implementaciones de 24 a 40, para el Reconocimiento Automático del Habla generalmente son usados los primeros 13 coeficientes.

4.2.7. Coeficientes Cepstrales Dinámicos

A los coeficientes obtenidos del proceso anteior, se agrega la información de la *energía*. Esto se realiza por cada frame $x_m[n]$ resultado del proceso de ventaneamiento

$$E = |x_s[n]|^2 \tag{4.16}$$

donde E representa la energía de un frame.

La señal de habla no es constante, los cambios temporales juegan un rol importante en la percepción humana. Una manera de capturar dicha información es calculando los *coeficientes delta*.

Finalmente los MFCC son construidos de la siguiente manera:

- 12 coeficientes MFCC c_k resultado del análisis cepstral.
- 12 coeficientes de primer orden delta calculados como sigue: $\Delta c_k = c_{k+2} - c_{k-2}$.
- 12 coeficientes de segundo orden delta calculados como sigue: $\Delta\Delta c_k = \Delta c_{k+2} - \Delta c_{k-2}$.

- 1 coeficiente de energía E.
- 1 coeficiente delta de energía ΔE.
- 1 coeficiente delta-delta de energía $\Delta\Delta E$.

En total es un vector de 39 coeficientes por frame de habla.

$$\begin{bmatrix} c_k \\ \Delta c_k \\ \Delta\Delta c_k \\ E \\ \Delta E \\ \Delta\Delta E \end{bmatrix} \tag{4.17}$$

Finalmente el algoritmo para calcular los MFCC se detalla a continiación:

ALGORITMO-MFCC($x_a, T, \|frame\|$)

1 ▷ Digitalización y cuantificación de la señal.
2 $T \leftarrow$ *Periodo de muestreo*
3 $x[n] \leftarrow$ ALGORITMO DIGITALIZACIÓN-CUANTIFICACIÓN(T, x_a)
4 ▷ Pre-énfasis en altas frecuencias.
5 $\alpha \leftarrow 0{,}95$
6 $y[n] \leftarrow$ ALGORITMO PRE-ÉNFASIS(α)
7 Precomputar valor para la ventana hamming
8 $h_h[i] \leftarrow$ ALGORITMO VENTANA-HAMMING ($\| frame \|$)
9 $T = \frac{N}{\|frame\|}$
10 **for** $t \leftarrow 1$ **to** T
11 **do**
12 ▷ Ventaneamiento.
13 $x_t[n] \leftarrow y[n]h_{h_t}[i]$
14 ▷ Análisis con Transformada Corta de Fourier. (Realizado con la FFT. (Sección 2.3)).
15 $X_t(\omega)$= *Algoritmo FFT*($x_t[n]$)
16 ▷ Banco del Filtros Mel.
17 **for** $m \leftarrow 1$ **to** M
18 **do**
19 **for** $k \leftarrow 0$ **to** $\| frame \|$
20 **do**
21 $$H_m[k] = \begin{cases} 0 & \text{si } k < f(m-1) \\ \frac{k-f(m-1)}{f(m)-f(m-1)} & f(m-1) \le k \le f(m) \\ \frac{k-f(m-1)}{f(m+1)-f(m)} & f(m) \le k \le f(m+1) \\ 0 & k > f(m+1) \end{cases}$$
22 $S(m) \leftarrow \ln(\sum_{k=0}^{N-1} |X_t(k)|^2 H_m(k))$
23 ▷ Análisis Cepstral.
24 $c_k(m) \leftarrow \sum_{m=0}^{M-1} S(m)\cos(\pi n(\frac{m-\frac{1}{2}}{M}))$
25 ▷ Coeficientes Cepstrales Dinámicos.
26 $$\vec{o_t} = \begin{bmatrix} c_k \\ \Delta c_k = c_{k+2} - c_{k-2} \\ \Delta\Delta c_k = \Delta c_{k+2} - \Delta c_{k-2} \\ E = |x_s[n]|^2 \\ \Delta E = E_{k+2} - E_{k-2} \\ \Delta\Delta E = \Delta E_{k+2} - \Delta E_{k-2} \end{bmatrix}$$

4.2.8. Tiempo de ejecución del Algoritmo y Criterio de Implementación

El principal criterio de implementación es pre-computar el banco de filtros en escala Mel (lineas 16-21 del Algoritmo para calcular los MFCC). Como el número de filtros M y el tamaño de los frames $\| \ frame \ \|$ son valores fijos para el sistema, (generalmente M=12 o 13 y N=256 o 512).

El análisis de Tiempo de ejecución se realiza teniendo en cuenta el tamaño de la señal de habla discreta. Sea N el tamaño de la señal de habla.

Digitalizar y cuantificar, aplicar el filtro pre-enfasis tiene un tiempo de ejecución determinado por el tamaño de $x[n]$ esto es $\Theta(N)$.

El análisis de la transformada corta de Fourier (lineas 14 del Algoritmo para calcular los MFCC) tiene tiempo de ejecución igual a $\Theta(\| \ frame \ \| \lg \| \ frame \ \|)$ esto es debido al algoritmo para calcular la FFT 2.3.

El análisis cepstral no depende del tamaño de la entrada, si no está en función del número del número de filtros M. El tiempo de ejecución es $\Theta(\| \ M \ \|)$.

El cálculo de los coeficientes ceptrales dinámicos es constante para cada frame su tiempo de ejecución es $\Theta(1)$.

El tiempo de ejecución dominante de cada frame es dado por análisis de la transformada corta de Fourier. El tiempo de ejecución de cada frame es $\Theta(\| \ frame \ \| \lg \| \ frame \ \|)$ que es el tiempo de ejecuciones dominante.

Si la señal de habla tiene T frames, entonces el tiempo de ejecución del cálculo de los MFCC será $\Theta(T \ \| \ frame \ \| \lg \| \ frame \ \|)$.

4.3. Modelado Acústico con Modelos Ocultos de Markov

Los Modelos Ocultos de Markov (HMM)fueron propuestos en una serie de artículos científicos por Baum [Mar67]. En la tesis de Phd, Jim Baker demostró el uso de los HMM en RAH [Bak].

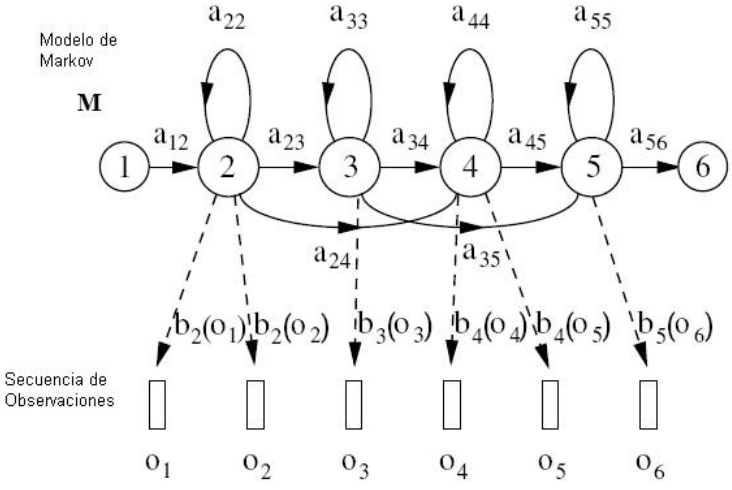


Figura 4.11: *Modelo Oculto de Markov.*

El nombre de modelo oculto de Markov proviene de que éstos modelos son una extensión de las cadenas de Markov que incorporan elementos ocultos en el proceso de modelado.

Un Modelo Oculto de Markov con observaciones discretas es definido como: [Rab90]:

Definición *(HMM-Discreto)* Es un tipo de modelo que modela símbolos discretos que pertenecen a un alfabeto finito

1. N es el número de *estados* del modelo donde S_i denota un estado, para $1 \leq i \leq N$.
   Un estado en un tiempo t es denotado como q_t.

2. M distintos *símbolos de observación* del sistema. Cada símbolo de observación corresponde a la parte observable del HMM. Cada uno de estos símbolos pertenecen a un alfabeto finito y son elementos discretos. Los símbolos de observación son denotados mediante $V = \{v_1, v_2, ..., v_M\}$.
   Los símbolos de observación corresponden a la salida física del sistema que es modelado.

3. $A = a_{ij}$ es la distribución de la probabilidad de las *transiciones* entre estados, donde $a_{ij} = P[q_{t+1} = S_j | q_t = S_i]$. para $1 \leq i, j \leq N$.
   Cada valor a_{ij} representa la probabilidad de ir del estado i al estado j.

4. $B = \{b_j(k)\}$ es una *matriz de probabilidad de salida* de los símbolos de observación en el estado j. Cada valor $b_j(k)$ indica la probabilidad de emitir el símbolo v_k en el estado j.
   Si $O = \vec{o_1}, o2, ...\vec{o_t}$ es la salida observada del HMM. donde cada $\vec{o_t}\ \epsilon\ V$.
   La secuencia de estados $q_1 q_2 ... q_t$ es no observable es decir es oculta.
   El valor de $b_j(k)$ es dado por:$b_j(k) = P[\vec{o_t} = v_k | q_t = S_j]$ para $1 \leq j \leq N$ y $1 \leq k \leq M$

5. $\pi = \{\pi_i\}$ es la distribución inicial de los estados, donde $\pi_i = P(q_1 = S_i)$ para $1 \leq i \leq N$.

En general dando apropiados valores para N, M, A, B y π, el modelo oculto de Markov puede ser visto como una máquina de estado finita probabilística que genera palabras que pertenecen al lenguaje definido por dicha máquina.

Un HMM puede ser utilizado como *generador* para obtener la secuencia de observaciones $O = \vec{o_1}\vec{o_2}...\vec{o_T}$, donde $\vec{o_t}\epsilon V$, y T es el número de observaciones de la secuencia.

Algormitmo HMM-Generador de observaciones

1 $q_1 \leftarrow S_i$ ▷ Escoger estado inicial de acuerdo a π
2 $t \leftarrow 1$
3 $\vec{o_t} \leftarrow v_k$ ▷ Conforme a B
4 $q_{t+1} \leftarrow S_j$ ▷ Conforme a A
5 $t \leftarrow t + 1$ ▷ Retornar a 3 si $t < T$ si no terminar

En general un HMM es denotado como $\lambda(A, B, \pi)$.

Los HMM tienen tres problemas básicos a ser resueltos.

1. **Evaluación**, Dada una secuencia de observaciones $\vec{o_1}\vec{o_2}...\vec{o_T}$ y un HMM $\lambda(A, B, \pi)$, como calcular eficientemente $P(O|\lambda)$.

2. **Decodificación** Dada una secuencia de observaciones $\vec{o_1}\vec{o_2}...\vec{o_T}$ y un HMM $\lambda(A, B, \pi)$, como escoger una correspondiente secuencia de estados $Q = q_1 q_2 ... q_T$, que sea óptima en el sentido de que mejor explique la observación generada.

3. **Aprendizaje-Entrenamiento** Como ajustar los parámetros del modelo $\lambda(A, B, \pi)$, para maximizar $P(O|\lambda)$.

4.3.1. Solución Problema 1: Evaluación

Para calcular la probabilidad de una secuencia de observaciones $\vec{o_1}\vec{o_2}...\vec{o_T}$ dado un modelo $\lambda(A, B, \pi)$. Una manera directa sería enumerar toda la posible secuencia de estados de longitud T.

Consideremos una secuencia fija de estados:

$$Q = q_1 q_2 ... q_T \tag{4.18}$$

donde q_1 es el estado inicial. La probabilidad de la secuencia de observación O para la secuencia de estados anterior esta dada por:

$$P(O|Q, \lambda) = \prod_{t=1}^{T} P(\vec{o_t}|q_t, \lambda) \tag{4.19}$$

La formulación anterior es dada por que los HMM asumen independencia estadística de las observaciones. Luego

$$P(O|Q, \lambda) = b_{q_1}(\vec{o_1}) b_{q_2}(\vec{o_2}) ... b_{q_T}(\vec{o_T}) \tag{4.20}$$

Por otro lado la probabilidad de la secuencia de estados puede ser escrita como:

$$P(Q|\lambda) = \pi_{q_1} a_{q_1 q_2} a_{q_2 q_3} ... a_{q_{T-1} q_T} \tag{4.21}$$

La probabilidad conjunta de las observaciones O y de la secuencia de estados Q es dada por:

$$O(O, Q|\lambda) = P(O|Q, \lambda) P(Q, |\lambda) \tag{4.22}$$

La probabilidad de O dado el modelo λ es obtenida sumarizando la probabilidad conjunta anterior sobre todas las posibles secuencias de estados.

$$P(O|\lambda) = \sum_{Todos los Q} P(O|Q, \lambda) P(Q|\lambda) \tag{4.23}$$

$$= \sum_{q_1, q_2, ..., q_T} \pi_{q_1} b_{q_1}(\vec{o_1}) a_{q_1 q_2} b_{q_2}(\vec{o_2}) ... a_{q_{T-1} q_T} b_{q_T}(\vec{o_T}) \tag{4.24}$$

Si bien la complejidad en espacio no es un problema significativo pues se tendrá en memoria como máximo T estados, la tiempo de ejecución para este caso es exponencial. Para cada $t = 1, 2, ..., T$, existen N posibles estados. Por tanto existen N^T posibles secuencias de estados, donde cada secuencia necesita aproximadamente $2T$ operaciones. El tiempo de ejecución del algoritmo es $\Theta(2T * N^T)$ pues tenemos $2T * N^T$ operaciones. En consecuencia realizar este cálculo directamente no es práctico. Afortunadamente existe un algoritmo llamado *algoritmo forward-backward*, que realiza el mismo cálculo lo mismo en menos operaciones, este es el algoritmo forward-backward.

Consideremos la variable α definida como:

$$\alpha_t(i) = P(\vec{o_1} o2 ... \vec{o_t}, q_t = S_i|\lambda) \tag{4.25}$$

que indica la probabilidad de la secuencia parcial de observaciones $\vec{o_1} o2 ... \vec{o_t}$, hasta el tiempo t y el estado Si en el tiempo t, dado el modelo λ.

El algoritmo puede ser dado inductivamente como sigue:

ALGORITMO FORWARD

1 $\triangleright$ Inicialización
2 $\alpha_1(i) \leftarrow \pi_i b_i(\vec{o_1})$ para $1 \leq i \leq N$
3 $\triangleright$ Inducción
4 $\alpha_{t+1}(j) \leftarrow [\sum_{i=1}^{N} \alpha_t(i) a_{ij}] b_j(o_{t+1})$ para $1 \leq t \leq T-1$, $1 \leq j \leq N$
5 $\triangleright$ Terminación
6 $P(O|\lambda) \leftarrow \sum_{i=1}^{N} \alpha_T(i)$

El tiempo de ejecución del algoritmo forward es $\Theta(N^2 T)$ operaciones.

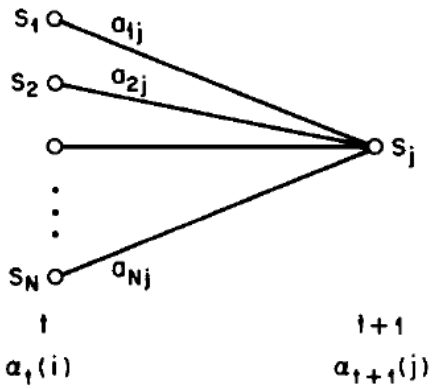


Figura 4.12: *Secuencia de operaciones para el cálculo de la variable forward $\alpha_{t+1}(j)$.*

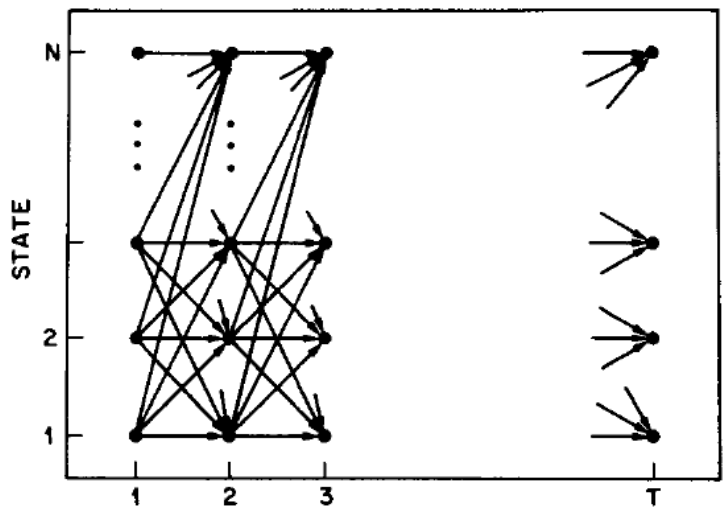


Figura 4.13: *Implementación del cálculo de la variable $\alpha_{t+1}(j)$ en términos de un látice de t observaciones e i estados.*

De la misma manera consideremos la variable β definida como:

$$\beta_t(i) = P(o_{t+1} o t+2 ... \vec{o_T} | q_t = S_i, \lambda) \tag{4.26}$$

que indica la probabilidad de la secuencia parcial de observaciones desde $t+1$ hasta el final, dado el estado S_i en el tiempo t y el modelo λ.

El algoritmo puede ser dado inductivamente como sigue:

ALGORITMO BACKWARD

1 ▷ Inicialización
2 $\beta_T(i) \leftarrow 1$ para $1 \leq i \leq N$
3 ▷ Inducción
4 $\beta_t(j) \leftarrow \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}\beta_{t+1}(j))$ para $1 \leq i \leq N$, $t = T-1, T-2, ..., 1$

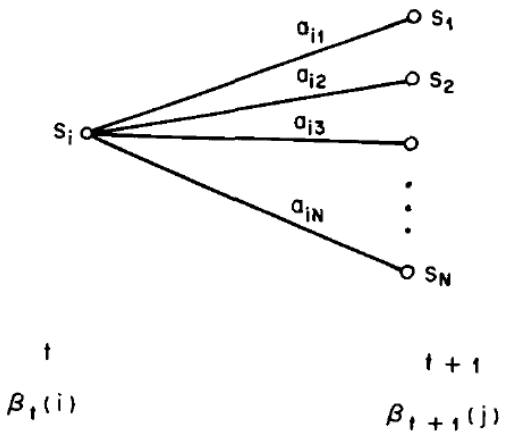


Figura 4.14: *Secuencia de operaciones para el cálculo de la variable backward* $\beta_t(j)$.

La parte backward se utilizará para solucionar el problema de decodificación y la parte forward para el problema de la Evaluación.

4.3.2. Solución Problema 2: Decodificación

Contrariamente al problema de evaluación, existen muchas maneras de solucionar el problema de decodificación. Este problema formula el encontrar una secuencia *óptima* de estados asociadas a una observación dada. La dificultad radica en la definición de secuencia óptima de estados (Pueden haber muchos criterios posibles de optimalidad).

Para encontrar la mejor secuencia de estados $Q = q_1 q_2 ... q_T$ para una secuencia de observaciones dada $O = \vec{o_1}\vec{o_2}...\vec{o_T}$ se define la variable :

$$\delta_t(i) = \max_{q_1 q_2 \cdots q_{t-1}} P[q_1 q_2 ... q_t = i, \vec{o_1} o2 ... \vec{o_t} | \lambda] \tag{4.27}$$

La variable $\delta_t(i)$ representa la mas alta probabilidad a través de un camino, en el tiempo t y finaliza en el estado S_i.

Para recuperar la secuencia de estados, se necesita guardad el argumento que es maximizado para cada t y cada j. Esto se hace con el array $\psi_t(j)$.

El algoritmo de Viterbi se detalla a continuación

ALGORITMO DE VITERBI

1 ▷ Inicialización.
2 $\delta_1(i) \leftarrow \pi_i b_i(o_i)$ para $1 \leq i \leq N$
3 $\psi_1(i) \leftarrow 0$
4 ▷ Recursión.
5 $\delta_t(j) \leftarrow \text{máx}[\delta_{t-1}(i)a_{ij}]b_j(\vec{o_t})$ para $2 \leq t \leq T$, para $1 \leq j \leq N$
6 $\psi_t(j) = argmax[\delta_{t-1}(i)a_{ij}]$ para $2 \leq t \leq T$, para $1 \leq j \leq N$
7 ▷ Terminación.
8 $p^* \leftarrow \text{máx}[\delta_T(i)]$
9 $q_T^* \leftarrow argmax[\delta_T(i)]$
10 ▷ path backtracking.
11 $q_t^* \leftarrow \psi_{t+1}(q_{t+1}^*)$ para $t = T-1, T-2, ..., 1$

El algoritmo de viterbi es similar en criterios de implementación al algoritmo forward, excepto por el backtracking.

4.3.3. Solución Problema 3: Aprendizaje-Entrenamiento

El problema más difícil de los HMM, es determinar los parámetros del modelo $\lambda = (A, B, \pi)$, que maximice la probabilidad de la secuencia de observaciones dado el modelo.

No se conoce aún una manera analítica para realizar esto. De hecho dada cualquier secuencia finita de observaciones como datos de entrenamiento, no existe una manera óptima de estimar los parámetros del modelo.

Sin embargo es posible escoger $\lambda = (A, B, \pi)$, de tal manera que $P(O|\lambda)$ es localmente maximizada, usando para esto diversos procedimientos como el algoritmo Baum-Welch, o usando técnica de gradiente.

Definiremos la probabilidad de empezar en el estado S_i en el tiempo t e ir estado S_j en el tiempo $t+1$, dado el modelo y la secuencia de observaciones como:

$$\begin{aligned} \xi_t(i,j) &= P(q_t = S_i, q_{t+1} = S_j|O, \lambda) \\ &= \frac{\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)}{P(O|\lambda)} \\ &= \frac{\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)}{\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)} \end{aligned} \tag{4.28}$$

Definiremos además otra variable que indica la probabilidad de empezar en el estado S_j en el tiempo t dada la secuencia de observaciones y el modelo:

$$\begin{aligned} \gamma_t(i) &= P(q_t = S_i|O, \lambda) \\ &= \frac{\alpha_t(i)\beta_t(i)}{P(O|\lambda)} \\ &= \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N}\alpha_t(i)\beta_t(i)} \end{aligned} \tag{4.29}$$

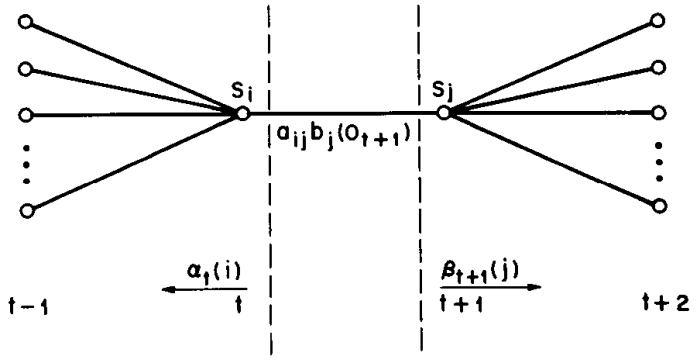


Figura 4.15: *Ilustración del cálculo de* $\xi_t(i,j)$.

donde

$$\Sigma_{i=1}^{N}\gamma_t(i) = 1 \tag{4.30}$$

Relacionando 4.28 con 4.29 tenemos:

$$\gamma_t(i) = \sum_{i=1}^{N}\xi_t(i,j) \tag{4.31}$$

El número esperado de transiciones que parten del estado S_i será:

$$\sum_{t=1}^{T-1}\gamma_t(i) \tag{4.32}$$

El número esperado de transiciones que parten del estado S_i al estado S_j será:

$$\sum_{t=1}^{T-1}\xi_t(i,j) \tag{4.33}$$

Usando esas fórmulas podemos estimar los parámetros de un HMM de la siguiente manera:
Frecuencia esperada de empezar en el estado S_i en el tiempo $t = 1$:

$$\bar{\pi} = \gamma_1(i) \tag{4.34}$$

Número esperado de transiciones del estado del estado S_i al estado S_j:

$$\bar{a_{ij}} = \frac{\sum_{t=1}^{T-1}\xi_t(i,j)}{\sum_{t=1}^{T-1}\gamma_t(i)} \tag{4.35}$$

Número esperado de veces de estar en el estado j y observar el símbolo v_k :

$$\bar{b_j}(k)\frac{\sum_{t=1,\vec{o_t}=v_k}^{T}\gamma_t(j)}{sum_{t=1}^{T}\gamma_t(j)} \tag{4.36}$$

Si definimos el modelo actual como $\lambda = (A, B, \pi)$ y usamos las ecuaciones 4.34-4.36 y obtenemos el modelo re-estimado $\bar{\lambda} = (\bar{A}, \bar{B}, \bar{\pi})$, fué probado por Baum [Mar67] que puede acontecer uno de los dos casos:

1. El modelo inicial λ define un punto crítico en la función de la probabilidad, en este caso $\bar{\lambda} = \lambda$
2. El modelo $\bar{\lambda}$ es más probable que el modelo λ, en este caso $P(O|\bar{\lambda}) > P(O|\lambda)$, esto indica que el

procedimiento anterior encontró un nuevo modelo $\bar{\lambda}$ para la cual la secuencia de observaciones es más probable que haya sido producida.

Basado en los resultados anteriores, si iterativamente calculamos $\lambda \leftarrow \bar{\lambda}$, la probabilidad de que la secuencia de observaciones O haya sido generada por el modelo aumentará hasta que un punto límite sea alcanzado.

Es de resaltar que el algoritmo Baum-Welch apunta alcanzar un máximo local solamente, y en muchos problemas la superficie de optimización es muy compleja y tienes muchos máximos locales.

El RAH utiliza un tipo de HMM llamado modelo izquierda-deracha o modelo Bakis [Rab90]. En donde el cálculo de la función de probabilidad B es una función de distribución de probabilidad en vez de una matriz. Esta función de probabilidad es generalmente una mezcla de funciones gaussianas ponderadas. A continuación definiremos todos estos elementos, así como también el criterio de implementación de estos modelos.

4.3.4. Modelos Izquierda-Derecha

Estos modelos tienen la propiedad de que conforme el tiempo incrementa, el índice asociado a cada estado incrementa o permanece igual. Claramente estos modelos son más adecuados para modelar señales de habla.

Figura 4.16: *HMM izquierda-derecha o modelos Bakis.*

Una propiedad fundamental de este tipo de modelo, es que los coeficientes de transición de estado están dados por:

$$a_{ij} = 0 \quad \text{si } j < i \tag{4.37}$$

Es decir no se permiten transiciones entre estados cuyos índices sean más bajos que el estado actual.

Otra propiedad importante es que la probabilidad inicial de estado están dados por:

$$\pi_i = \begin{cases} 0 & i \neq 1 \\ 1 & i = 1 \end{cases} \tag{4.38}$$

Estos modelos siempre inician el el estado 1 y finalizan en el estado N.

4.3.5. Modelos de Mezclas de Gaussianas

Los modelos acústicos para RAH están basados en calcular una *función de densidad de probabilidad* (pdf) en un espacio continuo.La manera más general de representar los pdf, es usando *modelos de mezclas de gaussianas* (GMM).

Empezaremos definiendo función gaussiana univariante hasta llegar a definir finalmente lo GMM.

Gaussiana Univariante

La distribución Gaussiana, también conocida como la distribución normal, es una función en forma de una campana que es parametrizada por la la media μ y la varianza σ^2.

Definición *(Gaussiana Univariante)* La distribución Gaussiana Univariante de una variable x es dada por:

$$f(x|\mu,\sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} exp(-\frac{(x-\mu)^2}{2\sigma^2}) \tag{4.39}$$

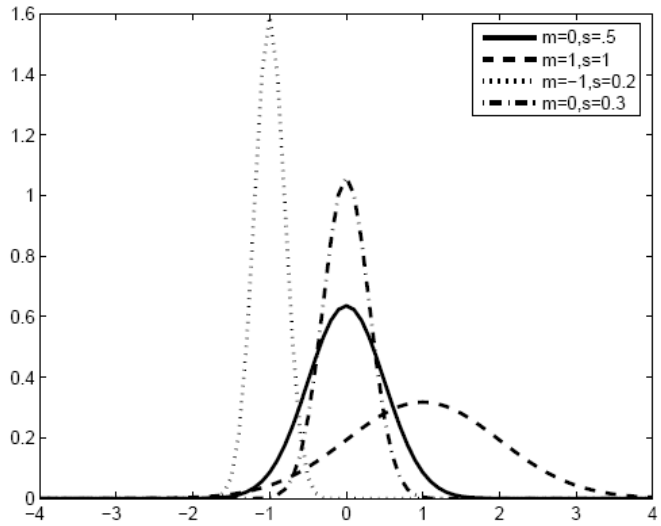


Figura 4.17: *Funciones Gaussianas con diferentes medias y varianzas [JM09].*

La media de una variable aleatoria X, es el valor esperado de X.

$$\mu = E(X) = \sum_{i=1}^{N} p(X_i) X_i \tag{4.40}$$

La varianza de la variable aleatoria X esta dada por:

$$\sigma^2 = E(X - E(X)) = \sum_{i=1}^{N} p(X_i)(X_i - E(X))^2 \tag{4.41}$$

Cuando la función gaussiana es usada como una función de densidad de probabilidad, la area bajo la curva es la unidad.

Gaussiana Multivariante

En RAH se tiene que cada observacion $\vec{o_t}$ es representado por un vector de 39 características (Sección 4.2), para poder asignar una probabilidad a un vector usaremos gaussianas multivariantes con un vector de media $\vec{\mu}$ de dimesión K y matriz de covarianza Σ

Definición *(Gaussiana Multivariante)* La distribución Gaussiana Multivariante de una variable $\vec{x}$ es dada por:

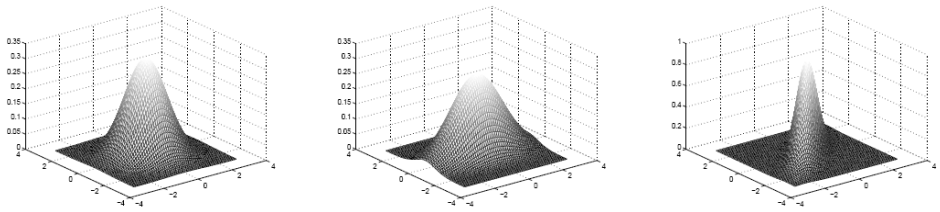
$$f(\vec{x}|\vec{\mu},\Sigma) = \frac{1}{|\Sigma|^{1/2}(2\pi)^{K/2}} exp(\frac{-(\vec{x}-\vec{u})^T\Sigma^{-1}(\vec{x}-\vec{u})}{2}) \tag{4.42}$$

donde la expresión $(\vec{x} - \vec{u})^T$ denota la transpuesta de $(\vec{x} - \vec{u})$. Donde la matriz de covarianza captura la varianza en cada dimensión.

La covarianza de dos variables aleatorias X e Y esta dada por:

$$\Sigma = E(X - E(X))E(Y - E(Y)) \tag{4.43}$$

Cuando no existe correlación entre las varianzas de diferentes dimensiones del vector de características (observación $\vec{o_t}$), se tiene una matriz de covarianza diagonal con elementos cero en cualquier posición excepto en la diagonal. La diagonal de dicha matriz contiene las varianzas de cada dimensión



(a) (b) (c)

Figura 4.18: *Tres Gaussianas Multivariantes en dos dimensiones. a) Gaussiana Multivariante con matriz de covarianza diagonal, con igual varianza en las dos dimensiones. b)Gaussiana Multivariante con matriz de covarianza diagonal, con diferente varianza en las dos dimensiones. c)Gaussiana Multivariante con matriz de covarianza completa.[JM09].*

Las gaussianas con matrices de covarianza completas son lentas de calcular, tienen más parámetros y necesitan más datos de entrenamiento. Es por esto que en los sistemas RAH se usan las gaussianas con matrices de covarianza diagonales.

Una gaussiana Multivariada con matriz de covarianza diagonal está dada por:

$$f(\vec{x}|\vec{\mu}, \Sigma) = \Pi_{k=1}^{K} \frac{1}{\sigma_k \sqrt{2\pi}} exp(-\frac{1}{2}(\frac{x_k - u_k}{\sigma_k})^2) \tag{4.44}$$

donde u_k es el elemento del vector $\vec{\mu}$ en la posición k y σ_k^2 es la varianza en la posición k.

Modelos de Mezclas de Gaussianas

Sirven para modelar distribuciones mas complejas usando varias gaussianas ponderadas Los modelos de mezclas de gaussianas están definidos como

Definición *(Modelos de Mezclas de Gaussianas)*

$$GMM(x) = \sum_{m=1}^{M} c_m \eta(x, u_m, \Sigma_m) \tag{4.45}$$

$$= \sum_{m=1}^{M} c_m \frac{1}{|\Sigma_m|^{1/2}(2\pi)^{K/2}} exp(\frac{-(\vec{x} - \vec{u_m})^T \Sigma_m^{-1} (\vec{x} - \vec{u_m})}{2}) \tag{4.46}$$

Si se considera una matriz de covarianza diagonal, los GMM están dados por

Definición *(Modelos de Mezclas de Gaussianas con covarianza diagonal)*

$$GMM(x) = \sum_{m=1}^{M} c_m \eta(x, u_m, \Sigma_m) \tag{4.47}$$

$$= \sum_{m=1}^{M} c_m \Pi_{k=1}^{K} \frac{1}{\sigma_{km}\sqrt{2\pi}} exp((-\frac{1}{2}(\frac{x_k - u_{km}}{\sigma_{km}})^2)) \tag{4.48}$$

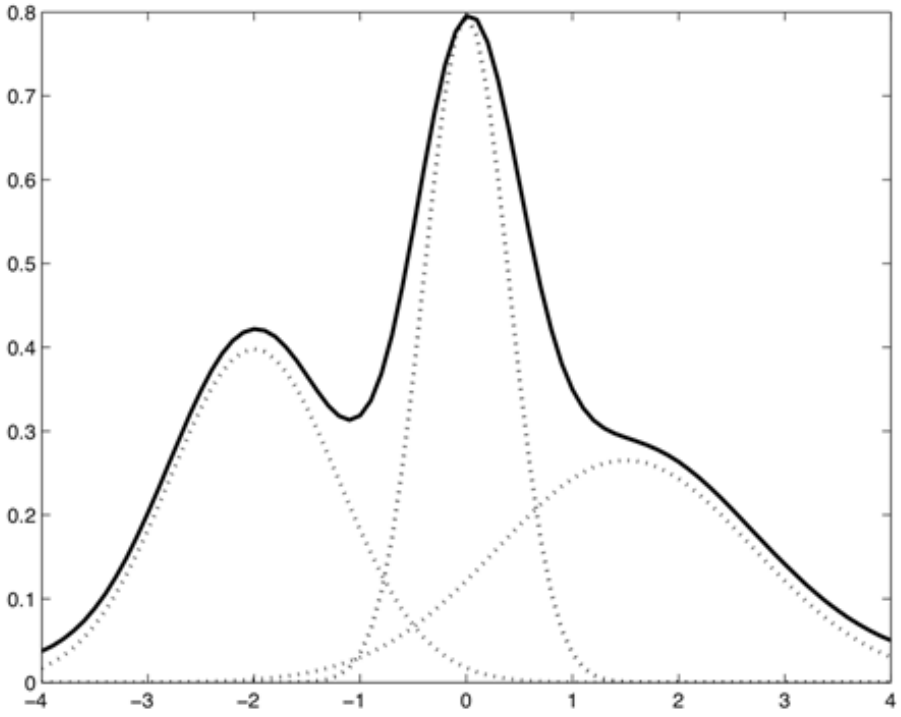


Figura 4.19: *Un Modelo de Mezclas de Gaussianas.[JM09].*

4.3.6. Modelos Ocultos de Markov con Modelos de Mezclas de Gaussianas

Los HMM vistos anteriormente (Seccion 4.3) solo consideran el caso de que las observaciones sean caracterizados por símbolos discretos de un alfabeto. y en consecuencia se puede usar una función de densidad de probabilidad discreta, en cada estado de estos modelos.

El problema con este enfoque es que las observaciones son continuas en la mayoría de los casos.

la manera más general de representar la función de densidad de probabilidad $b_j(\vec{o_t})$ (pdf en el estado j para la observación en el tiempo t), es usando GMM (Seccion 4.3.5)

$$b_j(\vec{o_t}) = \sum_{m=1}^{M} c_{jm}\eta(\vec{o_t}, u_{jm}, \Sigma_{jm}) \tag{4.49}$$

$$= \sum_{m=1}^{M} c_{jm}\frac{1}{|\Sigma_{jm}|^{1/2}(2\pi)^{K/2}}exp(\frac{-(\vec{o_t} - \vec{u_m})^T\Sigma_{jm}^{-1}(\vec{o_t} - \vec{u_{jm}})}{2}) \tag{4.50}$$

$$= \sum_{m=1}^{M} c_m\Pi_{k=1}^{K}\frac{1}{\sigma_{kjm}\sqrt{2\pi}}exp((-\frac{1}{2}(\frac{o_k - u_{kjm}}{\sigma_{kjm}})^2)) \tag{4.51}$$

Donde en este caso se considera una matriz de covarianza diagonal para realizar de una manera mas eficiente el cálculo.

Los HMM utilizados en el RAH utilizan modelos izquierda-derecha con dos estados adicionales llamados *estado inicial* I y *estado final* F, los cuales no tienen asociada la función de distribución de la probabilidad $B = \{b_j(k)\}$. Estos estados sirven para enlazar un modelo con otro. Reformulado las ecuaciones anteriores, el modelo empieza en el estado I e finaliza en el estado F de tal manera que ahora el modelo recorre una secuencia de $T + 2$ estados para generar T observaciones.
El algoritmo forward para RAH es dado a continuación:

Algoritmo Forward para RAH

1 $\triangleright$ Inicialización
2 $\alpha_1(j) \leftarrow a_{Ij}b_j(\vec{o_1})$
3 $\triangleright$ Inducción
4 $\alpha_{t+1}(j) \leftarrow [\sum_{i=1}^{N}\alpha_t(i)a_{ij}]b_j(o_{t+1})$ para $1 < t \leq T$
5 $\triangleright$ Terminación
6 $P(O|\lambda) \leftarrow \alpha_T(F) \leftarrow \sum_{i=1}^{N}\alpha_T(i)a_{iF}$

El algoritmo backward para RAH es dado a continuación:

Algoritmo Backward para RAH

1 $\triangleright$ Inicialización
2 $\beta_T(i) \leftarrow a_{iF}$
3 $\triangleright$ Inducción
4 $\beta_t(j) \leftarrow \sum_{j=1}^{N}a_{ij}b_j(o_{t+1}\beta_{t+1}(j))$ para $t = T-1, T-2, ..., 1$

Algoritmo de Viterbi para RAH

1 $\triangleright$ Inicialización.
2 $\delta_1(j) \leftarrow a_{Ij}b_j(\vec{o_1})$ para $1 \leq j \leq N$
3 $\psi_1(j) \leftarrow 0$
4 $\triangleright$ Recursión.
5 $\delta_t(j) \leftarrow$ máx$[\delta_{t-1}(i)a_{ij}]b_j(\vec{o_t})$ para $2 \leq t \leq T$
6 $\psi_t(j) = argmax[\delta_{t-1}(i)a_{ij}]$ para $2 \leq t \leq T$
7 $\triangleright$ Terminación.
8 $p^* \leftarrow \delta_T(F) \leftarrow$ máx$(\delta_T(i)a_{iF})$
9 $q_T^* \leftarrow argmax[\delta_T(i)]$
10 $\triangleright$ path backtracking.
11 $q_t^* \leftarrow \psi_{t+1}(q_{t+1}^*)$ para $t = T-1, T-2, ..., 1$

El algoritmo Baum-Welch será modificado para hacer la restimación de los parámetros de los GMM

Para el parámetro c_{jk}

$$c_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k)}{\sum_{t=1}^{T} \sum_{k=1}^{M} \gamma_t(j,k)} \tag{4.52}$$

Para el parámetro $u_{jk}^{\rightarrow}$

$$\vec{u} = \frac{\sum_{t=1}^{T} \gamma_t(j,k)\vec{o_t}}{\sum_{t=1}^{T} \sum_{k=1}^{M} \gamma_t(j,k)} \tag{4.53}$$

Para el parámetro Σ_{jk}

$$\Sigma_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k)(\vec{o_t} - u_{jk}^{\rightarrow})(\vec{o_t} - u_{jk}^{\rightarrow})}{\sum_{t=1}^{T} \sum_{k=1}^{M} \gamma_t(j,k)} \tag{4.54}$$

Donde:

$$\gamma_t(j,k) = \frac{\alpha_t(j)\beta_t(j)}{\Sigma_{j=1}^{N} \alpha_t(j)\beta_t(j)} \frac{c_{jk}\eta(\vec{o_t}, u_{jk}^{\rightarrow}, \Sigma)}{\sum_{m=1}^{M} c_{jm}\eta(\vec{o_t}, u_{jm}^{\rightarrow}, \Sigma)} \tag{4.55}$$

4.3.7. Limitaciones de los HMM

Los HMM realizan la siguientes suposiciones en su teoría:

Suposición de Markov

Las probabilidades de las transiciones:

$$a_{ij} = P(q_{t+1} = S_j | q_t = S_i) \tag{4.56}$$

esto es el siguiente estado solo depende del estado actual. A este tipo de suposición se le conoce como suposición de primer orden de Markov. Sin embargo el siguiente estado depende de los k estados anteriores, es decir:

$$a_{ij} = P(q_{t+1} = S_j | q_t = S_{i_1}, q_{t-1} = S_{i_2}, ..., q_1 = S_{i_t}) \tag{4.57}$$

Estos modelos son complejos por incrementar el tiempo de ejecución.

Suposición Estacionaria

Los HMM asumen que las probabilidades de transición entre estados son independientes del tiempo.

$$a_{ij} = P(q_{t_1+1} = S_j | q_{t_1} = S_i) = P(q_{t_2+1} = S_j | q_{t_2} = S_i) \tag{4.58}$$

para cualquier t_1 y t_2.

Suposición de Independencia Condicional

Esta tercera limitación de los HMM establece que las observaciones es estadísticamente independiente de las observaciones previas. Dada una secuencia de observaciones

$$O = o_1, o_2, ..., o_T \tag{4.59}$$

Los HMM suponen lo siguiente:

$$P(O | q_1, q_2, ..., q_T, \lambda) = \prod_{t=1}^{T} P(o_t | q_t, \lambda) \tag{4.60}$$

Duración del Modelo

Los HMM no proveen una adecuada representación de la estructura temporal del habla, pues la probabilidad de permanencia en un estado decrece exponencialmente con el tiempo. La probabilidad de t observaciones consecutivas en el estado i es la probabilidad de estar en el lazo del estado i t veces. Esto es descrito como:

$$d_i = a_{ii}^t = (1 - a_{ii}) \tag{4.61}$$

4.4. Medición de Errores en RAH

Es ampliamente usado de manera empírica la *tasa de error de palabras* WER [†1]. De manera empírica se necesitan mas de 500 sentencias, con 6 a 10 palabras por sentencia. de 5 s 10 diferentes hablantes para estimar de manera segura la tasa de error de reconocimiento. Es ampliamente aceptado por la comunicada científica que si se realiza una reducción de mas del 10 % el nuevo algoritmo.

Existen tres tipos de errores de reconocimiento de palabras en RAH:

- *Sustitución*: una palabra incorrecta fue sustituida por una palabra correcta.
- *Eliminación*: una palabra correcta fue omitida en la sentencia reconocida.
- *Inserción*: una palabra extra fue agregada en la sentencia reconocida.

Considere el siguiente ejemplo en el idioma ingles:

- **Sentencia Correcta**: Did mob mission area of the Copeland ever go to m4 in nineteen eighty one.
- **Sentencia Reconocida**: Did mob mission area ∗∗ the **copy** land ever go to m4 in nineteen **east** one.

Donde los errores por sustitución están en negrita, errores por inserción estan en subrayado, errores por supresión son denotados como ∗∗.

WER puede definirse como sigue:

Definición *(WER)*

$$WER = 100\,\% \times \frac{\#substituciones + \#eliminaciones + \#inserciones}{pcs} \tag{4.62}$$

donde *pcs* indica el número de palabras correctas por sentencia.

En general para medir el WER se necesita comparar las dos secuencias de palabras una a una. Esta alineación de palabras se conoce como el *problema de emparejamiento máximo de subcadenas*, el cual puede ser solucionado usando la técnica de diseño de algoritmos llamada *programación dinámica*.

Sea la cadena de palabras correcta:

$$w_1 w_2 ... w_n \tag{4.63}$$

donde w_i denota la palabra correcta número i en la cadena correcta de palabras.
Sea la cadena de palabras reconocidas:

$$\hat{w}_1 \hat{w}_2 ... \hat{w}_n \tag{4.64}$$

[1†] del inglés *word error rate*

donde $\hat{w}_i$ denota la palabra correcta número i en la cadena de palabras reconocidas.

Sea $R[i,j]$ el error mínimo pro alinear las subcadenas $w_1w_2...w_n$ y $\hat{w}_1\hat{w}_2...\hat{w}_n$.

El alineamiento óptimo y el WER es obtenido mediante el siguiente algoritmo usando programación dinámica:

Algoritmo para calcular WER

1 ▷ Inicialización
2 $R[0,0] \leftarrow 0$
3 $R[i,j] \leftarrow \infty$
4 $B[0,0] \leftarrow 0$
5 ▷ Iteración
6 **for** $i \leftarrow 1$ **to** n
7 **do**
8 **for** $j \leftarrow 1$ **to** m
9 **do**
10 $$R[i,j] = \text{mín} \begin{cases} R[i-1,j]+1 & \text{eliminacion} \\ R[i-1,j-1] & \text{emparejamiento} \\ R[i-1,j-1]+1 & \text{sustitución} \\ R[i,j-1] & \text{inserción} \end{cases}$$
11 $$B[i,j] = \text{mín} \begin{cases} 1 & \text{si es eliminacion} \\ 2 & \text{si es emparejamiento} \\ 3 & \text{si es sustitución} \\ 4 & \text{si es inserción} \end{cases}$$
12
13 ▷ Backtracking y terminación
14 $WER \leftarrow 100\,\% \times R[n,m]/n$
15 camino óptimo hacia atrás$=(s_1, s_2, ..., 0)$
16 donde
17 $$s_1 = B[n,m], s_t = \begin{cases} B[i-1,j] & \text{si } s_{t-1} = 1 \\ B[i,j-1] & \text{si } s_{t-1} = 2 \\ B[i-1,j-1] & \text{si } s_{t-1} = 3 \text{ o } 4 \end{cases}$$

4.5. El Lenguaje Hablado

La presente sección revisa algunos conceptos sobre el lenguaje hablado, en particular se revisarán aquellos conceptos útiles para el reconocimiento automático del habla.

4.5.1. El Habla como Sonido

El sonido es una onda producida por la vibración de moléculas de aire, esta vibración es percibida por el oído y transmitida al cerebro que lo interpreta como un sonido en particular.

la *amplitud* de un sonido mide el grado de desplazamiento de las moléculas de aire a partir de sus posiciones de reposo, debido al amplio rango de valores se utiliza una escala logarítmica para medirla llamada *decibel* (dB) que es definida como:

$$10 \log_{10} \frac{P_1}{P_2}. \tag{4.65}$$

Donde P_1 y P_2 son dos niveles de presión.

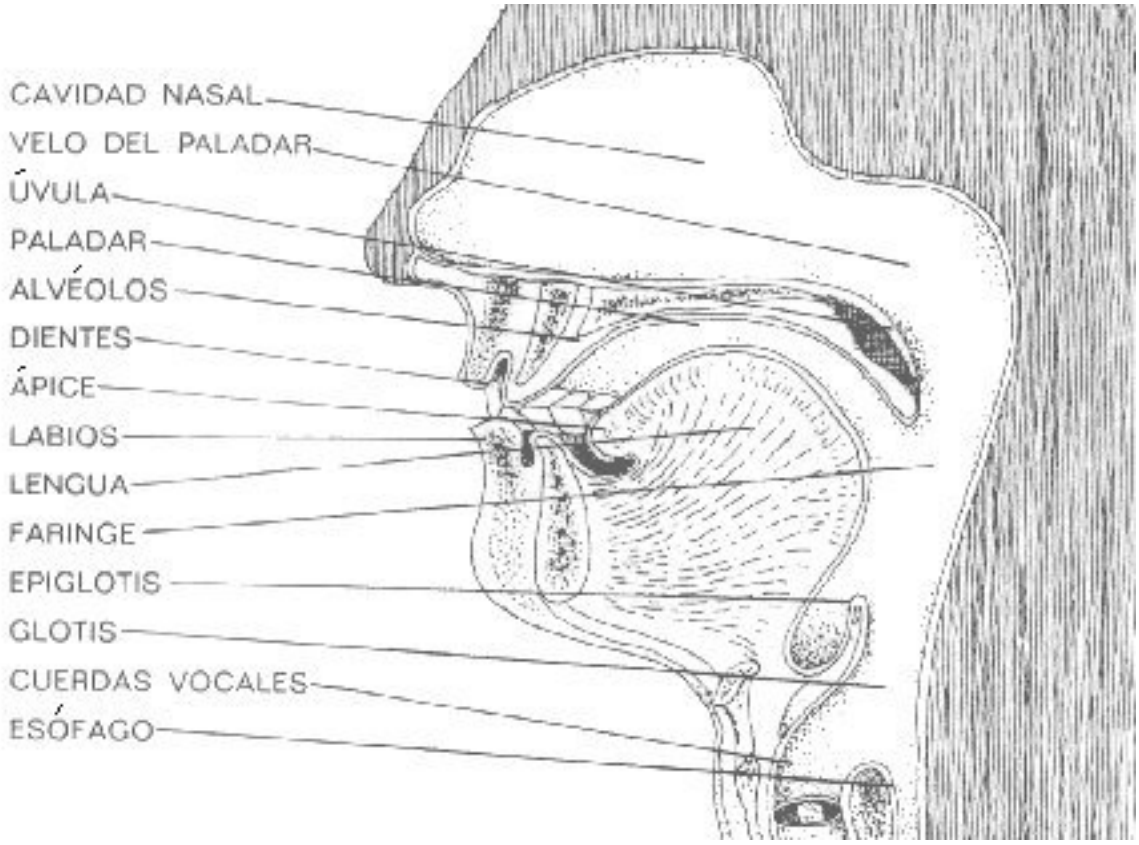


Figura 4.20: *Diagrama del aparato fonador humano.*

La medida absoluta del nivel de presión de un sonido SPL, es una medida absoluta de la presión P del sonido en decibeles:

$$SPL(dB) = 20 \log_{10} \frac{P}{P_0}. \tag{4.66}$$

Donde 0 $dBSPL$ corresponde al umbral de audición del oído humano y que corresponde al valor de $P_0 = 0{,}0002\ \mu Bar$ para 1 Khz. Ejemplo el nivel de conversación normal a 1,5 *metros* es de 3 $dBSPL$, y el sonido producido por un avión es de aproximadamente de 120 $dBSPL$.

El habla es el sonido que emite el aparato fonador humano con información asociada a cierta lengua o idioma.

4.5.2. Producción y Percepción del Habla

En la producción del habla el aire proveniente de los pulmones hace vibrar las cuerdas vocales en la glotis, luego el aire pasa por el paladar que actúa conjuntamente con diversas posiciones la lengua, labios y dientes para formar sonidos de diversas consonantes.

La frecuencia de la vibración de las cuerdas vocales se denomina *frecuencia fundamental*, diversas frecuencias adicionales producidas por la resonancia del tracto vocal acompañan a la frecuencia fundamental. El resultado final es el habla.

En la percepción del habla existen dos componentes : los órganos periféricos auditivos (oídos) y el sistema nervioso auditivo (cerebro).

El oído procesa la información acústica, luego la información resultante es enviada al cerebro a través del nervio auditivo para su procesamiento.

Tipo	Restricción	Autómata
gramática frase-estructura	$\alpha \rightarrow \beta$	Máquina de Turing
gramática sensitiva al contexto	$\|\alpha\| \leq \|\beta\|$	Autómata linear acotado
gramática libre del contexto (CGG)	$A \rightarrow \beta$	Autómata de Pila
gramática regular	$A \rightarrow w$,$A \rightarrow wB$	Autómata finito determinista

Tabla 4.1: *Jerarquía de Chomsky y las correspondientes máquinas que aceptan este lenguaje.*

4.5.3. Fonema y Fono

Fonética es el estudio de la producción, clasificación y transcripción de los sonidos del habla, *Fonología* es el estudio de la distribución de los sonidos del habla en un lenguaje y las reglas tácitas que gobiernan su pronunciación. El término fonema se usa para denotar a cualquier mínima unidad del sonido del habla, en un lenguaje, que sirve para distinguir una palabra de otra [HAH01]. El término fono se refiere a la realización acústica de un fonema en particular, por ejemplo la palabra *dedo*, tiene cuatro fonemas /d/,/e/,/d/,/o/. sin embargo su pronunciación es dada por los fonos /d/,/e/,/δ/,/o/, donde el primer y segundo sonido de la letra /δ/ varía en el grado de obstrucción, pero son similares en una serie de rasgos propios del fonema. Otro ejemplo es el fonema /t/ de la lengua inglesa, tiene dos fonos distintos en las palabras *sat* y *meter*. En un sistema de reconocimiento automático del habla se tiene que tratar las diferencias de estos fonos. De manera particular el reconocimiento automático del habla utiliza los términos fonema y fono de igual manera.

Para representar adecuadamente los fonemas, existe el alfabeto fonético internacional *IPA*, el cual es un sistema de notación fonética desarrollada por la Asociación Fonética Internacional, como una representación estándar del lenguaje hablado [AC99].

4.6. Modelado del Lenguaje

El conocimiento acerca del lenguaje es importante para reconocer y entender el habla. El conocimiento léxico dada por la definición del vocabulario y las pronunciaciones de las palabras es requerido, también es necesario el conocimiento sintáctico y semántico del lenguaje es decir las reglas que determina que secuencias de palabras son gramaticalmente bien formadas y con significado. La presente sección revisa algunos algoritmos del modelado del lenguaje en el reconocimiento automático del habla necesarios para la presente investigación.

4.6.1. Jerarquía de Chomsky

En la teoría de los lenguajes formales una gramática es definida como $G = (V, P, T, S)$, donde V y T son conjuntos finitos de símbolos no terminales y terminales respectivamente, P es un conjunto finito de reglas de producción y S es un símbolo no terminal especial llamado símbolo de inicio. El lenguaje es definido como un conjunto de símbolos terminales consecutivos. Un lenguaje es producido por una gramática al aplicarse las reglas de producción definidas por esta gramática. En la teoría de lenguajes formales cuatro principales clases de lenguajes y sus gramáticas asociadas se encuentran jerárquicamente estructuradas, conociéndose como la jerarquía de Chomsky [AU72].

Donde α y β son cadenas arbitrarias de símbolos V y T, $\alpha \neq \phi$.

4.6.2. Gramática Libre del Contexto

La gramática libre del contexto (CFG) es una estructura muy importante utilizada en los lenguajes de programación y en los lenguajes naturales. las CGF no son solo poderosas para describir la estructura del lenguaje hablado, si no también lo suficientemente restrictivas para tener parsers eficientes para analizar sentencias naturales. Como las CFG muestran un buen compromiso entre

eficiencia de parser y poder de representación de la estructura de un lenguaje ellas son ampliamente usadas en el procesamiento del lenguaje natural.

Formalmente una gramática libre del contexto es una 4-tupla $G = (N, T, P, S)$, donde N y T son conjuntos disjuntos de símbolos terminales y no terminales respectivamente, P es el conjunto de reglas de la forma $A \leftarrow \alpha$, donde $A \in N$, $\alpha \in (N \cup T)^*$ y $S \in T$ es el símbolo inicial. Una *derivación* es un elemento de la forma $\beta A\gamma \Longrightarrow \beta\alpha\gamma$, con β, $\gamma \in (N \cup T)^*$ y $A \longrightarrow \alpha \in P$.

4.6.3. Consideraciones Prácticas

- *Vocabulario* En el reconocimiento automático del habla para cada conjunto de palabras $W = w_1w_2...w_n$ tomadas de un vocabulario se les puede asignar una valor de probabilidad llamada probabilidad apriori que guía el proceso de reconocimiento y es un factor que contribuye en la determinación de la transcripción final de un conjunto de hipótesis parciales.
- *Diccionario de pronunciación* Se debe tener un diccionario de pronunciación por palabra de vocabulario. La pronunciación estará dada por la secuencia fonética de cada palabra es decir la palabra dividida en fonemas
- *Archivos de transcripción* Para cada sentencia grabada se debe tener su archivo de transcripción asociado, cada archivo de transcripción constituye en la secuencia de palabras de la sentencia. Estos archivos de transcripción junto con el diccionario de pronunciación constituirán el punto de partida para el entrenamiento supervisado de los HMM.

4.7. Creación de Fonemas Dependientes del Contexto

En esta sección se explica la creación de fonemas dependientes del contexto, a los cuales les llamaremos *trifonemas*.

4.7.1. Construcción de Trifonemas

Los fonemas por si solos no son suficientes para modelar el habla. Los fonemas dependen del contexto en el que se encuentran. Tomando en cuenta esta característica se define el trifonema como un fonema dentro de un contexto. La notación de los trifonemas es de un simbolo $-$ para indicar contexto por la izquierda y de $+$ para indicar contexto por la derecha, por ejemplo *t-eh+l* para indicar el fonema *eh* cuando se encuentra entre los fonemas *t* y *l*.

El proceso de creación de los trifonemas sigue el siguiente algoritmo.

Algoritmo Creacion de Trifonemas

1 Crear transcripciones de trifonemas a partir de las transcripciones de fonemas
2 Crear HMM por trifonema a partir del HMM del fonema central
3 Reestimar los parámetros del HMM del trifonema usando Baum Welch a partir de las transcripciones de los trifonemas
4 Buscar estados con similaridad acústica para que sean *ligados* (compartan parámetros)

El algoritmo realiza lo siguiente: primero se convierten las transcripciones de los fonemas a trascripciones de trifonemas, donde el conjunto de HMM de trifonemas es creado por copiar el fonema central para luego realizar el proceso de calcular los parámetros del modelo usando el algoritmo Baum Welch.4.3. Finalmente los estados que tengan similaridad acústica son ligados, esto permite que las distribuciones de los estados puedan ser robustamente estimadas [YEG+06].

Las transcripciones de trifonemas puede ser almacenada en un archivo de texto como sigue

```
sil
t+eh
t-eh+l
```

```
eh-l+eh
l-eh+f
eh-f+oh
...
```

Esta técnica hace hace mas robusto el sistema pues ahora se buscan secuencias de tres fonemas en vez de un fonema de manera aislada. Esto reduce la posibilidad de confundir un sonido con otro, pues se busca una secuencia de tres sonidos distintos.

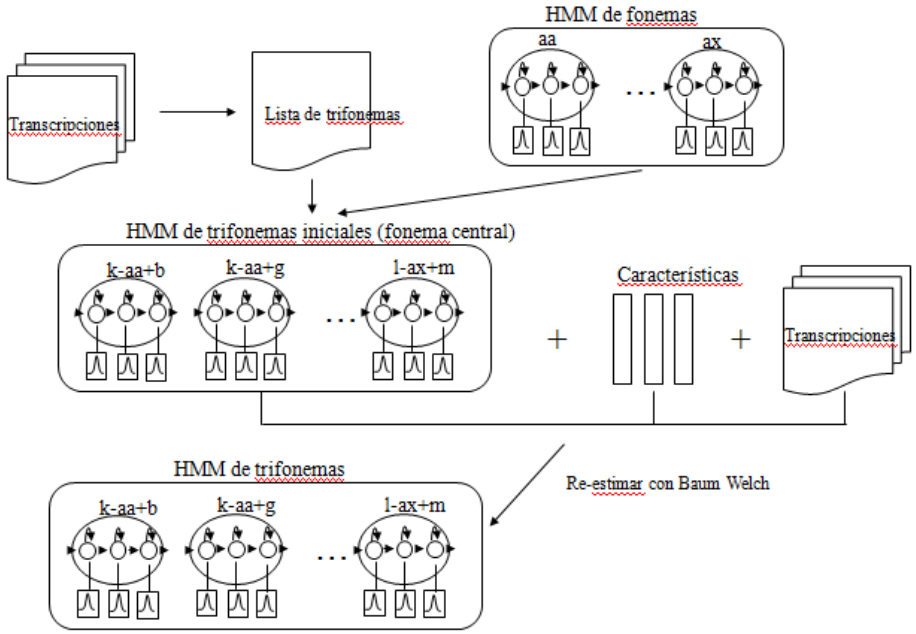


Figura 4.21: *Construcción de los HMM de los trifonemas a partir de los HMM de los fonemas.*

4.7.2. Clustering para Ligar Trifonemas

Un resultado de la creación de trifonemas es que ahora se tiene un número elevado de posibles combinaciones, por ejemplo si antes el lenguaje tenia 50 fonemas, con 3 estados por fonema, usando trifonemas se tienen $3 \times 50_3$ estados. Para reducir el número de los posibles estados del HMM se deben *ligar* algunos parámetros.

Para saber que estados deben ser ligados se debe usar una distancia euclideana entre las medias. Para esto se utiliza el siguiente algoritmo [YEG+06].

CLUSTERING PARA LIGAR TRIFONEMAS(N)

```
1  crear un cluster por estado
2  n ← numero de clusters
3  while n > N
4        do
5           Encontrar i y j para el cual g(i, j) es mínimo
6           Mezclar los clusters i y j
7           n ← n − 1
```

Donde $g(i, j)$ es la distancia entre los clusters i y j definida como la máxima distancia entre cualquier estado del cluster i y cualquier estado del cluster j

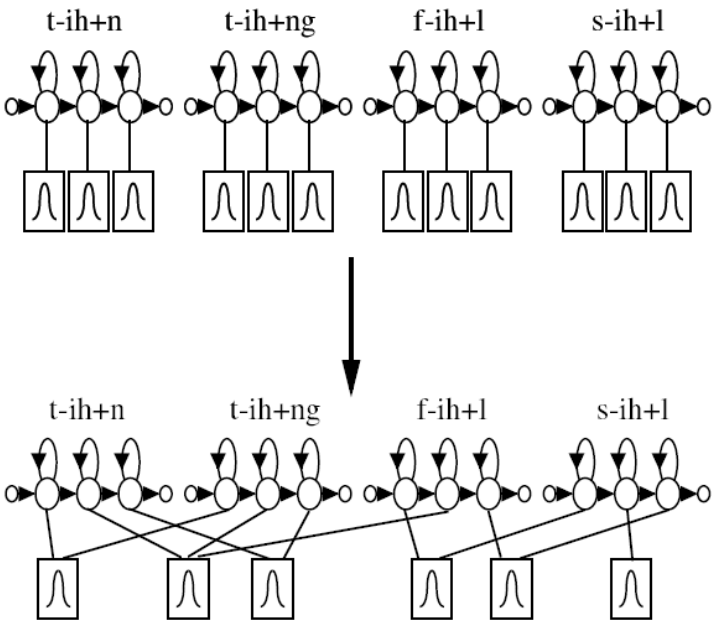


Figura 4.22: *Construcción de los HMM de los trifonemas a partir de los HMM de los fonemas.*

4.8. Decodificación: Algoritmo Token Passing

El proceso de decodificación requiere el grafo de palabras descrito en la sección 6.2.1, el diccionario, y el conjunto de HMM obtenidos de la etapa anterior.

En el grafo de palabras cada nodo es una red de HMM conectados. Este grafo de palabras en si es un HMM. Este grafo puede verse en tres niveles, nivel de palabras, nivel de red y nivel de HMM. A este grafo se le llama grafo de reconocimiento.

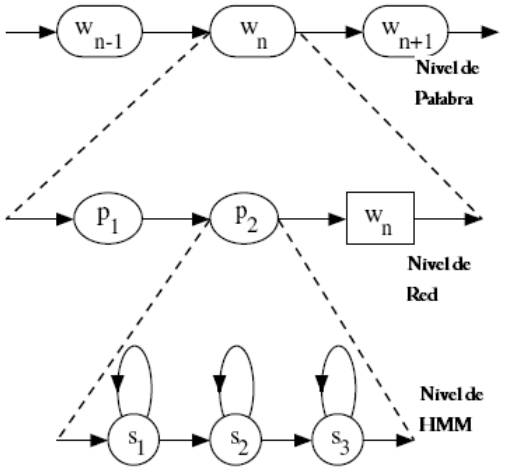


Figura 4.23: *Niveles del grafo de reconocimiento.*

Para una observación O desconocida que contiene T frames, cada camino que empieza en el nodo inicial del grafo y termina en el nodo final y que pasa por exactamente por T estados del

HMM, es una potencial hipótesis de reconocimiento.

El problema de decodificación es encontrar estos caminos a través del grafo de reconocimiento que tenga asociada la probabilidad mas alta. Estos caminos son encontrados usando el algoritmo. *Token Passing* [YRT89]. Un *token* representa el camino parcial a través del grafo de reconocimiento desde el tiempo 0 hasta el tiempo t. En el tiempo 0 el token es ubicado en cada posible estado inicial.

En cada paso los tokens son propagados, si hay muchas salidas de un nodo del grafo los tokens son copiados a todas las posibles salidas y son explorados de manera paralela. Cada vez que los tokens pasan por las transiciones y por los nodos la probabilidad asignada al token se va incrementando.

Un grafo de reconocimiento puede almacenar a lo más N tokens. Al final de cada iteración del algoritmo solo N tokens permanecen. Se debe mantener la historia de recorrido de cada token que será asociada a la palabra reconocida.

4.8.1. Algoritmo Token Passign usando Gramáticas libres del Contexto

Para mejorar la tasa de reconocimientos se debe usar restricciones gramaticales impuestas por el modelo del lenguaje [YRT89] en este caso la gramática libre del contexto que son establecidas a priori. Esto limita las conecciones entre modelos de palabras. La red resultante de aplicar este conocimiento sintáctico es la que se utiliza para encontrar la mejor secuencia de palabras.

Si el lenguaje es definido por un conjunto de reglas de producción de una gramática libre del contexto, las reglas gramaticales son transformadas a un conjunto de redes sintácticas ligadas conforme la gramática.

Los nodos de la red pueden ser de tres tipos: ligadores, terminales y no terminales. Los nodos ligadores son usados para guardar los tokens y son las ubicaciones donde son guardadas los valores precomputados de probabilidad. Los nodos terminales corresponden a modelos de palabras y los nodos no terminales separan las subredes dadas por cada producción gramatical.

A continuación el algoritmo Token Passing con gramática libre del contexto.

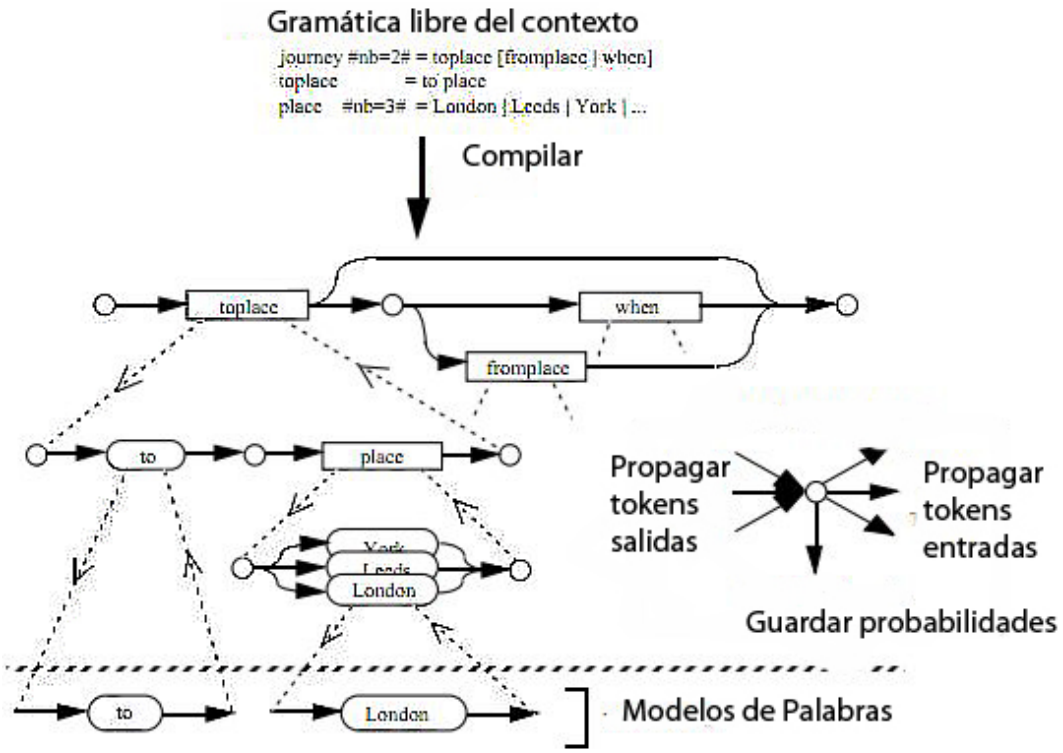


Figura 4.24: *Paso de tokens usando gramática libre del contexto [YRT89].*

ALGORITMO-TOKEN-PASSING-CFG

1 **Inicialización**
2 Almacenar un token con costo $s \leftarrow 0$ en la entrada del nodo raiz
3 Almacenar un token con costo $s \leftarrow \infty$ en los otros nodos
4 **for** $t \leftarrow 1$ **to** T
5 **do**
6 PROPAGAR-TOKENS-ENTRADA(GRAMÁTICA)
7 Copiar los tokens de todos los nodos terminales
en las entradas de los nodos de los correspondientes
modelos de palabras
8 MODELO-PALABRAS-PASO(T)
9 Copiar los tokens de todos los nodos de salida de todos
los modelos de palabras hacia los correspondientes nodos terminales
10 modelos de palabras
11 PROPAGAR-TOKENS-SALIDA(GRAMÁTICA)

12 **Finalización**
13 El token almacenado en el nodo de salida da el mejor matching
del modelo de secuencia de palabras

4.8.2. Penalidad por Inserción de Palabra y Factor de Escala de la Gramática

La penalidad por inserción de palabras es un valor fijo que se agrega a cada token cuando éste va del final de una palabra al inicio de otra. El factor de escala de la gramática es el monto por el cual el modelo de probabilidad del lenguaje es escalado antes de ser agregado a cada token cuando este va del final de una palabra hacia el inicio de la siguiente palabra en la red sintáctica ligada.

Dichos parámetros tienen un efecto significativo en el desempeño del algoritmo, y son establecidos de manera experimental [YEG+06].

Parte II

Material y Métodos

Capítulo 5

Material y Métodos

5.1. Enfoque

Este trabajo sigue el enfoque cuantitativo, pues utiliza técnicas como contar, medir y usar razonamiento abstracto para interpretar los resultados, mas no apreciaciones subjetivas [BE01].

5.2. Hipótesis

La hipótesis planteada fue la siguiente:

Utilizando técnicas de extracción de características de señales de habla asociadas a técnicas de clasificación de patrones, es posible recuperar la información contenida en archivos de audio con grabaciones de textos hablados de una persona en particular

5.3. Tipo de Investigación

Se aplicó el modelo experimental, pues el presente trabajo tiene características fundamentales de dicho modelo.

5.4. Universo y Muestra

El población está definido por todas las posibles sentencias que pertenecen al lenguaje libre del contexto generado por una gramática libre del contexto definida en A.7.

Las muestras fueron elegidas de manera aleatoria [BE01], en la cual todas las sentencias pertenecientes al universo tenían la misma probabilidad de ser elegidos La muestra está conformada por 200 grabaciones, cada una de las cuales contienen una sentencia con el texto hablado generado aleatoriamente por la gramática libre del contexto definida en A.7. Cada sentencia contiene en promedio 15 palabras, esto hace un total de 4000 palabras contenidas en el total de grabaciones.

Las grabaciones fueron realizadas del habla de una persona del sexo masculino de 30 años de edad, hablante de la lengua castellana.

5.5. Instrumentos

Para la recolección datos se usó:

- Un micrófono marca Shure modelo C606 semiprofesional
- Una computadora portátil marca Hp Pavilion dv2700 centrino, procesador Intel(R) Core(TM)2 Duo CPU T5850 @ 2.16GHz, 2167 Mhz, 2 procesadores, 2.00 GB de RAM
- Para la experimentación se implementó o usó el siguiente software.
  - *Lorito* [GD07], software necesario para procesar y analizar la señal.
  - *Extractor MFCC*, software para extraer las características MFCC de los archivos de audio.
  - *LectorAudio*, software para leer archivos de audio de memoria externa, llevarlas a memoria RAM y transformar los valores a una estructura de dato tipo arreglo.
  - *Archivos de configuración* Diversos programas en perl de configuración.
  - *HTK* HMM toolkit.

5.6. Procedimiento

Fue utilizado el método científico, a continuación se describe la secuencia de la presente investigación.

1. Revisión bibliográfica.
2. Delimitación del problema.
3. Elaboración de la realidad problemática.
4. Formulación de la hipótesis.
5. Elaboración del software para experimentación.
6. Recolección de datos.
7. Experimentación.
8. Documentación del informe.

5.7. Métodos y Procedimientos para la Recolección de Datos

La recolección de datos fue realizada de la siguiente manera: Se generaron 200 sentencias aleatorias en formato texto a partir de una gramática libre del contexto. Luego se solicitó a una persona de sexo masculino a leer de manera continua cada una de las sentencias. Cada sentencia fue grabada utilizando los instrumentos descritos en 5.5. Posteriormente para cada sentencia se generó un archivo en formato WAV. El nivel del ruido fue moderado.

5.8. Análisis Estadístico de los Datos

Se evaluó experimentalmente el algoritmo usando validación cruzada k-fold [DHS01], usando como medida de error la medida *word error rate-WER* (Sección 4.4) que es una técnica utilizada de manera común en el reconocimiento automático del habla. Los detalles de esta evaluación experimental se encuentran detallados en la Sección 7.3.

Fue utilizada la prueba de t-student con un nivel de significancia de $\alpha = 0{,}05$ pues no se conoce la media ni la varianza de la población pero se tiene información del error medio y la varianza obtenido por el algoritmo propuesto en las muestras [BE01]. Los detalles de la prueba se encuentran descritos en la Sección 7.4.

Fueron calculadas el porcentaje de palabras reconocidas, los detalles de estos cálculos se encuentran en la Sección 7.1 y en la Sección 7.2.

Capítulo 6

Recuperación de Información Usando RAH

La recuperación de información de textos hablados es la transcripción de archivos con habla hacia una secuencia de palabras usando un computador [HAH01]. En este contexto el término recuperación significa transcripción de los archivos de audio, información significa secuencia organizada de caracteres en palabras con algún significado y textos hablados son grabaciones de audio co habla realizada por una persona en particular.

Este capítulo describe el algoritmo propuesto para recuperar información en textos hablados dependientes del hablante donde cada texto es generado de manera aleatoria por una gramática libre del contexto usando los conceptos vistos en el Marco Teórico. Se tomará como ejemplo una base de datos de archivos de audio de habla que contiene sentencias generadas por una gramática, en este caso sentencias con algunos nombres y números.

En la Sección 6.1 se describen los pasos del algoritmo de manera general. La Sección 6.2 se describe la preparación de los datos. La Sección 6.3 describe la construcción de los HMM. La Sección 6.4 describe la creación de los fonema dependientes del contexto. La Sección 6.5 describe el proceso de decodificación con el algoritmo Token Passing. La Sección 6.6 describe de manera formal el algoritmo propuesto.

6.1. Descripción del algoritmo

A continuación se describen de manera general los pasos del algoritmo y su relación con el fundamento teórico descrito anteriormente. El algoritmo propuesto consta de 4 partes principales: Preparación de los datos, construcción de los HMM, creación de fonemas dependientes del contexto y la decodificación.

DESCRIPCIÓN GENERAL DEL ALGORITMO-DE-RECUPERACIÓN-DE-INFORMACION-EN-TEXTOS-HABLADOS

▷ *Preparación de los datos*
1 Diseño de la gramática de trabajo (Teoría en las Secciones 4.6, 4.6.1, 4.6.2)
2 Construcción del diccionario de pronunciación (Teoría en las Secciones 4.6.3)
3 Obtención de los datos (Teoría en Sección 2.1)
4 Construcción de los archivos de transcripción (Teoría en Sección 4.6.3)
5 Extracción de características usando MFCC (Teoría en las Secciones 2, 3, 4.2)
▷ *Construcción de los HMM*
6 Definición de la estructura del HMM (Teoría en las Secciones 4.3.4, 4.3.5, 4.3.6)
7 Estimación de los parámetros iniciales del HMM
8 Entrenamiento de los HMM (Teoría en Sección 4.3.3)
9 Establecer HMM para silencio y pausa (Teoría en Sección 4.3.3)
▷ *Creación de fonemas dependientes del contexto*
10 Creación de los trifonemas (Teoría en Sección 4.7.1)
11 Clustering de estados (Teoría Sección 4.7.2)
▷ *Decodificación*
12 Algoritmo Token-passing (Teoría Sección 4.8)

A continuación se describe cada parte de manera detallada, luego se describirá formalmente el algoritmo.

6.2. Preparación de los Datos

El primer paso es la preparación de los datos, es necesario archivos con audio de habla para el entrenamiento y test, una gramática libre del contexto y un diccionario de pronunciación fonética.

6.2.1. Construcción de la Gramática

La gramática libre del contexto (Sección 4.6.2) define el lenguaje a procesar. Se usó la notación BNF-extendida. Para ilustrar el algoritmo propuesto se utilizará la siguiente gramática:

```
S       -> (SENT-START(TELEFONO <$digit>| (LLAMAR|MARCAR)$name)SENT-END)

$digit -> UNO | DOS | TRES | CUATRO | CINCO
          | SEIS | SIETE | OCHO | NUEVE | CERO;

$name   -> [JORGE] LUIS |
           [LEISSI] LEON |
           [CARLOS] DIAZ |
           [LUIS] GUEVARA;
```

donde | denota alternativas, [] denota opcionalidad, {} denota cero o más repeticiones, <>denota una o mas repeticiones y → denota regla de producción.

Esta gramática genera un *grafo de palabras*, donde cada vértice del grafo corresponde a una palabra y las aristas del grafo modelan la transición entre palabras. (Fig 6.10).

La gramática es procesada y transformada a formato SLF. (Standard Lattice Format), que es una notación de bajo nivel en la cual cada palabra y cada transición entre palabras es listada de manera explícita y sirve para representar al grafo de palabras que la gramática genera [YEG+06].

El algoritmo de procesamiento de la gramática para la creación del grafo de palabras se detalla a continuación:

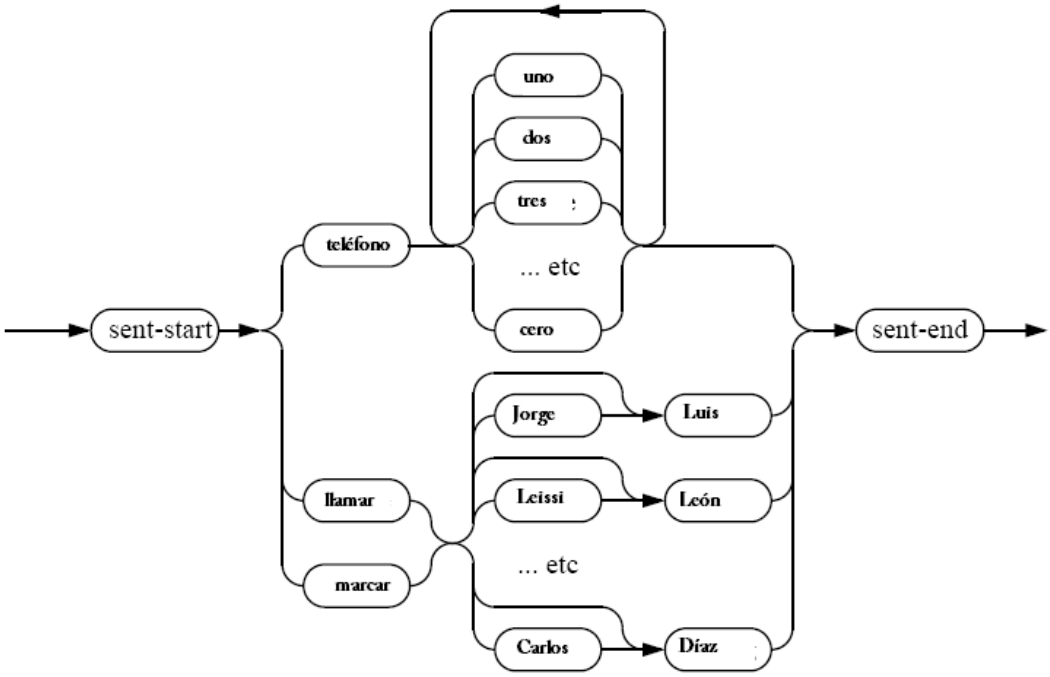


Figura 6.1: *Grafo de palabras correspondiente a la gramática anterior.*

ALGORITMO-PROCESAMIENTO-GRAMÁTICA (GRAMÁTICA GLC G)

▷ Construcción de los nodos
▷ Para cada símbolo de G
1 $i \leftarrow 0$
2 **for** cada X en G
do
3 **if** X =símbolo terminal
then
▷ Crear Nodo
4 $I_i \leftarrow$Crear-Nodo(X, $Palabra$)
5 $i \leftarrow i + 1$

▷ Construcción de las aristas
6 **for** $i \leftarrow 0$ **to** N
do
7 **for** $j \leftarrow 0$ **to** N
do
8 **if** Palabra(I_i) está antes de Palabra(I_j) en G
then
9 $J_{i,j} \leftarrow$Crear-Arista(I_i,I_j)

10 **return** Grafo(I,J)

El algoritmo anterior crea un nodo para cada palabra (símbolo terminal) de la gramática y luego genera las aristas según las transiciones de las palabras que define la gramática. El resultado del procesamiento de la gramática anterior se muestra a continuación en formato SLF.

```
# Define el tamaño del grafo: N=número de nodos, L=numero de arcos.
            N=28   L=55
# Lista de nodos: I=numero del nodo, W=palabra
            I=0    W=SENT-END
            I=1    W=GUEVARA
            I=2    W=!NULL
            I=3    W=LUIS
            I=4    W=DIAZ
            I=5    W=CARLOS
            I=6    W=LEON
            I=7    W=LEISSI
            .      .
            .      .
            .      .
            I=26   W=!NULL
            I=27   W=!NULL
# Lista de arcos: J=numero de arco, S=nodo inicial, E=nodo final
            J=0     S=2    E=0
            J=1     S=14   E=0
            J=2     S=3    E=1
            J=3     S=11   E=1
            .      .
            .      .
            .      .
            J=53    S=27   E=25
            J=54    S=0    E=26
```

De manera general cualquier lenguaje generado por una gramática libre del contexto puede ser establecido.

6.2.2. Construcción del Diccionario de Pronunciación

El siguiente paso es crear un diccionario de pronunciación, este diccionario contiene las palabras establecidas por la gramática con su respectiva pronunciación. La pronunciación de cada palabra está dada por la transcripción fonética de cada palabra; para este trabajo se adoptó el formato IPA [AC99]. A continuación se muestra el diccionario de pronunciación para las palabras definidas por la gramática anterior.

```
            CARLOS          k ah r l oh s sp
            CERO            th eh r oh sp
            CINCO           th ih ng k oh sp
            CUATRO          k w ah t r oh sp
            DIAZ            dh ih ah s sp
            DOS             dh oh s sp
            GUEVARA         g eh b ah r ah sp
            JORGE           j oh r j eh sp
            LEISSI          l eh y s ih sp
            LEON            l eh oh n sp
            LLAMAR          ll ah m ah r sp
            LUIS            l uh y s sp
            MARCAR          m ah r k ah r sp
            NUEVE           n w eh b eh sp
            OCHO            oh ch oh sp
            SEIS            s eh y s sp
            SENT-END   []   sil
            SENT-START []   sil
            SIETE           s y eh t eh sp
```

```
silence         sil
TELEFONO        t eh l eh f oh n oh sp
TRES            t r eh s sp
UNO             uh n oh sp
```

Los fonemas sp y sil denotan silencio.

El paso siguiente es extraer los fonemas del diccionario.

```
k
ah
r
l
oh
s
sp
.
.
.
sil
f
```

De manera genérica se puede trabajar con un diccionario de pronunciación mas grande como el diccionario de pronunciación BEEP que es un diccionario de mas de 250000 palabras con sus respectivas pronunciaciones [Rob10]. Lamentablemente aún no existe un diccionario de pronunciación en castellano disponible de manera digital que siga la terminología estándar IPA.

El algoritmo de construcción del diccionario de pronunciación es el siguiente:

ALGORITMO-CONSTRUCCIÓN-DICCIONARIO-PRONUNCIACIÓN (DICCIONARIO D, GRAMÁTICA GLC G)

▷ Conjunto de Fonemas
$S \leftarrow \phi$
▷ Para cada símbolo terminal de G
for a en G
do
    ▷ Extraer transcripción fonética de la palabra
    $t \leftarrow$ TranscripciónFonética(D,a)
    ▷ agregar nuevos fonemas encontrados
    $S \leftarrow S \cup t$
    ▷ escribir
    $DIC \leftarrow$ **write**(a,t)

return DIC

6.2.3. Obtención de los Datos

El siguiente paso es grabar los datos de entrenamiento. Para esto se generan aleatoriamente sentencias a partir de la gramática establecida y luego se graba cada sentencia. Las grabaciones se realizaron con el software HSLab [YEG+06].

Como ilustración se muestran las sentencias aleatoriamente generadas de acuerdo con la gramática y el diccionario utilizado en las dos secciones anteriores.

```
1. TELEFONO DOS SEIS CUATRO TRES UNO OCHO DOS CINCO CINCO CUATRO UNO SEIS CUATRO
   SEIS CERO CINCO SIETE CUATRO NUEVE CERO
2. LLAMAR LUIS GUEVARA
```

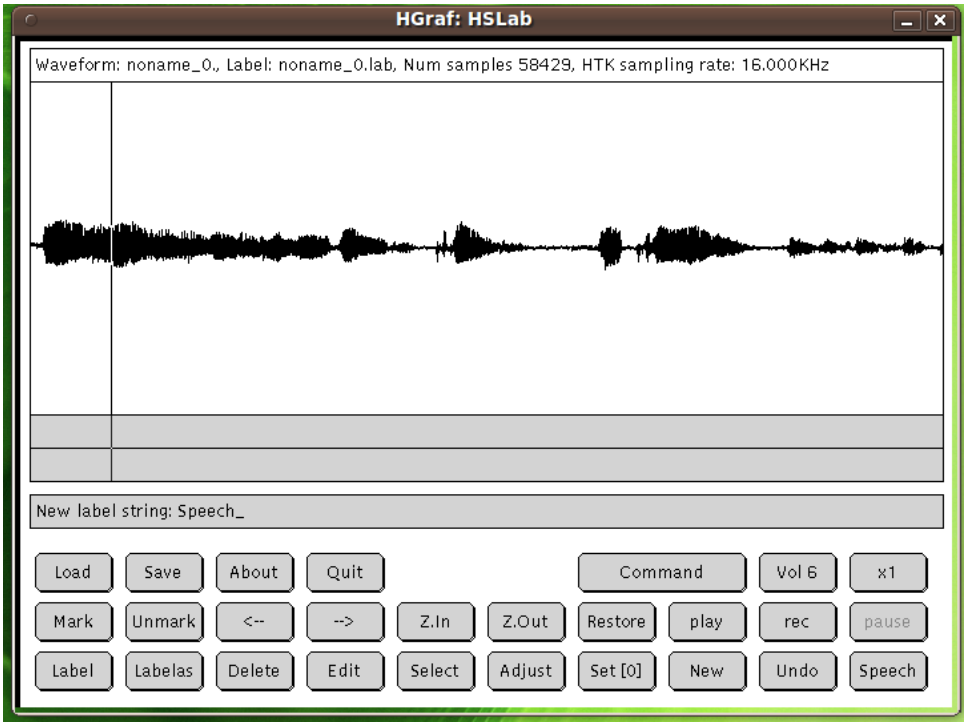


Figura 6.2: *Software HSLab para grabar las sentencias.*

```
3. TELEFONO SEIS SIETE SIETE UNO UNO DOS DOS UNO SIETE CINCO CUATRO OCHO
.
.
.
200. TELEFONO SEIS CUATRO CERO DOS UNO
```

Para generar las sentencias de manera aleatoria se usó el algoritmo propuesto por Purdom [Pur72], [ZW09], el cual describe un algoritmo para generar sentencias de gramáticas libres del contexto. Las sentencias son generadas empezando por el símbolo inicial y reescritas en cada paso usando el símbolo no terminal mas a la izquierda en la derivación usando como estructura de datos una pila.

ALGORITMO-GENERACIÓN-ALEATORIA-SENTENCIAS (GRAMÁTICA GLC G)

```
    ▷ S_0 es el símbolo inicial
 1  Pila.push(S_0)
 2  while (Not)Pila.vacia
        do
 3          s ← Pila.pop()
 4          if s es terminal
              then
 5                  sentencia.add(s)
 6                  print s
 7            else
 8                  ▷ Escoger una regla de manera aleatoria para s
 9                  p = s ⟶ α
10                  Pila.push(reverso(α))
11
12
13  return sentencia
```

6.2.4. Construcción de los Archivos de Transcripción

Cada archivo de audio de los datos de entrenamiento debe tener asociado una transcripción a nivel de fonemas, esto se logra convirtiendo las sentencias aleatoriamente generadas a una secuencia de fonemas de acuerdo al estándar IPA. Estos archivos de transcripción fonética de cada sentencia son necesarios para el entrenamiento posterior de los modelos HMM .

La idea del algoritmo es separar cada sentencia en un conjunto de palabras y posteriormente a un conjunto de fonemas. El algoritmo se describe a continuación:

ALGORITMO-CONSTRUCCIÓN-ARCHIVOS-TRANSCRIPCIÓN (SENTENCIA S, DICCIONARIO D)

▷ Conjunto de Palabras
1 $P \leftarrow \phi$
▷ Para cada palabra w en la sentencia S
2 **for** w en S
3 **do**
4 $P \leftarrow P \cup w$
5
6 archivoTranscripción ←**write**(P,extraerFonema(P,D))

7 **return** archivoTranscripción

```
#!MLF!#
"*/T0001.lab"
TELEFONO
DOS
SEIS
CUATRO
.
.
.
NUEVE
CERO
.
"*/T0002.lab"
LLAMAR
LUIS
GUEVARA
.
"*/T0003.lab"
.
.
.
"*/T0199.lab"
LLAMAR
LEON
.
"*/T0200.lab"
TELEFONO
SEIS
CUATRO
CERO
DOS
UNO
```

Posteriormente se convierte a nivel de fonema toda la secuencia de palabras obtenidas en el nivel anterior (Ver A.4.1).

6.2.5. Extracción de Características usando MFCC

La extracción de características de los archivos de audio es realizado usando el algoritmo MFCC descrito en la Sección 4.2. Se deben establecer valores adecuados para la tasa de muestreo, para el valor α del filtro pre-enfasis, para el tamaño y desplazamiento de la ventana de análisis. Por ejemplo 16000 Hz,$\alpha = 0{,}97$, 25 ms y 10 ms respectivamente.

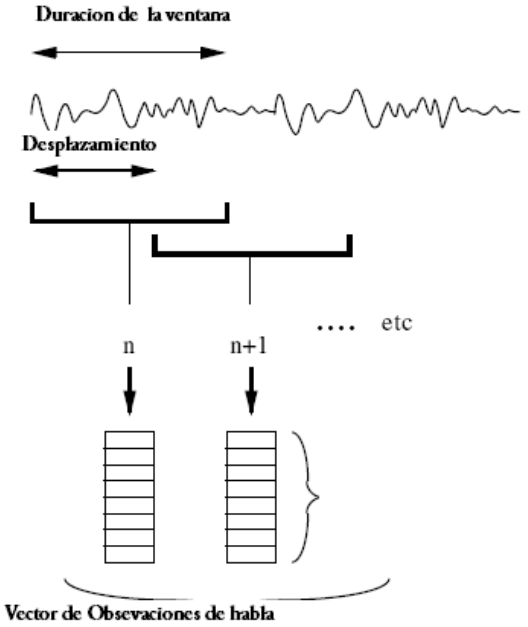


Figura 6.3: *Extracción de características*

Para esta paso se usa el ALGORITMO-MFCC($x_a, T, \|frame\|$) descrito en la Sección 4.2, donde x_a representa el archivo de audio, T es el periodo de muestreo, y $\|frame\|$ es la longitud del frame.

6.3. Construcción de los HMM's

A continuación se deben crear HMM's por cada fonema. La topología a usar son modelos izquierda-derecha de 5 estados [YEG+06], donde el primer y el ultimo estado no tienen asociadas secuencias de observaciones, esto permite modelar con tres estados cada fonema. Para modelar la distribución de los datos por estado se usó modelos de mezclas de gaussianas GMM 4.3.5.

6.3.1. Definición de la Estructura de los HMM

Cada HMM es definido mediante los siguientes parámetros:

- Numero de estados
- Para cada estado definir GMM, con las medias y matrices de covarianza asociadas.
- Matriz de Transiciones

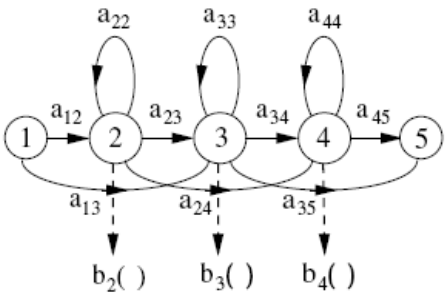


Figura 6.4: *Topología de los HMM para cada fonema*

Por ejemplo la definición de la topología de un HMM para el fonema *ah* tiene 5 estados, cada estado tiene un GMM con 9 gaussianas que tienen asociadas un vector de medias y una matriz de covarianzas diagonales. Se puede definir dicha configuración en un archivo como sigue:

```
~h "ah"
<BEGINHMM>
   <NUMSTATES> 5
   <STATE> 2 <NumMixes> 9
      <Mixture> 1 0.4
         <MEAN> 39
          -1.376990e+01 2.798773e+00 ... 5.959347e-04
         <VARIANCE> 39
           4.839668e+01 1.709005e+01 ... 7.078791e-02
           .
           .
           .
      <Mixture> 9 0.01
         <MEAN> 39
          -1.376990e+01 2.798773e+00 ... 5.959347e-04
         <VARIANCE> 39
           4.839668e+01 1.709005e+01 ... 7.078791e-02
   <STATE> 3 <NumMixes> 9
      <Mixture> 1 0.4
         <MEAN> 39
          -1.376990e+01 2.798773e+00 ... 5.959347e-04
         <VARIANCE> 39
           4.839668e+01 1.709005e+01 ... 7.078791e-02
           .
           .
           .
      <Mixture> 9 0.01
         <MEAN> 39
          -1.376990e+01 2.798773e+00 ... 5.959347e-04
         <VARIANCE> 39
           4.839668e+01 1.709005e+01 ... 7.078791e-02
   <STATE> 4 <NumMixes> 9
      <Mixture> 1 0.4
         <MEAN> 39
          -1.376990e+01 2.798773e+00 ... 5.959347e-04
         <VARIANCE> 39
           4.839668e+01 1.709005e+01 ... 7.078791e-02
           .
           .
           .
      <Mixture> 9 0.01
```

```
        <MEAN> 39
         -1.376990e+01 2.798773e+00 ... 5.959347e-04
        <VARIANCE> 39
          4.839668e+01 1.709005e+01 ... 7.078791e-02
  <TRANSP> 5
   0.000000e+00 1.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
   0.000000e+00 6.000000e-01 4.000000e-01 0.000000e+00 0.000000e+00
   0.000000e+00 0.000000e+00 6.000000e-01 4.000000e-01 0.000000e+00
   0.000000e+00 0.000000e+00 0.000000e+00 7.000000e-01 3.000000e-01
   0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
<ENDHMM>
```

Para evitar crear cada uno de los HMM por cada fonema, se debe crear un HMM prototipo, este servirá para construir los demás HMM. Los parámetros iniciales de este HMM prototipo no son importantes pues cada HMM actualizará sus parámetros iniciales de acuerdo a los datos de entrenamiento. En A.5 se describe la configuración del HMM prototipo.

6.3.2. Estimación de los Parámetros Iniciales de los HMM

Luego de definir la topología de los HMM, cada uno de los parámetros iniciales de los HMM debe ser definido a partir de los datos de entrenamiento. El principal inconveniente en este paso del algoritmo es que la inicialización de los HMM por fonema requiere tener los datos de entrenamiento etiquetados con la posición de los fonemas. Una manera es realizar un etiquetado manual, esto es posible cuando se tienen pocos fonemas. Otra solución es usar la técnica de inicialización descrita en [YEG+06] conocida como *flat start* que lo que hace es establecer todos los estados de los modelos iguales. La primera iteración segmenta los datos de manera uniforme, luego establece la media y la covarianza de los HMM a la media y covarianza global de los datos de entrenamiento.

ALGORITMO-ESTIMACIÓN-PARÁMETROS-INICIALES-HMM (HMM hmm, DATOSENTRENAMIENTO E)

▷ Calcular media y covarianza de los datos de entrenamiento
$\Sigma \leftarrow covarianza(E)$
$\mu \leftarrow media(E)$
▷ Calcular numero de estados de todos los HMM's
$N \leftarrow length(hmm)$
for $i \leftarrow 1$ **to** N
    do
        ▷ Establecer la media y covarianza de todos los estados a la media y covarianza global
        $hmm_i \leftarrow \mu$
        $hmm_i \leftarrow \Sigma$

▷ Segmentar los datos de manera uniforme
$segmento \leftarrow \frac{length(E)}{N}$
▷ Asignar cada segmento a cada estado del HMM
for $i \leftarrow 1$ **to** N
    do
        $hmm_i \leftarrow [E_i ... E_{i,i+segmento-1}]$

return hmm

6.3.3. Entrenamiento de los HMM

Para entrenar los HMM, se usaron todos los datos de entrenamiento con sus correspondientes transcripciones (A.4 y A.4.1). El entrenamiento fue realizado como lo describe el siguiente algoritmo:

ALGORITMO-ENTRENAMIENTO-HMM'S(HMM h, DATOSENTRENAMIENTO E, ARCHIVOSTRANSCRIPCIÓN)

```
1  P ←Read(archivoTranscripción)
2  for e ← 0 to E − 1
3        do
4           h[e] ←Algoritmo Baum-Welch(h[e],e,P_e)
5  retornar a la linea 2 si fuera necesario
6  return h
```

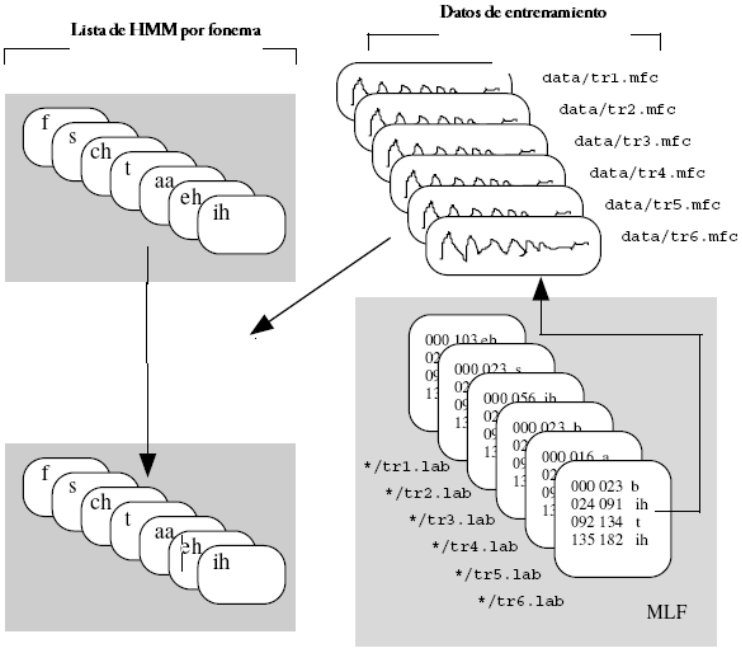


Figura 6.5: *Proceso de entrenamiento de los HMM.*

6.3.4. Establecer HMM para silencio y pausa

El sistema incrementa el porcentaje de reconocimientos si el silencio también es modelado. El fonema *sil* que identifica al silencio es modificado siguiendo el siguiente esquema: agregar transiciones del estado 2 al estado 4 y del estado 4 al estado 2 [YEG+06]. Esto es realizado para que el modelo maneje mejor los ruidos impulsivos presentes en los datos de entrenamiento. La transición del estado 4 a 2 es creada para que el modelo maneje los ruidos impulsivos evitando que transite al siguiente fonema, como se puede observar en la Figura 6.6.

También en este punto se construye un *modelo pequeño* para modelar pequeñas pausas. Llamaremos al fonema que modela este HMM *sp*, este modelo solo tiene 3 estados, donde existe una transición directa del estado inicial al estado final, otra característica de este modelo es que su estado intermedio es compartido con el modelo *sil*. Este modelo se llama modelo de soporte y modelan de manera opcional efectos del habla como pausas cortas, pequeños ruidos particularmente entre palabras [YEG+06], este modelo se puede observar en la Figura 6.6.

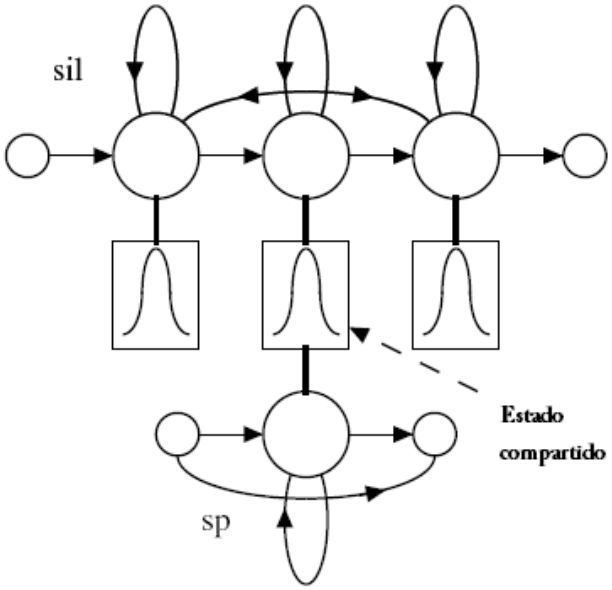


Figura 6.6: *Los modelos de silencio* sil *y* sp.

ALGORITMO-SILENCIO-PAUSA(HMM *hmm*)
▷ Extraer fonema SIL del HMM compuesto
1 $hmmSIL \leftarrow extraerHMMFonemaSil(hmm)$
▷ Agregar transiciones adicionales
2 $hmmSIL \leftarrow agregarTrans(estado2, estado4)$
3 $hmmSIL \leftarrow agregarTrans(estado4, estado2)$
▷ Crear HMM llamado sp
4 $hmmSP \leftarrow crearHMM()$
5 $hmmSP \leftarrow numeroEstados(3)$
6 $hmmSP \leftarrow agregarTrans(estado1, estado3)$
7 $hmmSP \leftarrow agregarTrans(estado2, estado2)$
▷ Compartir estado central del HMM sil con HMM sp
8 $hmmSILSP \leftarrow compartirEstado(hmmSIL_3 hmmSP_2)$
▷ agregar fonema SILSP al HMM compuesto
9 $setHMMFonemaSilSp(hmm, hmmSILSP)$
10 **return** *hmm*

6.4. Creación de Fonemas Dependientes del Contexto

Para lograr HMM's mas robustos se debe incluir información fonética que dependa del contexto, esta información dependiente del contexto es modelada con trifonemas (Secciones 4.7, 4.7.1 y 4.7.2). Esto es realizado en dos pasos: Primero crear las transcripciones para los trifonemas a partir de las trascripciones por fonema y finalmente crear los modelos HMM-trifonema a partir de los HMM por fonema.

6.4.1. Creación de los trifonemas

Primero se convierte los HMM por fonema entrenados a un conjunto de HMM dependientes del contexto llamados HMM-trifonemas, el nombre de trifonemas proviene del hecho que para un fonema dado se considera el fonema anterior y el fonema posterior según los archivos de transcripción.

Los HMM-trifonema son creados a partir de los HMM por fonema, y son entrenados usando las transcripciones mediante el algoritmo Baum-Welch 4.3. Los trifonemas obtenidos en los experimentos se encuentran en A.6.

El proceso de creación de los trifonemas sigue el siguiente algoritmo.

ALGORITMO-CREACION-TRIFONEMAS(HMM-COMPUESTO hmm, ARCHIVOTRANSCRIPCIÓN)

▷ Crear transcripciones de trifonemas a partir de las transcripciones de fonemas
1 $P \leftarrow read(archivoTranscripcin)$
2 $N \leftarrow length(hmm)$
3 **for** $i \leftarrow 1$ **to** N
4 **do** ▷ Para todos los fonemas, crear trifonemas dependientes del contexto según el archivo de transcripción
5 $hmmTrifonema \leftarrow crearTrifonema(fonema_{izquierdo}(P_i), fonema_{central}(P_i), fonema_{derecho}(P_i))$
6 $hmmTrifonema \leftarrow getMatrizTransicion(fonema_{central}(P_i))$

7 ▷ Reestimar los parámetros del HMM del trifonema usando Baum Welch a partir de las transcripciones de los trifonemas
8 $Baum\text{-}Welch(hmmTrifonema)$
9 **return** $hmmTrifonema$

La descripción del algoritmo anterior es la siguiente: Para cada fonema presente, se crea un trifonema según el contexto impuesto por el archivo de transcripción, luego se crea un HMM para este trifonema, el cual tiene la matriz de transición del fonema central. Finalmente se estiman los parámetros del HMM con el algoritmo Baum-Welch.

6.4.2. Clustering de estados

Los HMM trifonemas creados con el algoritmo anterior comparten la misma matriz de transición, por la insuficiencia de los datos de entrenamiento se deben realizar un procedimiento de clustering a los estados de todos los HMM para que los estados que se encuentran en el mismo cluster compartan los mismos parámetros. Para esto se sigue el siguiente algoritmo de clustering

CLUSTERING-ESTADOS(N,$hmmTrifonema$)

1 Crear un cluster por estado para todos los estados de $hmmTrifonema$
2 $n \leftarrow$ numero de clusters
3 **while** $n > N$
4 **do**
5 Encontrar i y j para el cual $g(i, j)$ es mínimo
6 Mezclar los clusters i y j
7 $n \leftarrow n - 1$
8 Para cada cluster los estados deben compartir los mismos parámetros
9 **return** $hmmTrifonema$ actualizado

Donde $g(i, j)$ es la distancia entre los clusters i y j definida como la máxima distancia entre cualquier estado del cluster i y cualquier estado del cluster j. La distancia es una distancia euclideana entre las medias de los GMM. Finalmente todos los estados en cada cluster comparte los parámetros, los cuales son estimados nuevamente con el algoritmo Baum-Welch visto en la Sección 4.3.

6.5. Decodificación

En esta paso del algoritmo se aplica el ALGORITMO-TOKEN-PASSING-CFG descrito en la Sección 4.8. El proceso de decodificación requiere el grafo de palabras generado por la gramática, el

diccionario de pronunciación y el conjunto de HMM obtenidos en el paso anterior. El grafo de palabras cada nodo es una red de HMM conectados. Este grafo de palabras es un HMM de gran tamaño.

Para una observación O desconocida que contiene T frames (dada por sus correspondientes coeficientes MFCC), cada camino que empieza en el nodo inicial del grafo y termina en el nodo final y que pasa por exactamente por T estados del HMM, es una potencial hipótesis de reconocimiento.

El problema de decodificación es encontrar estos caminos a través del grafo de palabras que tenga asociada la probabilidad mas alta. Estos caminos son encontrados usando el algoritmo *Token Passing* [YRT89]. Un *token* representa el camino parcial a través del grafo de reconocimiento desde el tiempo 0 hasta el tiempo t. En el tiempo 0 el token es ubicado en cada posible estado inicial.

En cada paso los tokens son propagados, si hay muchas salidas de un nodo del grafo los tokens son copiados a todas las posibles salidas y son explorados de manera paralela. Cada vez que los tokens pasan por las transiciones y por los nodos la probabilidad asignada al token se va incrementando.

El grafo de palabras puede almacenar a lo más N tokens. Al final de cada iteración del algoritmo solo N tokens permanecen. Se debe mantener la historia de recorrido de cada token que será asociada a la palabra reconocida.

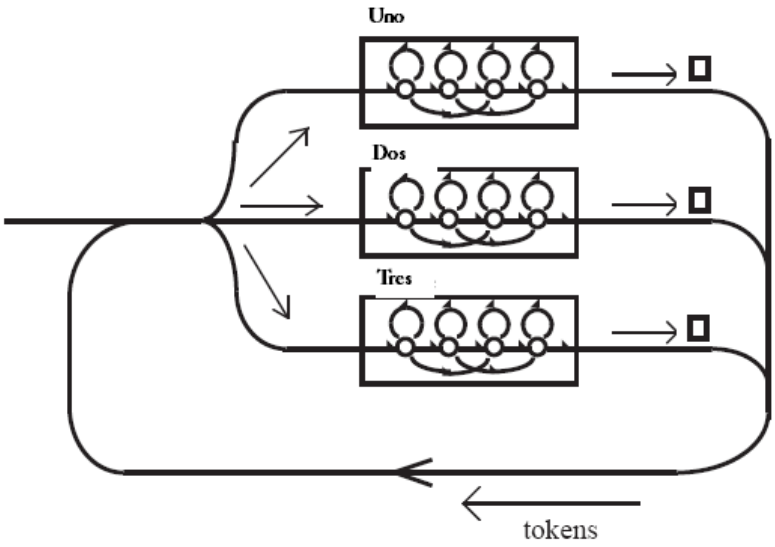


Figura 6.7: *Esquema de decodificación usando el algoritmo token passing.*

6.6. Algoritmo Propuesto para RITH

Luego de la descripción de los diversos pasos del algoritmo, describimos este de manera mas formal.

ALGORITMO-RECUPERACIÓN-INFORMACIÓN-TEXTOS-HABLADOS(G, D, N)

Entrada Gramática GLC G, Diccionario de pronunciación D, Número de Sentencias N, Periodo de muestreo T, tamaño del frame $\|frame\|$

Preparación de los datos

▷ Construcción del grafo de palabras
$Grafo \leftarrow$ ALGORITMO-PROCESAMIENTO-GRAMÁTICA (G)
▷ Construcción del diccionario de pronunciación
$Dic \leftarrow$ ALGORITMO-CONSTRUCCIÓN-DICCIONARIO-PRONUNCIACIÓN (D, G)
▷ Generación aleatoria y grabación de sentencias
for $i \leftarrow 1$ **to** N
    do
        $S_i \leftarrow$ ALGORITMO-GENERACIÓN-ALEATORIA-SENTENCIAS (G)
        $xa_i \leftarrow$ GRABAR-SENTENCIA(S_i)

▷ Construcción de los archivos de transcripción
for $i \leftarrow 1$ **to** N
    do
        archivoTranscripción $\leftarrow$ ALGORITMO-CONSTRUCCIÓN-ARCHIVOS-TRANSCRIPCIÓN (S_i,Dic)

▷ Extracción de características usando MFCC
for $i \leftarrow 1$ **to** N
    do
        $mfcc_i \leftarrow$ ALGORITMO-MFCC($xa_i, T, \|frame\|$)

Construcción de los HMM's

▷ Definición de la estructura de los HMM's
for $i \leftarrow 1$ **to** *length*(*extraerFonemas*(*Dic*))
    do ▷ Por cada fonema (incluido el fonema 'sil' para el silencio) del diccionario crear HMM
        $hmm_i \leftarrow$ *crearHMM*(*HMM-Prototipo*)
        $hmm_i \leftarrow$ *numeroEstados*(*5*) ▷ Definir un HMM con 5 estados
        for $e \leftarrow 2$ **to** 4
            do ▷ Para los 3 estados intermedios
                ▷ Asignar un GMM con un arreglo de medias y un arreglo de matrices de covarianza
                $hmm_{i,e} \leftarrow$ *establecerGMM*($\mu[], \Sigma[]$)

        $hmm_i \leftarrow$ *matrizTransiciones*(A) ▷ Definir la matriz de transiciones del HMM

▷ Estimación de los parámetros iniciales de los HMM´s
for $i \leftarrow 2$ **to** *length*(*extraerFonemas*(*Dic*))
    do ▷ Creación de un solo HMM compuesto
        $hmm_i \leftarrow$ *concatenar*(hmm_{i-1}, hmm_i)

$hmm \leftarrow$ ALGORITMO-ESTIMACIÓN-PARÁMETROS-INICIALES-HMM (hmm, $mfcc$)

▷ Entrenamiento de los HMM
$hmm \leftarrow$ ALGORITMO-ENTRENAMIENTO-HMM'S(hmm, $mfcc$, ARCHIVOTRANSCRIPCIÓN)
▷ Establecer HMM para silencio y pausa
$hmm \leftarrow$ ALGORITMO-SILENCIO-PAUSA(hmm)

Creación de los fonemas dependientes del contexto

▷ Creación de los trifonemas

27 $hmmTrifonema \leftarrow$ ALGORITMO-CREACION-TRIFONEMAS(hmm, ARCHIVOTRANSCRIPCIÓN)

▷ Clustering de estados

28 $hmmTrifonema \leftarrow$ CLUSTERING-ESTADOS(N,$hmmTrifonema$)

Decodificación

29 **for** $i \leftarrow 1$ **to** numeroArchivosRecuperar

30 $R_i \leftarrow$ ALGORITMO-TOKEN-PASSING-CFG($Grafo$,Dic,G,$mfcc_i$,$hmmTrifonema$)

31 **return** R

El algoritmo descrito anteriormente tiene dos etapas principales: La etapa de *Entrenamiento* comprendida por : Preparación de los datos (Lineas 1-12), Construcción de los HMM's (Lineas 12-26), Creación de los fonemas dependientes del contexto (Lineas 27-28) y La etapa de *Recuperación de información* (Lineas 29-31). En etapa de entrenamiento se definen diversos parámetros y se establece el lenguaje a procesar, en una aplicación verdadera, la etapa de entrenamiento se realiza una vez (o muy pocas veces). Los parámetros calculados son guardados. La etapa de reconocimiento sin embargo es ejecutada cada vez que se requiere recuperar información de algún archivo, usando los parámetros calculados en la etapa de entrenamiento.

En las Figuras 6.8, 6.9 y 6.10 mostramos el diagrama de flujo del algoritmo.

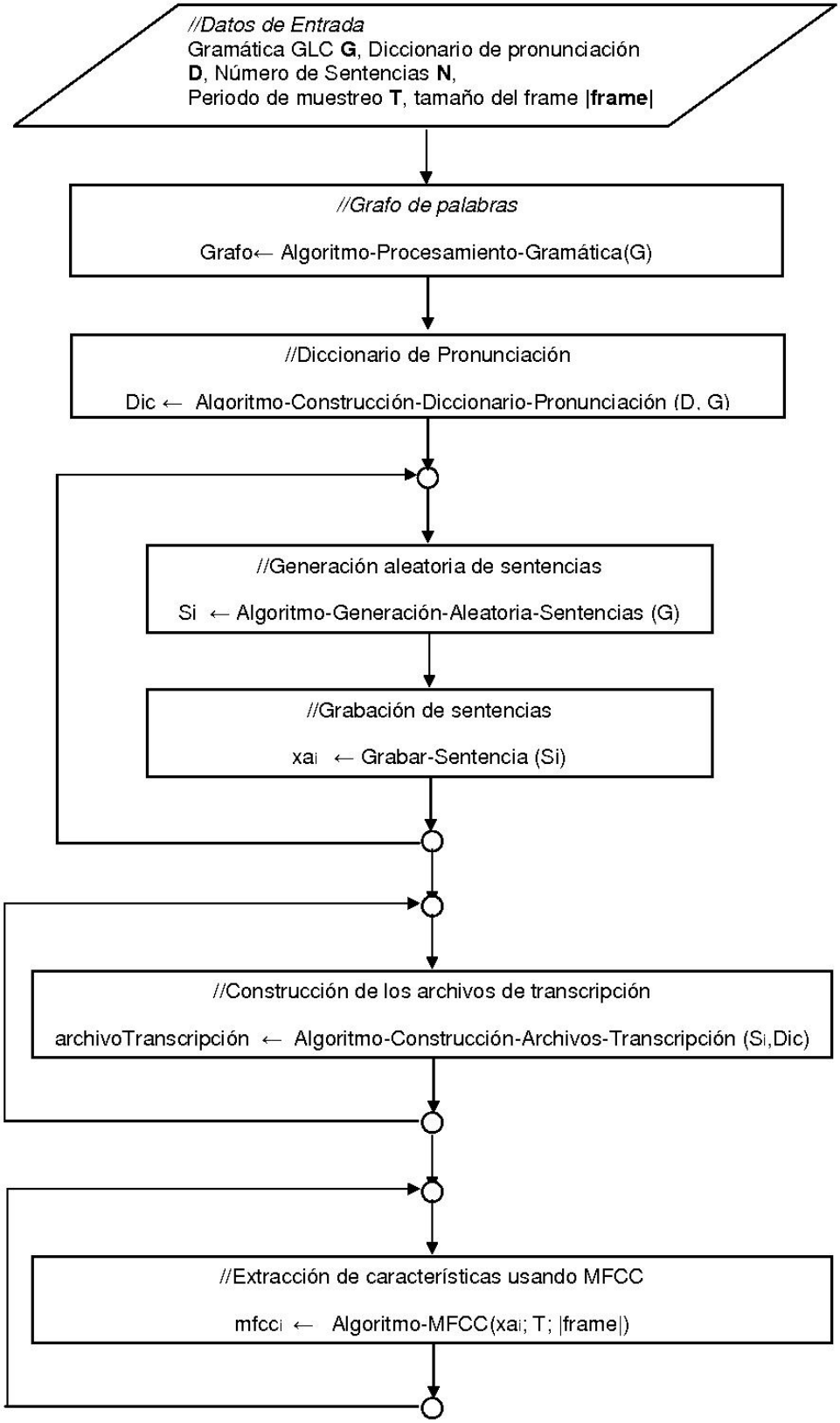


Figura 6.8: *Diagrama de Flujo de Datos.*

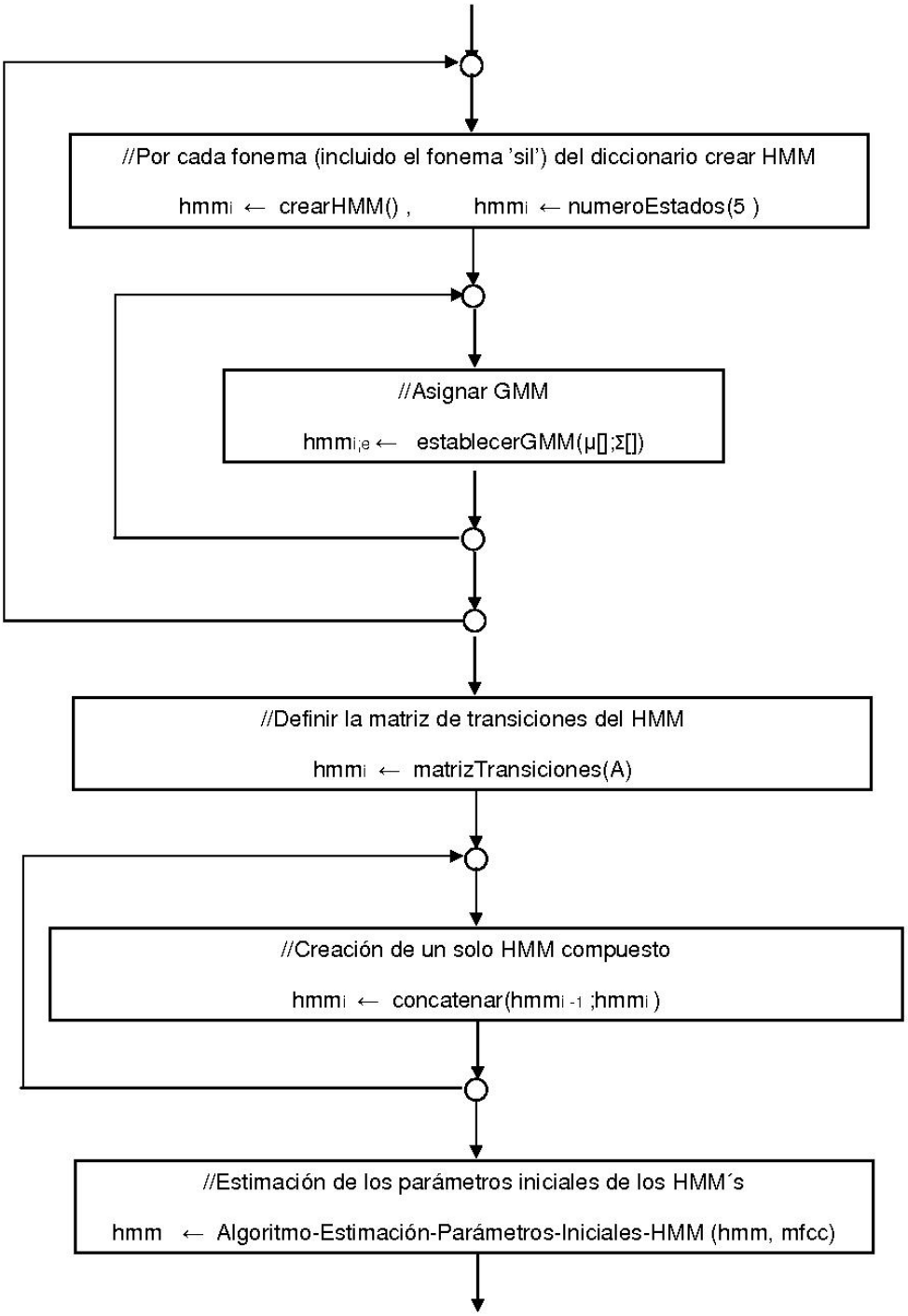


Figura 6.9: *Diagrama de Flujo de Datos (Continuación).*

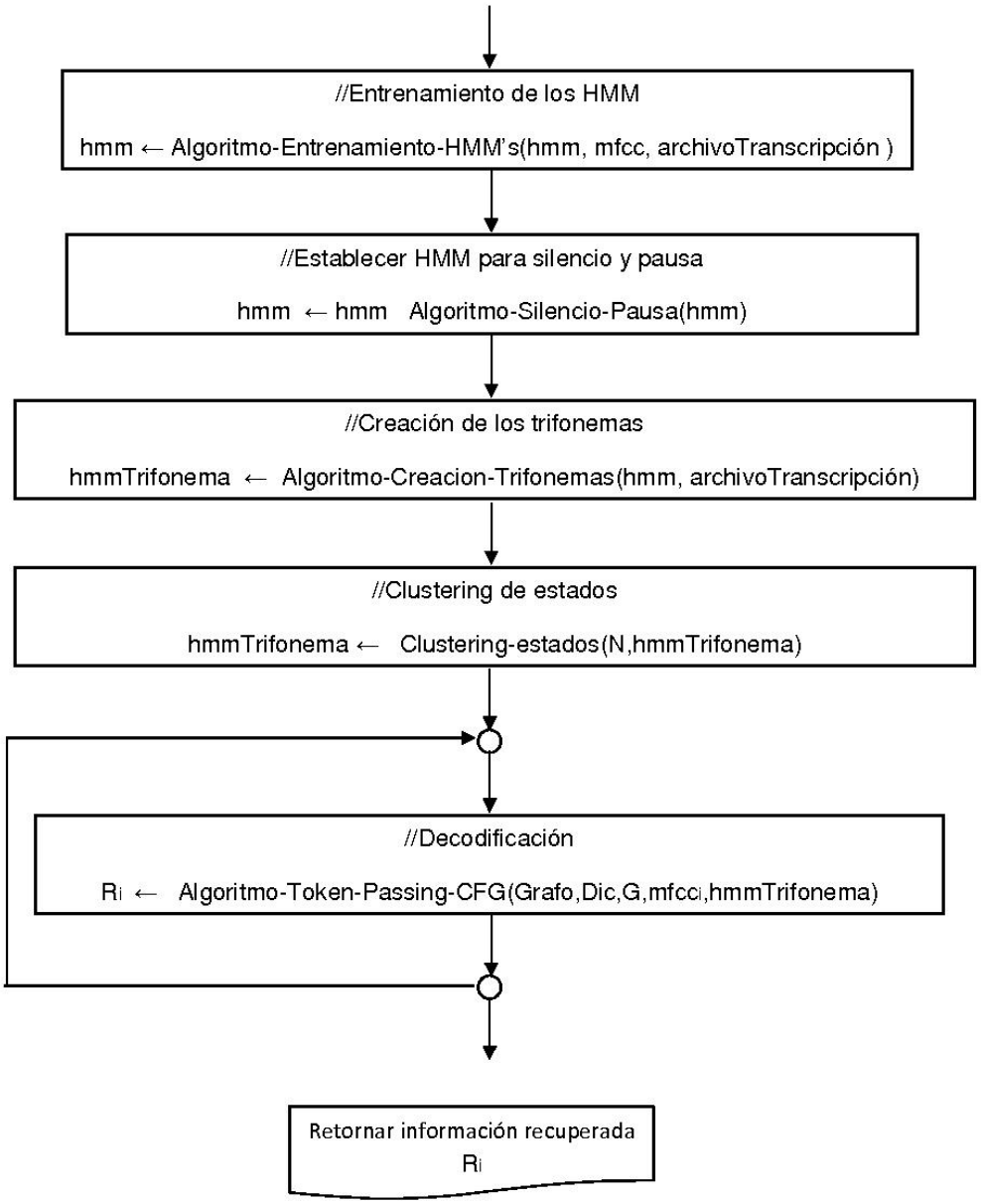


Figura 6.10: *Diagrama de Flujo de Datos (Continuación).*

Parte III

Resultados

Capítulo 7

Resultados

En el presente capítulo mostramos los resultados obtenidos a través de una serie de experimentos. Primeramente mostramos los resultados obtenidos teniendo en cuenta el números de palabras correctas recuperadas de todos los archivos de audio, Luego los resultados obtenidos de manera general variando los parámetros de *penalidad por inserción de palabra* y el *factor escala de la gramática* descritos en la Sección 4.8.2, la validación del clasificador usando la técnica de validación cruzada k-fold [DHS01], la prueba de significancia estadística t-student y el análisis del tiempo de ejecución del algoritmo.

La información usada en los experimentos se encuentran descritas de manera detallada en el Apéndice (Sección A), el cual contiene la información de la transcripción de los archivos de audio utilizados (Sección A.4), el diccionario con la pronunciación fonética de cada palabra en estándar IPA (SecciónA.2.1), la gramática utilizada para modelar el lenguaje (SecciónA.7), la lista de fonemas obtenidos de todas las palabras presentes del conjunto de entrenamiento (SecciónA.2.2), las sentencias generadas aleatoriamente que sirvieron para realizar el entrenamiento, test y validación del clasificador (Sección A.3), el grafo de dependencias de palabras obtenido a partir de la gramática (Sección A.1.1), la transcripción fonética de los archivos de transcripción de las grabaciones (Sección A.4.1), los trifonemas (Sección 1.1) y finalmente los modelos ocultos de Markov de cada fonema (Sección A.5).

7.1. Resultados por Sentencia de Palabras

Los siguientes resultados muestran los resultados de la recuperación de la información por sentencia vs la transcripción de cada archivo de test. Todas las sentencias fueron generadas aleatoriamente y se encuentran descritas en A.3. Las sentencias están numeradas del 1 al 200. Se usó la siguiente notación: H es el número de palabras correctas por sentencia, E es el número de eliminaciones, S es el número de sustituciones, I es el número de inserciones y N es el número total de palabras en la transcripción de los archivos de test, según la tasa de error por palabras WER, (Sección 4.4).

El porcentaje de palabras correctamente reconocidas (Correctas) [YEG+06] y la la precisión [YEG+06] son definidas como:

$$\text{Correctas} = \frac{H}{N} \times 100 \tag{7.1}$$

$$\text{Precisión} = \frac{H - I}{N} \times 100 \tag{7.2}$$

La relación entre Correctas y Precisión es determinada por el número de palabras insertadas, puede verificarse a partir de las fórmulas que el rango aceptable de valores de es de 0 a 100 para el valor de *Correctas* y de -100 a 100 para el valor de *Precisión*.

Sentencia	Correctas	Precisión	H	E	S	I	N
1	100.00	90.48	21	0	0	2	21
2	100.00	33.33	3	0	0	2	3
3	100.00	84.62	13	0	0	2	13
4	100.00	0.00	2	0	0	2	2
5	100.00	0.00	2	0	0	2	2
6	100.00	0.00	2	0	0	2	2
7	100.00	33.33	3	0	0	2	3
8	100.00	80.00	10	0	0	2	10
9	100.00	0.00	2	0	0	2	2
10	100.00	0.00	2	0	0	2	2
11	100.00	0.00	2	0	0	2	2
12	100.00	33.33	3	0	0	2	3
13	100.00	33.33	3	0	0	2	3
14	100.00	84.21	19	0	0	3	19
15	100.00	33.33	3	0	0	2	3
16	100.00	91.67	24	0	0	2	24
17	100.00	0.00	2	0	0	2	2
18	100.00	33.33	3	0	0	2	3
19	100.00	87.50	16	0	0	2	16
20	100.00	0.00	2	0	0	2	2
21	100.00	33.33	3	0	0	2	3
22	100.00	33.33	3	0	0	2	3
23	100.00	33.33	3	0	0	2	3
24	100.00	33.33	3	0	0	2	3
25	100.00	33.33	3	0	0	2	3
26	100.00	81.82	11	0	0	2	11
27	100.00	71.43	7	0	0	2	7
28	100.00	83.33	12	0	0	2	12
29	50.00	-50.00	1	0	1	2	2
30	100.00	0.00	2	0	0	2	2
31	100.00	0.00	2	0	0	2	2
32	100.00	0.00	2	0	0	2	2
33	100.00	33.33	3	0	0	2	3
34	100.00	0.00	2	0	0	2	2
35	100.00	0.00	2	0	0	2	2
36	100.00	0.00	2	0	0	2	2
37	100.00	33.33	3	0	0	2	3
38	100.00	33.33	3	0	0	2	3
39	100.00	33.33	3	0	0	2	3
40	100.00	33.33	3	0	0	2	3
41	100.00	0.00	2	0	0	2	2
42	100.00	0.00	2	0	0	2	2
43	100.00	33.33	3	0	0	2	3
44	100.00	33.33	3	0	0	2	3
45	100.00	33.33	3	0	0	2	3
46	100.00	0.00	2	0	0	2	2
47	100.00	33.33	3	0	0	2	3
48	100.00	0.00	2	0	0	2	2
49	100.00	33.33	3	0	0	2	3
50	100.00	0.00	2	0	0	2	2

Tabla 7.1: *Resultados por sentencias, resultados de las primeras 50 sentencias.*

Sentencia	Correctas	Precisión	H	E	S	I	N
51	100.00	60.00	5	0	0	2	5
52	100.00	0.00	2	0	0	2	2
53	100.00	50.00	4	0	0	2	4
54	100.00	0.00	2	0	0	2	2
55	100.00	0.00	2	0	0	2	2
56	100.00	33.33	3	0	0	2	3
57	100.00	0.00	2	0	0	2	2
58	100.00	94.74	38	0	0	2	38
59	100.00	0.00	2	0	0	2	2
60	100.00	0.00	2	0	0	2	2
61	100.00	33.33	3	0	0	2	3
62	100.00	0.00	2	0	0	2	2
63	100.00	0.00	2	0	0	2	2
64	100.00	60.00	5	0	0	2	5
65	100.00	0.00	2	0	0	2	2
66	100.00	92.31	26	0	0	2	26
67	100.00	85.71	14	0	0	2	14
68	100.00	0.00	2	0	0	2	2
69	100.00	60.00	5	0	0	2	5
70	100.00	0.00	2	0	0	2	2
71	100.00	92.00	25	0	0	2	25
72	100.00	71.43	7	0	0	2	7
73	100.00	33.33	3	0	0	2	3
74	100.00	71.43	7	0	0	2	7
75	100.00	77.78	9	0	0	2	9
76	100.00	33.33	3	0	0	2	3
77	100.00	60.00	5	0	0	2	5
78	100.00	71.43	7	0	0	2	7
79	100.00	33.33	3	0	0	2	3
80	100.00	0.00	2	0	0	2	2
81	100.00	0.00	2	0	0	2	2
82	100.00	60.00	5	0	0	2	5
83	100.00	0.00	2	0	0	2	2
84	100.00	88.24	17	0	0	2	17
85	100.00	50.00	4	0	0	2	4
86	100.00	87.50	16	0	0	2	16
87	100.00	0.00	2	0	0	2	2
88	100.00	33.33	3	0	0	2	3
89	100.00	33.33	3	0	0	2	3
90	100.00	0.00	2	0	0	2	2
91	100.00	33.33	3	0	0	2	3
92	100.00	71.43	7	0	0	2	7
93	100.00	50.00	4	0	0	2	4
94	100.00	33.33	3	0	0	2	3
95	100.00	33.33	3	0	0	2	3
96	100.00	0.00	2	0	0	2	2
97	100.00	33.33	3	0	0	2	3
98	100.00	96.23	53	0	0	2	53
99	100.00	33.33	3	0	0	2	3
100	100.00	0.00	2	0	0	2	2

Tabla 7.2: *Resultados por sentencias, resultados de la sentencia 51 a la sentencia 100.*

Sentencia	Correctas	Precisión	H	E	S	I	N
101	100.00	0.00	2	0	0	2	2
102	100.00	0.00	2	0	0	2	2
103	100.00	0.00	2	0	0	2	2
104	100.00	0.00	2	0	0	2	2
105	100.00	33.33	3	0	0	2	3
106	100.00	85.71	14	0	0	2	14
107	100.00	33.33	3	0	0	2	3
108	100.00	33.33	3	0	0	2	3
109	100.00	92.59	27	0	0	2	27
110	100.00	83.33	12	0	0	2	12
111	100.00	80.00	10	0	0	2	10
112	100.00	33.33	3	0	0	2	3
113	100.00	66.67	6	0	0	2	6
114	100.00	33.33	3	0	0	2	3
115	100.00	0.00	2	0	0	2	2
116	100.00	33.33	3	0	0	2	3
117	100.00	33.33	3	0	0	2	3
118	100.00	66.67	6	0	0	2	6
119	100.00	33.33	3	0	0	2	3
120	100.00	75.00	8	0	0	2	8
121	100.00	0.00	2	0	0	2	2
122	100.00	0.00	2	0	0	2	2
123	100.00	0.00	2	0	0	2	2
124	100.00	0.00	2	0	0	2	2
125	100.00	0.00	2	0	0	2	2
126	100.00	80.00	10	0	0	2	10
127	100.00	33.33	3	0	0	2	3
128	100.00	33.33	3	0	0	2	3
129	100.00	77.78	9	0	0	2	9
130	100.00	33.33	3	0	0	2	3
131	100.00	33.33	3	0	0	2	3
132	100.00	33.33	3	0	0	2	3
133	100.00	33.33	3	0	0	2	3
134	100.00	0.00	2	0	0	2	2
135	100.00	33.33	3	0	0	2	3
136	100.00	33.33	3	0	0	2	3
137	100.00	90.91	22	0	0	2	22
138	100.00	0.00	2	0	0	2	2
139	100.00	60.00	5	0	0	2	5
140	100.00	33.33	3	0	0	2	3
141	100.00	93.33	30	0	0	2	30
142	100.00	33.33	3	0	0	2	3
143	100.00	0.00	2	0	0	2	2
144	100.00	0.00	2	0	0	2	2
145	100.00	0.00	2	0	0	2	2
146	100.00	0.00	2	0	0	2	2
147	100.00	0.00	2	0	0	2	2
148	100.00	33.33	3	0	0	2	3
149	66.67	0.00	2	0	1	2	3
150	100.00	33.33	3	0	0	2	3

Tabla 7.3: *Resultados por sentencias, resultados de la sentencia 101 a la sentencia 150.*

Sentencia	Correctas	Precisión	H	E	S	I	N
151	100.00	0.00	2	0	0	2	2
152	100.00	33.33	3	0	0	2	3
153	100.00	95.24	42	0	0	2	42
154	100.00	0.00	2	0	0	2	2
155	100.00	0.00	2	0	0	2	2
156	100.00	33.33	3	0	0	2	3
157	100.00	81.82	11	0	0	2	11
158	100.00	33.33	3	0	0	2	3
159	100.00	92.00	25	0	0	2	25
160	100.00	33.33	3	0	0	2	3
161	100.00	75.00	8	0	0	2	8
162	100.00	33.33	3	0	0	2	3
163	100.00	84.62	13	0	0	2	13
164	100.00	77.78	9	0	0	2	9
165	100.00	33.33	3	0	0	2	3
166	100.00	0.00	2	0	0	2	2
167	100.00	33.33	3	0	0	2	3
168	100.00	71.43	7	0	0	2	7
169	100.00	0.00	2	0	0	2	2
170	100.00	0.00	2	0	0	2	2
171	100.00	84.62	13	0	0	2	13
172	100.00	0.00	2	0	0	2	2
173	100.00	33.33	3	0	0	2	3
174	100.00	0.00	2	0	0	2	2
175	100.00	88.89	18	0	0	2	18
176	100.00	92.00	25	0	0	2	25
177	100.00	33.33	3	0	0	2	3
178	100.00	81.82	11	0	0	2	11
179	100.00	33.33	3	0	0	2	3
180	100.00	71.43	7	0	0	2	7
181	100.00	0.00	2	0	0	2	2
182	100.00	33.33	3	0	0	2	3
183	100.00	33.33	3	0	0	2	3
184	100.00	0.00	2	0	0	2	2
185	100.00	50.00	4	0	0	2	4
186	100.00	0.00	2	0	0	2	2
187	100.00	33.33	3	0	0	2	3
188	100.00	86.67	15	0	0	2	15
189	100.00	33.33	3	0	0	2	3
190	100.00	33.33	3	0	0	2	3
191	100.00	0.00	2	0	0	2	2
192	100.00	0.00	2	0	0	2	2
193	100.00	50.00	4	0	0	2	4
194	100.00	33.33	3	0	0	2	3
195	100.00	0.00	2	0	0	2	2
196	100.00	0.00	2	0	0	2	2
197	100.00	33.33	3	0	0	2	3
198	100.00	90.91	22	0	0	2	22
199	100.00	0.00	2	0	0	2	2
200	100.00	66.67	6	0	0	2	6

Tabla 7.4: *Resultados por sentencias, resultados de la sentencia 151 a la sentencia 200.*

p	q	Correctas	Precisión	H	E	S	I	N
0	5	99.83	64.96	1148	0	2	401	1150
0	10	99.83	64.96	1148	0	2	401	1500
0	20	99.83	64.96	1148	0	2	401	1150
0	40	99.83	64.96	1148	0	2	401	1150
0	75	99.83	64.96	1148	0	2	401	1150
0	100	99.83	64.96	1148	0	2	401	1150
5	0	99.83	64.96	1148	0	2	401	1150
10	0	99.83	64.96	1148	0	2	401	1150
20	0	99.83	64.96	1148	0	2	401	1150
40	0	99.83	64.96	1148	0	2	401	1150
75	0	99.83	64.43	1148	0	2	407	1150
100	0	99.83	63.65	1148	0	2	416	1150
5	95	99.83	64.96	1148	0	2	401	1150
20	80	99.83	64.96	1148	0	2	401	1150
30	70	99.83	64.96	1148	0	2	401	1150
45	55	99.83	64.96	1148	0	2	401	1150
50	50	99.83	64.87	1148	0	2	402	1150

Tabla 7.5: *Resultados para experimentos variando los valores:* penalidad por inserción de palabra *y el* factor escala de la gramática.

7.2. Resultados Generales variando parámetros: Penalidad por inserción y factor de escala

Los resultados mostrados en la Tabla 7.5 describen los resultados generales variando dos parámetros: *penalidad por inserción de palabra* y el *factor escala de la gramática* denotados por las etiquetas p y q de manera respectiva, descritas en la Sección 4.8.2. La penalidad por inserción de palabra es un valor constante añadido a cada token cuando este va del final de una palabra hasta el inicio de otra. El factor escala de la gramática, es un valor por el cual la probabilidad del modelo del lenguaje es escalado antes de ser agregado a cada token que va del final de una palabra al inicio de otra palabra.

7.3. Evaluación del Algoritmo usando Validación Cruzada k-fold

La validación cruzada k-fold [DHS01], es una técnica ampliamente aceptada y usada para validar el clasificador construido de manera experimental.

La validación cruzada parte el conjunto de datos en k subconjuntos, uno de ellos es usado a manera de test, mientras los demás conjuntos son usados para entrenar el clasificador, luego de manera iterativa se escoge otro conjunto de test, y los siguientes conjuntos sobrantes son usados para entrenar el clasificador, esto sucede de manera sucesiva k veces. Finalmente se obtiene el error promedio del clasificador.

El valor de k es establecido usualmente a 10. Es decir los datos fueron divididos en grupos de 20, siendo 20 datos los datos de test, y el resto los datos de entrenamiento, iterando el algoritmo 20 veces en total.

iter	1	2	3	4	5	6	7	8	9	10
EI	0.68	0.17	0.10	3.00	6.65	0.18	0.20	2.65	1.55	1.57
EP	0.68	0.43	0.32	0.99	2.12	1.80	1.57	1.70	1.69	1.68
iter	11	12	13	14	15	16	17	18	19	20
EI	0.17	0.17	0.17	0.13	0.15	1.00	4.53	0.16	0.17	0.17
EP	1.54	1.42	1.33	1.24	1.17	1.16	1.36	1.29	1.23	1.18

Tabla 7.6: *Error del clasificador obtenido usando validación cruzada k-fold*

La validación cruzada k-fold es usada para determinar la precisión de un algoritmo de aprendizaje para predecir datos con los cuales no fueron entrenados. El algoritmo k-fold se detalla a continuación.

ALGORITMO VALIDACIÓN CRUZADA K-FOLD $(k, datos)$

▷ Conjunto de datos de entrenamiento
1 $T \leftarrow \phi$
▷ Error por iteración
2 $EI \leftarrow 0$
▷ Error promedio
3 $EP \leftarrow 0$
4 **for** $i \leftarrow 1$ **to** k
▷ Conjunto T contiene todos los datos de entrenamiento menos el grupo k
5 $T \leftarrow datos \setminus datos^k$
6 Entrenar el algoritmo usando T
7 Testar el algoritmo usando $datos^k$
▷ Registrar el número de errores.
8 $EI \leftarrow \%errores.$
9 $EP \leftarrow EP + EI.$
10
11 **return** $\frac{EP}{k}$

los valores EI e EP son los errores por iteración y los errores promedio por iteración para la validación cruzada k-fold.

7.4. Prueba de Significancia Estadística de t-student

Para contrastar la hipótesis planteada en la Sección 5.2 fue usada la prueba de t-student con un nivel de significancia de $\alpha = 0{,}05$ pues no se conoce la media ni la varianza de la población pero se tiene información del error medio y la varianza obtenido por el algoritmo propuesto en las muestras.

Bajo el esquema de que una hipótesis estadística es una afirmación verdadera o falsa sobre alguna característica desconocida de la población y definiendo el parámetro μ como como el nivel de error de recuperación de información, podemos definir la hipótesis nula H_0 de la siguiente manera:

$$H_0 : \mu \quad \leq \quad 0{,}05 \tag{7.3}$$

es decir la hipótesis es verdadera si el error promedio es menor o igual a 0.05. La hipótesis alternativa H_1 es definida como

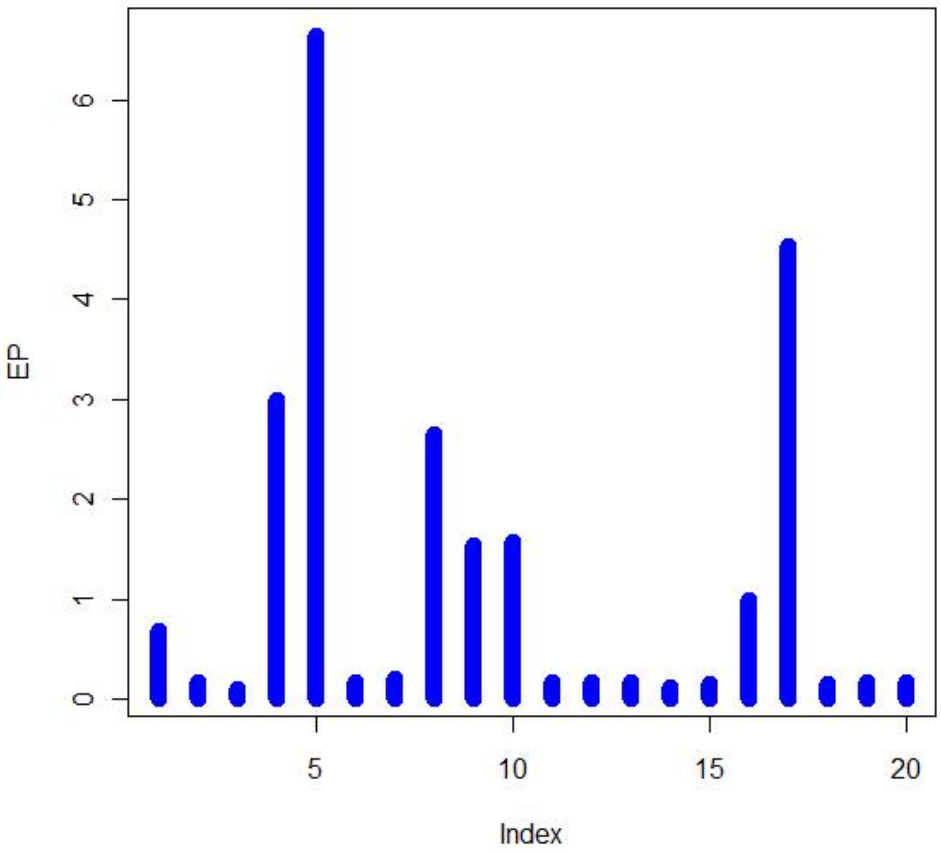


Figura 7.1: *Error por iteración del k-fold.*

$$H_1 : \mu \quad > \quad 0{,}05 \tag{7.4}$$

El estadístico de prueba es una función sobre las 200 muestras y del valor especificado por la hipótesis nula, definido como el error promedio obtenido. Como el valor promedio de correctas es de 99.38 %, entonces el error promedio es 100 %-99.83 %=0.17 %. , Las ecuaciones anteriores muestran que el test de hipótesis es un test de contraste unilateral o de una cola. Luego el valor del estadístico de prueba t de la técnica de t-student es calculado de la siguiente manera:

$$\begin{aligned} t &= \frac{(\bar{x} - \mu)}{\frac{S}{\sqrt{n}}} \\ &= \frac{(0{,}0017 - 0{,}05)}{\frac{0{,}17}{\sqrt{200}}} \\ &= -0{,}0200901515 \end{aligned}$$

donde S es la varianza del error de las muestras, y n es el número de muestras. Como el contraste es unilateral, buscamos en las tablas de la t de Student, con 199 grados de libertad, el valor que

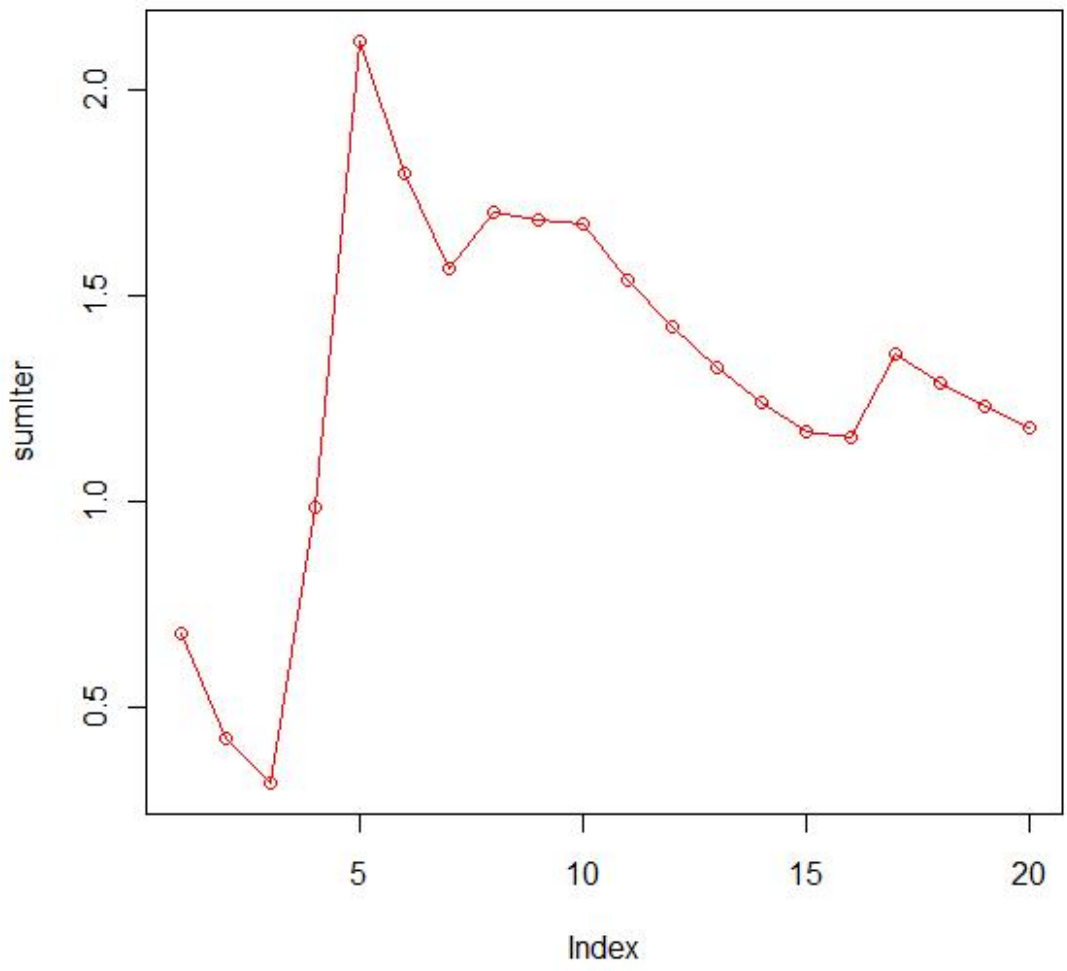


Figura 7.2: *Error promedio por iteración del k-fold.*

deja por debajo de sí una probabilidad de $1 - \alpha = 0,95$, que resulta ser 0.254.

La región de aceptación es definida para contraste de hipótesis unilateral a partir del valor del nivel de confianza $1 - \alpha$. Luego la región de aceptación esta definida mediante el intervalo

$$\begin{gathered}(-\infty, t_\alpha) \\ (-\infty, 0{,}254)\end{gathered} \tag{7.5}$$

Luego los valores de aceptación y rechazo para H_0 son los siguientes:

- Aceptar H_0 si $t \in (-\infty, 0{,}254)$
- Rechazar H_0 si $t \notin (-\infty, 0{,}254)$

Finalmente como $\bar{x} = -0{,}0200901515$ se encuentra en la región de aceptación tenemos que se acepta la hipótesis nula H_0

7.5. Tiempo de Ejecución del Algoritmo

Fue realizado un análisis del tiempo de ejecución del algoritmo teniendo en cuenta la etapa de entrenamiento comprendida por : Preparación de los datos (Líneas 1-12), Construcción de los HMM's (Líneas 12-26), Creación de los fonemas dependientes del contexto (Líneas 27-28) y La etapa de *Recuperación de información* (Líneas 29-31)

Repetimos el algoritmo para el análisis

ALGORITMO-RECUPERACIÓN-INFORMACIÓN-TEXTOS-HABLADOS(G, D, N)

Entrada Gramática GLC G, Diccionario de pronunciación D, Número de Sentencias N, Periodo de muestreo T, tamaño del frame $\|frame\|$

Preparación de los datos

▷ Construcción del grafo de palabras

1 $Grafo \leftarrow$ ALGORITMO-PROCESAMIENTO-GRAMÁTICA (G)

▷ Construcción del diccionario de pronunciación

2 $Dic \leftarrow$ ALGORITMO-CONSTRUCCIÓN-DICCIONARIO-PRONUNCIACIÓN (D, G)

▷ Generación aleatoria y grabación de sentencias

3 **for** $i \leftarrow 1$ **to** N
4 **do**
5 $S_i \leftarrow$ ALGORITMO-GENERACIÓN-ALEATORIA-SENTENCIAS (G)
6 $xa_i \leftarrow$ GRABAR-SENTENCIA(S_i)

▷ Construcción de los archivos de transcripción

7 **for** $i \leftarrow 1$ **to** N
8 **do**
9 archivoTranscripción $\leftarrow$ ALGORITMO-CONSTRUCCIÓN-ARCHIVOS-TRANSCRIPCIÓN (S_i,Dic)

▷ Extracción de características usando MFCC

10 **for** $i \leftarrow 1$ **to** N
11 **do**
12 $mfcc_i \leftarrow$ ALGORITMO-MFCC($xa_i, T, \|frame\|$)

Construcción de los HMM's

▷ Definición de la estructura de los HMM's

13 **for** $i \leftarrow 1$ **to** $length(extraerFonemas(Dic))$
14 **do** ▷ Por cada fonema (incluido el fonema 'sil' para el silencio) del diccionario crear HMM
15 $hmm_i \leftarrow crearHMM(HMM\text{-}Prototipo)$
16 $hmm_i \leftarrow numeroEstados(5)$ ▷ Definir un HMM con 5 estados
17 **for** $e \leftarrow 2$ **to** 4
18 **do** ▷ Para los 3 estados intermedios
▷ Asignar un GMM con un arreglo de medias y un arreglo de matrices de covarianza
19 $hmm_{i,e} \leftarrow establecerGMM(\mu[], \Sigma[])$

20 $hmm_i \leftarrow matrizTransiciones(A)$ ▷ Definir la matriz de transiciones del HMM

▷ Estimación de los parámetros iniciales de los HMM´s
21 **for** $i \leftarrow 2$ **to** $length(extraerFonemas(Dic))$
22 **do** ▷ Creación de un solo HMM compuesto
23 $hmm_i \leftarrow concatenar(hmm_{i\text{-}1}, hmm_i)$

24 $hmm \leftarrow$ Algoritmo-Estimación-Parámetros-Iniciales-HMM $(hmm, mfcc)$

▷ Entrenamiento de los HMM
25 $hmm \leftarrow$ Algoritmo-Entrenamiento-HMM's(hmm, $mfcc$, archivoTranscripción)
▷ Establecer HMM para silencio y pausa
26 $hmm \leftarrow$ Algoritmo-Silencio-Pausa(hmm)
Creación de los fonemas dependientes del contexto
▷ Creación de los trifonemas
27 $hmmTrifonema \leftarrow$ Algoritmo-Creacion-Trifonemas(hmm, archivoTranscripción)

▷ Clustering de estados
28 $hmmTrifonema \leftarrow$ Clustering-estados(N,$hmmTrifonema$)

Decodificación
29 **for** $i \leftarrow 1$ **to** numeroArchivosRecuperar
30 $R_i \leftarrow$ Algoritmo-Token-Passing-CFG($Grafo$,Dic,G,$mfcc_i$,$hmmTrifonema$)
31 **return** R

7.5.1. Análisis Etapa Entrenamiento

La línea 1 hace referencia al Algoritmo-Procesamiento-Gramática (G), el tiempo de ejecución de este algoritmo es determinado por el número de símbolos no terminales de la gramática. Sea n_g el número de símbolos no terminales de la gramática. Podemos notar de las líneas 6-9 del Algoritmo-Procesamiento-Gramática (G) que el tiempo de ejecución es $\theta(n_g^2)$.

La línea 2 hace referencia al Algoritmo-Construcción-Diccionario-Pronunciación (D, G), si el diccionario de entrada es implementado como una tabla hash, entonces por las líneas 2-8 que el Algoritmo-Construcción-Diccionario-Pronunciación (D, G) es lineal en el número de símbolos no terminales de la gramática. Por tanto su tiempo de ejecución es de $\theta(n_g)$.

Las líneas 3-5 hace referencia al Algoritmo-Generación-Aleatoria-Sentencias (G), si se generan N sentencias con un tamaño máximo Q palabras y la pila del algoritmo almacena como máximo P elementos, entonces el tiempo de ejecución es de $\theta(NQP)$.

Las líneas 7-9 hace referencia al Algoritmo-Construcción-Archivos-Transcripción $(S_i$,$Dic)$, el cual depende del número de palabras y el número de sentencias generadas, entonces el tiempo de ejecución es de $\theta(n_g N)$, donde N es el número de sentencias generadas

Las líneas 10-12 hace referencia al Algoritmo-MFCC$(xa_i, T, \|frame\|)$, como fue descrito en la Sección 2, la Sección 2.2 y la Sección 3, el tiempo de ejecución en la etapa de extracción de características es dominado por el algoritmo de la transformada rápida de Fourier. Luego el tiempo de ejecución es $\theta(N\|frame\| \lg \|frame\|)$.

Las líneas 13-28 hacen referencia al proceso de construcción de HMM por fonema, entrenamiento del HMM-compuesto y creación de los modelos HMM dependientes del contexto. como fue descrito en la Sección 4.3, el tiempo de ejecución dominante es el del algoritmo Baum-Welch. Si en el análisis del peor caso se realizan W iteraciones, entonces el tiempo de ejecución del algoritmo de entrenamiento

del HMM es de $O(W^2T)$.

En el caso de los Modelos Ocultos de Markov, un análisis del tiempo de ejecución del algoritmo fue mostrado en la Sección 4.3 es de $\theta(N^2T)$, donde N es el número de estados del modelo y T es el número de observaciones. La complejidad del algoritmo de entrenamiento está dominada por el número de iteraciones que se realiza el algoritmo BaumWelch, si en el análisis del peor caso se realizan W iteraciones, entonces el tiempo de ejecución del algoritmo de entrenamiento del HMM es de $O(Wf^2t)$, donde f es el número de estados del hmm-trifonema compuesto y t es el tamaño máximo de los vectores de observación.

Luego el tiempo de ejecución es dado por la suma de estos tiempos individuales.

7.5.2. Análisis Etapa Recuperación de información

Las líneas 29-31 el ALGORITMO-TOKEN-PASSING-CFG($Grafo$,Dic,G,$mfcc_i$,$hmmTrifonema$), por las líneas - del ALGORITMO-TOKEN-PASSING-CFG($Grafo$,Dic,G,$mfcc_i$,$hmmTrifonema$) vemos que el tiempo de ejecución depente del tamaño del vector de observaciones t y del número de tokens k. Por tanto el tiempo de ejecución es $\theta(tk)$

Capítulo 8

Discusión

Los resultados por sentencia de palabras de la Sección 7.1 fueron obtenidos procesando cada sentencia con el algoritmo propuesto, donde se calculó el número de palabras correctas por sentencia H, el número de eliminaciones de palabras de una sentencia E, el número de sustituciones S, el número de inserciones I. Luego con estos valores fueron calculados los valores de *Correctas* y de *Presición*.

Por ejemplo la sentencia número 126 de la Tabla *Resultados por sentencias*, el porcentaje de *Correctas* fue 100 %, el valor *Precisión* fue 80, esto se debe a que la primera y última palabra no fueron reconocidas de manera correcta, por eso hubieron 2 inserciones, en un total de 10 palabras. En la siguiente tabla se muestra los resultados del procesamiento de la sentencia 126, donde la transcripción verdadera del archivo de audio tiene la etiqueta *testT0126.lab*, y la sentencia reconocida tiene la etiqueta *testT0126.rec*.

sentencia 126						
testT0126.lab testT0126.rec	TELEFONO CUATRO TRES CUATRO DOS NUEVE UNO OCHO DOS DOS SENT-START TELEFONO CUATRO TRES CUATRO DOS NUEVE UNO OCHO DOS DOS SENT-END					
Correctas	Precisión	H	E	S	I	N
100	80	10	0	0	2	10

En los resultados especificados en la Tabla *Resultados por sentencias* se dio un caso particular con la sentencia 29, en dicha sentencia el valor de *Correctas* fue 50 % y el valor *Precisión* fue -50, esto se debió a que la primera y última palabra no fueron reconocidas de manera correcta, por eso hubieron 2 inserciones de las cuales una tiene un error por sustitución, donde fue sustituida la palabra *MARCAR* por la palabra *LLAMAR* de un total de 2 palabras.

sentencia 29						
testT0029.lab testT0029.rec	LLAMAR LUIS SENT-START MARCAR LUIS SENT-END					
Correctas	Precisión	H	E	S	I	N
50.00	-50.00	1	0	1	2	2

Otro resultado particular en la Tabla *Resultados por sentencias* fue en la sentencia 149 en donde el valor de *Precisión* obtenido fue 0.0, pues se hicieron inserciones de 2 palabras *TELEFONO* y *CINCO* y una sustitución, la sustitución de la palabra *SIETE* por la palabra *CERO*. el valor de Correctas fue de 66.7 %

sentencia 149						
testT0149.lab testT0149.rec	TELEFONO SIETE CINCO SENT-START TELEFONO CERO CINCO SENT-END					
Correctas	Precisión	H	E	S	I	N
66.67	0.00	2	0	1	2	3

La Tabla *Resultados por sentencias* de manera general muestra un valor de *Correctas* de 100 %, este valor es alto pues existen inserciones que siguen la gramática dada y hacen que el clasificador

corrija e inserte ciertas palabras no reconocidas, sin embargo el valor *Precisión* en las sentencias varía de -50.00 en la sentencia 29 a 96.32 en la sentencia 98. La precisión brinda idea del rol del modelo del lenguaje representando por la gramática definida en A.7 en la tasa de aciertos del clasificador. El algoritmo presentado en esta tesis funcionará bien cuando se tiene conocimiento a priori de la distribución de las palabras en los archivos de audio, es decir se conoce el modelo del lenguaje con el que se va a tratar.

Trabajar con modelos de lenguaje más complejos, requiere utilizar conceptos más avanzados en el reconocimiento automático del habla de palabras continuas de gran vocabulario, esto se conoce como el problema LVSRS y existen diversos enfoques para tratarlo pero aún es un problema abierto en constante investigación.

Tanto el grado de sustitución como eliminación fueron bajos, siendo más alto las inserciones. Todos los experimentos tienen un grado de inserción mínimo de 2, esto se debe a que la gramática tiene dos palabras especiales para iniciar y finalizar una sentencia: SENT-START y SENT-END, las cuales no formaron parte del conjunto de test ni de entrenamiento del clasificador.

La Tabla 7.5 muestra 17 experimentos realizados variando los parámetros p y q. Cada experimento fue hecho usando 200 sentencias generadas aleatoriamente, con un promedio de 10 palabras por sentencia, esto hace un total de 17 x 200 sentencias y 17 x 200 x 10 palabras en promedio. Analizando la precisión podemos notar que los valores de p muy altos cercanos a 100 decrementan la precisión. El valor de *Correctas* fue 99.83 %, el valor *Precisión* está en un rango de 63.65 a 64.96.

La Tabla 7.6 describe la validación del clasificador. Estos resultados muestran que el error promedio del es de 1.18 % después de 20 iteraciones. Esta validación experimental brinda una idea del desempeño del clasificador usando los datos originales como test y entrenamiento escogidos de manera aleatoria, donde se escogen $k-1$ grupos de entrenamiento y 1 grupo de test, esto se realiza k veces. El valor de k utilizado fue de 10, que es un valor que es usado generalmente [DHS01].

La prueba de significancia de t-student proporciona una estimación de la frecuencia con que podrían ocurrir por azar los resultados experimentales. Los resultados de una prueba de este tipo se plantean como una prueba de probabilidad, indicando las posibilidades de que la diferencia observada se haya debido al azar. Fue aceptada la hipótesis nula con un nivel de significancia de 0.05.

El tiempo de ejecución del algoritmo es lineal en la etapa de decodificación, y es dominado en la etapa de entrenamiento por el entrenamiento de los HMM.

Capítulo 9

Conclusiones

En esta tesis describimos un algoritmo para recuperación de información en textos hablados dependientes del hablante, el cual usa información a priori determinada por el lenguaje generado usando una gramática libre del contexto, el abordaje utilizado fue emplear técnicas de extracción de características, de clasificación de patrones y de reconocimiento automático del habla.

El algoritmo planteado está basado en el diseño de un modelo de lenguaje definido por una gramática libre del contexto y el diccionario de pronunciación. Un modelo acústico dado por los coeficientes MFCC, por los modelos HMM de los fonemas y la construcción de los trifonemas. Y el decodificador dado por el algoritmo Token-passing. Los resultados obtenidos actualmente muestran que el algoritmo propuesto puede ser usado en aplicaciones donde sea posible definir un modelo de lenguaje a priori para sistemas dependientes del hablante.

El error obtenido en los experimentos como muestran los resultados por sentencia de palabras, validación cruzada k-fold, así como los resultados obtenidos variando los parámetros de penalidad por inserción y factor de escala de la gramática, muestran un error del clasificador muy pequeño, siendo el error de 0.0 % en los experimentos de sentencia por palabras, 0.17 % variando los parámetros de penalidad por inserción y factor de escala y 1.18 % usando validación cruzada k-fold.

Los resultados obtenidos también muestran la importancia del modelo del lenguaje, el valor *Precisión* muestra esta importancia, pues este valor es dependiente de las inserciones realizadas por el algoritmo, estas inserciones dependen de la gramática y del grafo de reconocimiento generado, el cual es construido a partir de conocimiento a priori del lenguaje. Como trabajo futuro es posible investigar ls recuperación de información de archivos de audio con habla cuando no se tiene el conocimiento a priori del modelo del lenguaje. Un análisis cuidadoso de los resultados, permiten observar que si bien se tiene un error pequeño, ciertas sentencias tienen el valor *Precisión* menor que cero, esto indica que sin un buen modelo del lenguaje, el desempeño del clasificador se ve afectado de gran manera.

El análisis del tiempo de ejecución del algoritmo indica que el proceso de recuperación es rápido, siendo del orden $\theta(tk)$ esto indica que la recuperación de información es realizada rápidamente.

Para todos los test experimentales fue utilizado el mismo ambiente de grabación, el cual fue un ambiente controlado, sin ruido de fondo. Fueron grabadas 200 sentencias dependientes del hablante, las cuales fueron generadas aleatoriamente entre un rango de 1 a 30 palabras, haciendo un promedio de 15 palabras por sentencia, y 5 fonemas en promedio por palabra, haciendo un total de $200 \times 15 \times 5$ fonemas generados.

Modelos de recuperación simples como archivos de grabaciones telefónicas son relativamente menos complicados que grabaciones de conversaciones reales con ruido de fondo incluido, como consecuencia, es posible aplicar esta tecnología a situaciones donde se tenga un número limitado de términos y el entorno sea limpio de ruido.

Capítulo 10

Recomendaciones

Existe la posibilidad de explorar modelos del lenguaje como los n-grams [JM09] los cuales permitirán construir gramáticas mas complejas asignando valor de probabilidades diferentes a las transiciones definidas por la gramática libre del contexto. Se debe investigar la posibilidad de recuperar información de ambientes ruidosos, por ejemplo ruido de ambiente. Se debe también investigar la recuperación de información de audio cuando existen varias voces presentes al mismo tiempo. Se debe también investigar el hecho de como recuperar información cuando no se tiene conocimiento a priori del modelo del lenguaje
De manera general el algoritmo presentado es útil en ambientes con ruido controlado, una voz presente por instante de tiempo, en sistemas dependientes del hablante, siendo posible extender los resultados presentados para sistemas independientes del hablante, con técnicas conocidas de adaptación del modelo acústico [JM09].

Apéndice A

Apéndice

El presente capitulo presenta de manera detallada los diversos archivos de configuración usados en la presente tesis, así como también el código fuente de la aplicación usada.

En la Sección A.7 se muestra la gramática usada, así como el grafo de palabras generado. En la Sección A.2 se muestra el diccionario usado y la lista de fonemas asociada al diccionario. La Sección A.3 muestra las 200 sentencias generadas aleatoriamente a partir de la gramática, las cuales fueron grabadas y sirvieron para testar el algoritmo propuesto. La Sección A.4 se encuentran los archivos de transcripción de las 200 sentencias generadas, que servirán para entrenar al algoritmo. Así como también los archivos de transcripción a nivel fonema. La Sección A.5 contiene las definiciones de los HMM's. La Sección A.6 contiene los trifonemas construidos

A.1. Gramática

La gramática a usar es una gramática libre del contexto 4.6.2. Cabe resaltar que el diseño de la gramática debe ser realizado teniendo en cuenta el lenguaje que se quiere reconocer. La gramática a continuación genera un lenguaje donde la primera palabra corresponde a la cadena *TELEFONO* inmediatamente seguido de una secuencia de cadenas que corresponden a dígitos, luego sigue la cadena *LLAMAR* ó *MARCAR*, para finalizar en una cadena que representa el nombre de una persona, los corchetes indican que la cadena dentro de estos, puede ocurrir cero o una vez.

```
$digit = UNO | DOS | TRES | CUATRO | CINCO
        | SEIS | SIETE | OCHO | NUEVE | CERO;

$name  =[JORGE] LUIS |
        [LEISSI] LEON |
        [CARLOS] DIAZ |
        [LUIS] GUEVARA;

(SENT-START}(TELEFONO <$digit>| (LLAMAR|MARCAR)$name)SENT-END)}
```

A.1.1. Grafo de Palabras

La gramática anterior genera un grafo. Este grafo es mostrado en el formato SLF (Standard lattice Format), que es una notación de bajo nivel en la cual cada palabra y cada transición entre palabras es listada de manera explícita [YEG+06], la herramienta usada para generarla fue HTK.

```
# Define el tamaño del grafo: N=número de nodos, L=numero de arcos.
            N=28   L=55
# Lista de nodos: I=numero del nodo, W=palabra
            I=0    W=SENT-END
            I=1    W=GUEVARA
            I=2    W=!NULL
            I=3    W=LUIS
```

```
            I=4    W=DIAZ
            I=5    W=CARLOS
            I=6    W=LEON
            I=7    W=LEISSI
            I=8    W=LUIS
            I=9    W=JORGE
            I=10   W=MARCAR
            I=11   W=!NULL
            I=12   W=LLAMAR
            I=13   W=CERO
            I=14   W=!NULL
            I=15   W=NUEVE
            I=16   W=OCHO
            I=17   W=SIETE
            I=18   W=SEIS
            I=19   W=CINCO
            I=20   W=CUATRO
            I=21   W=TRES
            I=22   W=DOS
            I=23   W=UNO
            I=24   W=TELEFONO
            I=25   W=SENT-START
            I=26   W=!NULL
            I=27   W=!NULL
# Lista de arcos: J=numero de arco, S=nodo inicial, E=nodo final
            J=0     S=2    E=0
            J=1     S=14   E=0
            J=2     S=3    E=1
            J=3     S=11   E=1
            J=4     S=1    E=2
            J=5     S=4    E=2
            J=6     S=6    E=2
            J=7     S=8    E=2
            J=8     S=11   E=3
            J=9     S=5    E=4
            J=10    S=11   E=4
            J=11    S=11   E=5
            J=12    S=7    E=6
            J=13    S=11   E=6
            J=14    S=11   E=7
            J=15    S=9    E=8
            J=16    S=11   E=8
            J=17    S=11   E=9
            J=18    S=25   E=10
            J=19    S=10   E=11
            J=20    S=12   E=11
            J=21    S=25   E=12
            J=22    S=14   E=13
            J=23    S=24   E=13
            J=24    S=13   E=14
            J=25    S=15   E=14
            J=26    S=16   E=14
            J=27    S=17   E=14
            J=28    S=18   E=14
            J=29    S=19   E=14
            J=30    S=20   E=14
            J=31    S=21   E=14
            J=32    S=22   E=14
            J=33    S=23   E=14
```

```
J=34    S=14   E=15
J=35    S=24   E=15
J=36    S=14   E=16
J=37    S=24   E=16
J=38    S=14   E=17
J=39    S=24   E=17
J=40    S=14   E=18
J=41    S=24   E=18
J=42    S=14   E=19
J=43    S=24   E=19
J=44    S=14   E=20
J=45    S=24   E=20
J=46    S=14   E=21
J=47    S=24   E=21
J=48    S=14   E=22
J=49    S=24   E=22
J=50    S=14   E=23
J=51    S=24   E=23
J=52    S=25   E=24
J=53    S=27   E=25
J=54    S=0    E=26
```

A.2. Diccionario

El diccionario debe construirse de acuerdo con la gramática, donde a cada palabra es asociado una secuencia de fonemas que corresponden a la pronunciación de cada palabra. A continuación el diccionario de pronunciación siguiendo el estándar IPA [AC99]

A.2.1. Diccionario de Pronunciación

```
CARLOS          k ah r l oh s sp
CERO            th eh r oh sp
CINCO           th ih ng k oh sp
CUATRO          k w ah t r oh sp
DIAZ            dh ih ah s sp
DOS             dh oh s sp
GUEVARA         g eh b ah r ah sp
JORGE           j oh r j eh sp
LEISSI          l eh y s ih sp
LEON            l eh oh n sp
LLAMAR          ll ah m ah r sp
LUIS            l uh y s sp
MARCAR          m ah r k ah r sp
NUEVE           n w eh b eh sp
OCHO            oh ch oh sp
SEIS            s eh y s sp
SENT-END   []   sil
SENT-START []   sil
SIETE           s y eh t eh sp
silence         sil
TELEFONO        t eh l eh f oh n oh sp
TRES            t r eh s sp
UNO             uh n oh sp
```

A.2.2. Lista de Fonemas

```
k
ah
```

r
l
oh
s
sp
th
eh
ih
ng
w
t
dh
g
b
j
y
n
ll
m
uh
ch
sil
f

A.3. Sentencias Generadas Aleatoriamente

Fueron generadas 200 sentencias de manera aleatoria las que se muestran a continuación

1. TELEFONO DOS SEIS CUATRO TRES UNO OCHO DOS CINCO CINCO CUATRO UNO SEIS CUATRO SEIS CERO CINCO SIETE CUATRO NUEVE CERO
2. LLAMAR LUIS GUEVARA
3. TELEFONO SEIS SIETE SIETE UNO UNO DOS DOS UNO SIETE CINCO CUATRO OCHO
4. LLAMAR DIAZ
5. LLAMAR GUEVARA
6. MARCAR LEON
7. MARCAR LUIS GUEVARA
8. TELEFONO OCHO SIETE SIETE DOS CERO TRES OCHO CERO CUATRO
9. TELEFONO UNO
10. LLAMAR LEON
11. LLAMAR LUIS
12. TELEFONO SEIS TRES
13. LLAMAR LEISSI LEON
14. TELEFONO DOS OCHO NUEVE SIETE CUATRO CINCO CERO DOS CERO TRES CINCO CINCO SEIS NUEVE NUEVE CINCO OCHO DOS
15. MARCAR JORGE LUIS
16. TELEFONO CERO CINCO DOS OCHO CINCO CUATRO OCHO CUATRO OCHO TRES CINCO SIETE CINCO SEIS TRES UNO SIETE CINCO CINCO CINCO TRES CERO DOS
17. LLAMAR DIAZ
18. MARCAR LUIS GUEVARA
19. TELEFONO UNO UNO SIETE OCHO DOS TRES CUATRO CERO CERO DOS SEIS CINCO CUATRO SIETE SEIS
20. LLAMAR GUEVARA
21. LLAMAR LEISSI LEON
22. LLAMAR LEISSI LEON
23. LLAMAR LEISSI LEON
24. LLAMAR JORGE LUIS
25. LLAMAR LEISSI LEON
26. TELEFONO SEIS TRES OCHO OCHO OCHO CINCO OCHO OCHO CINCO UNO
27. TELEFONO UNO CINCO NUEVE DOS CINCO CUATRO
28. TELEFONO DOS CUATRO CERO SEIS CINCO SIETE NUEVE NUEVE SEIS CUATRO SEIS
29. LLAMAR LUIS
30. TELEFONO SIETE
31. LLAMAR LUIS
32. LLAMAR DIAZ
33. LLAMAR JORGE LUIS
34. LLAMAR DIAZ
35. MARCAR LEON
36. MARCAR LUIS
37. MARCAR LEISSI LEON
38. MARCAR JORGE LUIS
39. LLAMAR JORGE LUIS
40. MARCAR LEISSI LEON
41. LLAMAR DIAZ
42. MARCAR LUIS
43. MARCAR JORGE LUIS
44. LLAMAR LUIS GUEVARA
45. LLAMAR LUIS GUEVARA
46. LLAMAR LEON
47. LLAMAR LEISSI LEON
48. LLAMAR DIAZ
49. MARCAR CARLOS DIAZ
50. LLAMAR GUEVARA

51. TELEFONO SEIS DOS CUATRO SIETE
52. LLAMAR LEON
53. TELEFONO UNO UNO CINCO
54. MARCAR DIAZ
55. MARCAR LEON
56. MARCAR CARLOS DIAZ
57. TELEFONO DOS
58. TELEFONO CERO DOS SIETE UNO CUATRO NUEVE NUEVE CERO TRES CUATRO CUATRO OCHO NUEVE OCHO NUEVE SIETE UNO CUATRO CINCO CUATRO DOS SEIS CUATRO CINCO SEIS CINCO CINCO TRES SIETE DOS CUATRO SEIS CINCO SEIS CERO DOS CERO
59. MARCAR DIAZ
60. LLAMAR DIAZ
61. TELEFONO NUEVE SEIS
62. LLAMAR DIAZ
63. LLAMAR LUIS
64. TELEFONO SEIS SIETE CERO NUEVE
65. TELEFONO CERO
66. TELEFONO SEIS NUEVE UNO NUEVE CINCO NUEVE OCHO DOS TRES SIETE SIETE CERO CERO NUEVE UNO UNO TRES DOS CUATRO DOS UNO CINCO CINCO CUATRO SEIS
67. TELEFONO SIETE TRES SEIS SEIS SIETE TRES NUEVE SEIS NUEVE TRES DOS CUATRO DOS
68. MARCAR LEON
69. TELEFONO SEIS UNO CERO UNO
70. TELEFONO DOS
71. TELEFONO SIETE DOS OCHO SEIS CERO NUEVE SIETE SIETE OCHO UNO UNO OCHO SEIS TRES CINCO SIETE OCHO OCHO DOS OCHO UNO DOS CINCO OCHO
72. TELEFONO UNO UNO CERO CERO CINCO UNO
73. LLAMAR LEISSI LEON
74. TELEFONO CUATRO CINCO UNO SEIS OCHO DOS
75. TELEFONO SEIS SIETE CERO TRES CINCO TRES TRES TRES
76. MARCAR LEISSI LEON
77. TELEFONO TRES TRES CINCO UNO
78. TELEFONO DOS DOS OCHO CINCO DOS SEIS
79. LLAMAR CARLOS DIAZ
80. LLAMAR GUEVARA
81. TELEFONO SEIS
82. TELEFONO OCHO SIETE CERO OCHO
83. MARCAR LUIS
84. TELEFONO SIETE UNO CERO CINCO UNO TRES SIETE CERO CINCO CERO NUEVE NUEVE TRES CERO DOS TRES
85. TELEFONO TRES OCHO CERO
86. TELEFONO OCHO TRES DOS NUEVE SIETE CINCO CUATRO CERO SIETE OCHO OCHO CUATRO NUEVE UNO DOS
87. LLAMAR DIAZ
88. LLAMAR LEISSI LEON
89. LLAMAR CARLOS DIAZ
90. LLAMAR LUIS
91. LLAMAR CARLOS DIAZ
92. TELEFONO DOS UNO CINCO DOS TRES CERO
93. TELEFONO OCHO DOS DOS
94. MARCAR LEISSI LEON
95. MARCAR LUIS GUEVARA
96. TELEFONO UNO
97. LLAMAR CARLOS DIAZ
98. TELEFONO TRES CUATRO DOS OCHO CINCO CUATRO UNO CUATRO CINCO DOS CERO SIETE CERO TRES TRES SIETE CERO CINCO TRES SIETE CINCO CINCO SIETE CERO UNO SEIS SIETE OCHO DOS UNO UNO OCHO UNO NUEVE NUEVE NUEVE UNO DOS DOS CUATRO SIETE SIETE SIETE CUATRO TRES CERO CUATRO DOS NUEVE TRES CINCO TRES
99. MARCAR LEISSI LEON
100. LLAMAR LUIS
101. LLAMAR LUIS
102. LLAMAR DIAZ
103. TELEFONO UNO
104. LLAMAR GUEVARA
105. MARCAR CARLOS DIAZ
106. TELEFONO DOS CUATRO CINCO CUATRO CERO NUEVE CERO TRES TRES NUEVE TRES TRES NUEVE
107. LLAMAR JORGE LUIS
108. LLAMAR CARLOS DIAZ
109. TELEFONO SEIS TRES DOS TRES SEIS UNO SIETE TRES SIETE OCHO CERO DOS OCHO CINCO DOS CERO CERO NUEVE CUATRO SIETE SIETE CERO UNO SEIS TRES CUATRO
110. TELEFONO UNO CINCO SEIS NUEVE SIETE SIETE OCHO CUATRO CUATRO OCHO CINCO
111. TELEFONO CINCO SEIS SIETE SIETE CINCO OCHO UNO CINCO CINCO
112. LLAMAR CARLOS DIAZ
113. TELEFONO SEIS SEIS TRES CINCO CINCO
114. MARCAR JORGE LUIS
115. TELEFONO SIETE
116. LLAMAR LUIS GUEVARA
117. LLAMAR JORGE LUIS
118. TELEFONO CERO NUEVE OCHO UNO CINCO
119. MARCAR LEISSI LEON
120. TELEFONO DOS NUEVE SIETE CINCO DOS OCHO CERO
121. MARCAR GUEVARA
122. MARCAR LUIS
123. TELEFONO UNO
124. LLAMAR DIAZ
125. MARCAR LUIS
126. TELEFONO CUATRO TRES CUATRO DOS NUEVE UNO OCHO DOS DOS
127. MARCAR CARLOS DIAZ
128. LLAMAR JORGE LUIS
129. TELEFONO UNO CERO SIETE SIETE CUATRO SIETE SEIS TRES
130. TELEFONO SIETE DOS
131. MARCAR LUIS GUEVARA
132. MARCAR JORGE LUIS
133. MARCAR LEISSI LEON
134. MARCAR LEON
135. MARCAR LUIS GUEVARA
136. LLAMAR LEISSI LEON
137. TELEFONO TRES CUATRO UNO UNO DOS UNO OCHO SEIS SIETE CERO UNO TRES OCHO DOS SIETE SEIS TRES DOS NUEVE CERO CINCO
138. LLAMAR DIAZ
139. TELEFONO CERO CINCO OCHO TRES
140. LLAMAR JORGE LUIS

```
141. TELEFONO CERO OCHO CINCO NUEVE CINCO CINCO DOS TRES DOS UNO CINCO TRES DOS SIETE
     CINCO UNO SEIS DOS CUATRO CINCO NUEVE NUEVE SEIS SIETE CINCO TRES NUEVE OCHO OCHO
142. MARCAR JORGE LUIS
143. MARCAR LUIS
144. LLAMAR GUEVARA
145. LLAMAR DIAZ
146. MARCAR DIAZ
147. MARCAR GUEVARA
148. MARCAR LUIS GUEVARA
149. TELEFONO SIETE CINCO
150. LLAMAR LUIS GUEVARA
151. LLAMAR LEON
152. LLAMAR LUIS GUEVARA
153. TELEFONO SIETE SEIS CERO CERO SEIS SIETE OCHO SEIS CUATRO TRES SEIS SIETE CUATRO
     CUATRO SIETE SIETE NUEVE CERO DOS OCHO UNO CERO TRES CINCO OCHO DOS CUATRO CERO
     OCHO TRES CUATRO NUEVE DOS SIETE CERO SIETE CUATRO NUEVE CERO SEIS OCHO
154. LLAMAR LEON
155. LLAMAR DIAZ
156. LLAMAR CARLOS DIAZ
157. TELEFONO UNO NUEVE CERO TRES NUEVE UNO TRES NUEVE OCHO OCHO
158. LLAMAR JORGE LUIS
159. TELEFONO NUEVE TRES SEIS OCHO NUEVE CINCO NUEVE NUEVE CUATRO UNO SIETE SIETE SEIS
     UNO UNO OCHO SIETE OCHO CUATRO SIETE SEIS DOS CERO CINCO
160. TELEFONO SIETE UNO
161. TELEFONO DOS CERO SEIS UNO SIETE DOS CINCO
162. LLAMAR LUIS GUEVARA
163. TELEFONO CINCO CINCO CUATRO CUATRO SIETE CUATRO NUEVE OCHO CINCO NUEVE CUATRO
     NUEVE
164. TELEFONO CERO OCHO TRES CUATRO UNO UNO TRES SIETE
165. LLAMAR JORGE LUIS
166. LLAMAR GUEVARA
167. MARCAR LUIS GUEVARA
168. TELEFONO SIETE CUATRO SEIS SIETE SEIS NUEVE
169. MARCAR DIAZ
170. TELEFONO UNO
171. TELEFONO SIETE UNO TRES SIETE CUATRO NUEVE CINCO DOS CINCO UNO CUATRO SIETE
172. LLAMAR LUIS
173. LLAMAR JORGE LUIS
174. MARCAR DIAZ
175. TELEFONO CERO SEIS DOS OCHO SEIS CERO CUATRO SEIS CINCO CUATRO DOS CINCO SIETE UNO
     SEIS TRES SIETE
176. TELEFONO SIETE SIETE TRES CERO CINCO SEIS DOS UNO NUEVE SIETE TRES CINCO CUATRO
     CERO CINCO SIETE CUATRO SIETE SIETE CUATRO CINCO UNO OCHO UNO
177. MARCAR LEISSI LEON
178. TELEFONO TRES SIETE CINCO TRES SEIS OCHO CERO DOS SIETE SEIS
179. MARCAR LUIS GUEVARA
180. TELEFONO SIETE CUATRO SEIS CINCO TRES TRES
181. LLAMAR LUIS
182. LLAMAR JORGE LUIS
183. MARCAR JORGE LUIS
184. MARCAR GUEVARA
185. TELEFONO CUATRO DOS TRES
186. LLAMAR GUEVARA
187. MARCAR JORGE LUIS
188. TELEFONO SIETE CERO UNO CINCO SEIS CERO SIETE TRES SEIS SEIS UNO UNO UNO CINCO
189. LLAMAR JORGE LUIS
190. MARCAR LEISSI LEON
191. LLAMAR GUEVARA
192. LLAMAR LUIS
193. TELEFONO DOS TRES SIETE
194. LLAMAR CARLOS DIAZ
195. TELEFONO CERO
196. MARCAR GUEVARA
197. MARCAR CARLOS DIAZ
198. TELEFONO CINCO CERO OCHO SEIS SEIS CERO DOS UNO CERO CINCO OCHO CUATRO TRES SEIS
     CINCO SIETE TRES DOS CINCO CUATRO CERO
199. LLAMAR LEON
200. TELEFONO SEIS CUATRO CERO DOS UNO
```

A.4. Archivos de Transcripción

Cada sentencia generada es asociada a un archivo de transcripción con la información asociada-da, necesaria para el entrenamiento de los modelos ocultos de Markov.

```
#!MLF!#                "*/T0001.lab"          TELEFONO               DOS                    SEIS                   CUATRO                 TRES                   UNO                    OCHO                   DOS                    CINCO                  CINCO                  CUATRO                 UNO                    SEIS                   CUATRO                 CERO                   CINCO                  SIETE                  SIETE                  UNO                    UNO                    DOS                    DOS                    UNO                    SIETE                  CUATRO                 NUEVE                  CERO                   .                      "*/T0002.lab"          LLAMAR                 LUIS                   GUEVARA                .                      "*/T0003.lab"          TELEFONO               SEIS                   SIETE                  SIETE                  UNO                    UNO                    DOS                    DOS                    UNO                    SIETE                  CINCO                  CUATRO                 LLAMAR                 LLAMAR                 DIAZ                   .                      "*/T0005.lab"          LLAMAR                 GUEVARA                .                      "*/T0006.lab"          MARCAR                 LEON                   .                      "*/T0007.lab"          MARCAR                 LUIS                   GUEVARA                .                      "*/T0008.lab"          TELEFONO               OCHO                   SIETE                  SIETE                  DOS                    CERO                   TRES                   OCHO                   CERO                   CUATRO                 .                      "*/T0009.lab"          TELEFONO               UNO                    .                      "*/T0010.lab"          LLAMAR                 LEON                   .                      "*/T0011.lab"          LLAMAR                 LUIS                   .                      "*/T0012.lab"          TELEFONO               SEIS                   TRES                   .                      "*/T0013.lab"          LLAMAR                 LEISSI                 LEON                   .                      "*/T0014.lab"          TELEFONO               DOS                    OCHO                   NUEVE                  .                      SIETE                  CUATRO                 CINCO                  CERO                   DOS                    CERO                   TRES                   CINCO                  CINCO                  SEIS                   NUEVE                  NUEVE                  CINCO                  OCHO                   DOS                    .                      "*/T0015.lab"          MARCAR                 JORGE                  LUIS                   .                      "*/T0016.lab"          TELEFONO               CERO                   CINCO                  DOS                    OCHO                   CINCO                  CUATRO                 OCHO                   CUATRO                 OCHO                   TRES                   CINCO                  SIETE                  CINCO                  SEIS                   TRES                   UNO                    SIETE
```

```
CINCO
CINCO
CINCO
TRES
CERO
DOS
.
"*/T0017.lab"
LLAMAR
DIAZ
.
"*/T0018.lab"
MARCAR
LUIS
GUEVARA
.
"*/T0019.lab"
TELEFONO
UNO
UNO
SIETE
OCHO
DOS
TRES
CUATRO
CERO
CERO
DOS
SEIS
CINCO
CUATRO
SIETE
SEIS
.
"*/T0020.lab"
LLAMAR
GUEVARA
.
"*/T0021.lab"
LLAMAR
LEISSI
LEON
.
"*/T0022.lab"
LLAMAR
LEISSI
LEON
.
"*/T0023.lab"
LLAMAR
LEISSI
LEON
.
"*/T0024.lab"
LLAMAR
JORGE
LUIS
.
"*/T0025.lab"
LLAMAR
LEISSI
LEON
.
"*/T0026.lab"
TELEFONO
SEIS
TRES
OCHO
OCHO
OCHO
CINCO
OCHO
OCHO
CINCO
UNO
.
"*/T0027.lab"
TELEFONO
UNO
CINCO
NUEVE
DOS
CINCO
CUATRO
.
"*/T0028.lab"
TELEFONO
DOS
CUATRO
CERO
SEIS
CINCO
SIETE
NUEVE
NUEVE
SEIS
CUATRO
SEIS
.
"*/T0029.lab"
LLAMAR
LUIS
.
"*/T0030.lab"
TELEFONO
SIETE
.
"*/T0031.lab"
LLAMAR
LUIS
.
"*/T0032.lab"
LLAMAR
DIAZ
.
"*/T0033.lab"
LLAMAR
JORGE
LUIS
.
"*/T0034.lab"
LLAMAR
DIAZ
.
"*/T0035.lab"
MARCAR
LEON
.
"*/T0036.lab"
MARCAR
LUIS
.
"*/T0037.lab"
MARCAR
LEISSI
LEON
.
"*/T0038.lab"
MARCAR
JORGE
LUIS
.
"*/T0039.lab"
LLAMAR
JORGE
LUIS
.
"*/T0040.lab"
MARCAR
LEISSI
LEON
.
"*/T0041.lab"
LLAMAR
DIAZ
.
"*/T0042.lab"
MARCAR
LUIS
.
"*/T0043.lab"
MARCAR
JORGE
LUIS
.
"*/T0044.lab"
LLAMAR
LUIS
GUEVARA
.
"*/T0045.lab"
LLAMAR
LUIS
GUEVARA
.
"*/T0046.lab"
LLAMAR
LEON
.
"*/T0047.lab"
LLAMAR
LEISSI
LEON
.
"*/T0048.lab"
LLAMAR
DIAZ
.
"*/T0049.lab"
MARCAR
CARLOS
DIAZ
.
"*/T0050.lab"
LLAMAR
GUEVARA
.
"*/T0051.lab"
TELEFONO
SEIS
DOS
CUATRO
SIETE
.
"*/T0052.lab"
LLAMAR
LEON
.
"*/T0053.lab"
TELEFONO
UNO
UNO
CINCO
.
"*/T0054.lab"
MARCAR
DIAZ
.
"*/T0055.lab"
MARCAR
LEON
.
"*/T0056.lab"
MARCAR
CARLOS
DIAZ
.
"*/T0057.lab"
TELEFONO
DOS
.
"*/T0058.lab"
TELEFONO
CERO
DOS
SIETE
UNO
CUATRO
NUEVE
NUEVE
CERO
TRES
CUATRO
CUATRO
OCHO
NUEVE
OCHO
NUEVE
SIETE
UNO
CUATRO
CINCO
CUATRO
DOS
SEIS
CUATRO
CINCO
SEIS
CINCO
CINCO
TRES
SIETE
DOS
CUATRO
SEIS
CINCO
SEIS
CERO
DOS
CERO
.
"*/T0059.lab"
MARCAR
DIAZ
.
"*/T0060.lab"
LLAMAR
DIAZ
.
"*/T0061.lab"
TELEFONO
NUEVE
SEIS
.
"*/T0062.lab"
LLAMAR
DIAZ
.
"*/T0063.lab"
LLAMAR
LUIS
.
"*/T0064.lab"
TELEFONO
SEIS
SIETE
CERO
NUEVE
.
"*/T0065.lab"
TELEFONO
CERO
.
"*/T0066.lab"
TELEFONO
SEIS
NUEVE
UNO
NUEVE
CINCO
NUEVE
OCHO
DOS
TRES
SIETE
SIETE
CERO
CERO
NUEVE
UNO
UNO
TRES
DOS
CUATRO
DOS
UNO
CINCO
CINCO
CUATRO
SEIS
.
"*/T0067.lab"
TELEFONO
SIETE
TRES
SEIS
SEIS
SIETE
TRES
NUEVE
SEIS
NUEVE
TRES
DOS
CUATRO
DOS
.
"*/T0068.lab"
MARCAR
LEON
.
"*/T0069.lab"
TELEFONO
SEIS
UNO
CERO
UNO
.
"*/T0070.lab"
TELEFONO
DOS
.
"*/T0071.lab"
TELEFONO
SIETE
DOS
OCHO
SEIS
CERO
NUEVE
SIETE
SIETE
OCHO
UNO
UNO
OCHO
SEIS
TRES
CINCO
SIETE
OCHO
OCHO
DOS
OCHO
UNO
DOS
CINCO
OCHO
.
"*/T0072.lab"
TELEFONO
UNO
UNO
CERO
CERO
CINCO
UNO
.
"*/T0073.lab"
LLAMAR
LEISSI
LEON
.
"*/T0074.lab"
TELEFONO
CUATRO
CINCO
UNO
SEIS
OCHO
.
"*/T0075.lab"
TELEFONO
SEIS
SIETE
CERO
TRES
CINCO
TRES
TRES
.
"*/T0076.lab"
MARCAR
LEISSI
LEON
.
"*/T0077.lab"
TELEFONO
TRES
TRES
CINCO
UNO
.
"*/T0078.lab"
TELEFONO
DOS
DOS
OCHO
CINCO
DOS
SEIS
.
"*/T0079.lab"
LLAMAR
CARLOS
DIAZ
.
"*/T0080.lab"
LLAMAR
GUEVARA
.
"*/T0081.lab"
TELEFONO
SEIS
.
"*/T0082.lab"
TELEFONO
OCHO
SIETE
CERO
OCHO
.
"*/T0083.lab"
MARCAR
LUIS
.
"*/T0084.lab"
TELEFONO
SIETE
UNO
CERO
CINCO
UNO
TRES
SIETE
CERO
CINCO
CERO
NUEVE
NUEVE
TRES
CERO
DOS
TRES
.
"*/T0085.lab"
TELEFONO
TRES
OCHO
CERO
.
"*/T0086.lab"
TELEFONO
OCHO
TRES
DOS
NUEVE
SIETE
CINCO
CUATRO
CERO
SIETE
OCHO
OCHO
CUATRO
NUEVE
UNO
SEIS
SEIS
UNO
DOS
.
"*/T0087.lab"
LLAMAR
DIAZ
.
"*/T0088.lab"
LLAMAR
LEISSI
LEON
.
"*/T0089.lab"
LLAMAR
CARLOS
DIAZ
.
"*/T0090.lab"
LLAMAR
LUIS
.
"*/T0091.lab"
LLAMAR
CARLOS
DIAZ
.
"*/T0092.lab"
TELEFONO
DOS
UNO
CINCO
DOS
TRES
CERO
.
"*/T0093.lab"
TELEFONO
OCHO
DOS
DOS
.
"*/T0094.lab"
MARCAR
LEISSI
LEON
.
"*/T0095.lab"
MARCAR
LUIS
GUEVARA
.
"*/T0096.lab"
TELEFONO
UNO
.
"*/T0097.lab"
LLAMAR
CARLOS
DIAZ
.
"*/T0098.lab"
TELEFONO
TRES
CUATRO
DOS
OCHO
CINCO
CUATRO
UNO
CUATRO
CINCO
DOS
CERO
SIETE
CERO
TRES
TRES
SIETE
CINCO
CINCO
DOS
CERO
CERO
NUEVE
CUATRO
SIETE
SIETE
CERO
UNO
SEIS
TRES
UNO
.
"*/T0099.lab"
MARCAR
LEISSI
LEON
.
"*/T0100.lab"
LLAMAR
LUIS
.
"*/T0101.lab"
LLAMAR
LUIS
.
"*/T0102.lab"
LLAMAR
DIAZ
.
"*/T0103.lab"
TELEFONO
UNO
.
"*/T0104.lab"
LLAMAR
GUEVARA
.
"*/T0105.lab"
MARCAR
CARLOS
DIAZ
.
"*/T0106.lab"
TELEFONO
DOS
CUATRO
CINCO
CUATRO
CERO
NUEVE
CERO
TRES
TRES
NUEVE
TRES
TRES
NUEVE
.
"*/T0107.lab"
LLAMAR
JORGE
LUIS
.
"*/T0108.lab"
LLAMAR
CARLOS
DIAZ
.
"*/T0109.lab"
TELEFONO
SEIS
TRES
DOS
TRES
SEIS
UNO
SIETE
TRES
SIETE
OCHO
CERO
DOS
OCHO
CINCO
DOS
CERO
CERO
NUEVE
CUATRO
SIETE
SIETE
CERO
UNO
SEIS
TRES
CUATRO
```

```
.
"*/T0110.lab"
TELEFONO
UNO
CINCO
SEIS
NUEVE
SIETE
SIETE
OCHO
CUATRO
CUATRO
OCHO
CINCO
.
"*/T0111.lab"
TELEFONO
CINCO
SEIS
SIETE
SIETE
CINCO
OCHO
UNO
CINCO
CINCO
.
"*/T0112.lab"
LLAMAR
CARLOS
DIAZ
.
"*/T0113.lab"
TELEFONO
SEIS
SEIS
TRES
CINCO
CINCO
.
"*/T0114.lab"
MARCAR
JORGE
LUIS
.
"*/T0115.lab"
TELEFONO
SIETE
.
"*/T0116.lab"
LLAMAR
LUIS
GUEVARA
.
"*/T0117.lab"
LLAMAR
JORGE
LUIS
.
"*/T0118.lab"
TELEFONO
CERO
NUEVE
OCHO
UNO
CINCO
.
"*/T0119.lab"
MARCAR
LEISSI
LEON
.
"*/T0120.lab"
TELEFONO
DOS
NUEVE
SIETE
CINCO
DOS
OCHO
CERO
.
"*/T0121.lab"
MARCAR
GUEVARA
.
"*/T0122.lab"
MARCAR
LUIS
.
"*/T0123.lab"
TELEFONO
UNO
.
"*/T0124.lab"
LLAMAR
DIAZ
.
"*/T0125.lab"
MARCAR
LUIS
.
"*/T0126.lab"
TELEFONO
CUATRO
TRES
CUATRO
DOS
NUEVE
UNO
OCHO
DOS
DOS
.
"*/T0127.lab"
MARCAR
CARLOS
DIAZ
.
"*/T0128.lab"
LLAMAR
JORGE
LUIS
.
"*/T0129.lab"
TELEFONO
UNO
CERO
SIETE
SIETE
CUATRO
SIETE
SEIS
TRES
.
"*/T0130.lab"
TELEFONO
SIETE
DOS
.
"*/T0131.lab"
MARCAR
LUIS
GUEVARA
.
"*/T0132.lab"
MARCAR
JORGE
LUIS
.
"*/T0133.lab"
MARCAR
LEISSI
LEON
.
"*/T0134.lab"
MARCAR
LEON
.
"*/T0135.lab"
MARCAR
LUIS
GUEVARA
.
"*/T0136.lab"
LLAMAR
LEISSI
LEON
.
"*/T0137.lab"
TELEFONO
TRES
CUATRO
UNO
UNO
DOS
UNO
OCHO
SEIS
SIETE
CERO
UNO
TRES
OCHO
DOS
SIETE
SEIS
TRES
DOS
NUEVE
CERO
CINCO
.
"*/T0138.lab"
LLAMAR
DIAZ
.
"*/T0139.lab"
TELEFONO
CERO
CINCO
OCHO
TRES
.
"*/T0140.lab"
LLAMAR
JORGE
LUIS
.
"*/T0141.lab"
TELEFONO
CERO
OCHO
CINCO
NUEVE
CINCO
CINCO
DOS
TRES
DOS
UNO
CINCO
TRES
DOS
SIETE
CINCO
UNO
SEIS
DOS
CUATRO
CINCO
NUEVE
NUEVE
SEIS
SIETE
CINCO
TRES
.
"*/T0142.lab"
MARCAR
JORGE
LUIS
.
"*/T0143.lab"
MARCAR
LUIS
.
"*/T0144.lab"
LLAMAR
GUEVARA
.
"*/T0145.lab"
LLAMAR
DIAZ
.
"*/T0146.lab"
MARCAR
DIAZ
.
"*/T0147.lab"
MARCAR
GUEVARA
.
"*/T0148.lab"
MARCAR
LUIS
GUEVARA
.
"*/T0149.lab"
TELEFONO
SIETE
CINCO
.
"*/T0150.lab"
LLAMAR
LUIS
GUEVARA
.
"*/T0151.lab"
LLAMAR
LEON
.
"*/T0152.lab"
LLAMAR
LUIS
GUEVARA
.
"*/T0153.lab"
TELEFONO
SIETE
SEIS
CERO
CERO
SEIS
SIETE
OCHO
SEIS
CUATRO
TRES
SEIS
SIETE
CUATRO
CUATRO
CUATRO
SIETE
SIETE
NUEVE
CERO
DOS
OCHO
UNO
CERO
OCHO
CINCO
CUATRO
NUEVE
DOS
SIETE
CERO
SIETE
CUATRO
NUEVE
CERO
SEIS
OCHO
.
"*/T0154.lab"
LLAMAR
LEON
.
"*/T0155.lab"
LLAMAR
DIAZ
.
"*/T0156.lab"
LLAMAR
CARLOS
DIAZ
.
"*/T0157.lab"
TELEFONO
UNO
NUEVE
CERO
TRES
NUEVE
UNO
TRES
NUEVE
OCHO
OCHO
.
"*/T0158.lab"
LLAMAR
JORGE
LUIS
.
"*/T0159.lab"
TELEFONO
NUEVE
TRES
SEIS
OCHO
NUEVE
CINCO
NUEVE
NUEVE
CUATRO
UNO
SIETE
SIETE
SEIS
UNO
UNO
OCHO
SIETE
OCHO
CUATRO
SIETE
SEIS
DOS
CERO
CINCO
.
"*/T0160.lab"
TELEFONO
SIETE
UNO
.
"*/T0161.lab"
TELEFONO
DOS
CERO
SEIS
UNO
SIETE
DOS
CINCO
.
"*/T0162.lab"
LLAMAR
LUIS
GUEVARA
.
"*/T0163.lab"
TELEFONO
CINCO
CINCO
CUATRO
CUATRO
SIETE
CUATRO
NUEVE
OCHO
CINCO
NUEVE
CUATRO
NUEVE
.
"*/T0164.lab"
TELEFONO
CERO
OCHO
TRES
CUATRO
UNO
UNO
TRES
SIETE
.
"*/T0165.lab"
LLAMAR
JORGE
LUIS
.
"*/T0166.lab"
LLAMAR
GUEVARA
.
"*/T0167.lab"
MARCAR
LUIS
GUEVARA
.
"*/T0168.lab"
TELEFONO
SIETE
CUATRO
SEIS
SIETE
SEIS
NUEVE
.
"*/T0169.lab"
MARCAR
DIAZ
.
"*/T0170.lab"
TELEFONO
UNO
.
"*/T0171.lab"
TELEFONO
SIETE
UNO
TRES
SIETE
CUATRO
NUEVE
CINCO
DOS
CINCO
UNO
CUATRO
SIETE
.
"*/T0172.lab"
LLAMAR
LUIS
.
"*/T0173.lab"
LLAMAR
JORGE
LUIS
.
"*/T0174.lab"
MARCAR
DIAZ
.
"*/T0175.lab"
TELEFONO
CERO
SEIS
DOS
OCHO
SEIS
CERO
DOS
SEIS
CUATRO
CINCO
SIETE
CUATRO
SIETE
SIETE
CUATRO
CINCO
UNO
OCHO
UNO
.
"*/T0177.lab"
MARCAR
LEISSI
LEON
.
"*/T0178.lab"
TELEFONO
TRES
SIETE
CINCO
TRES
SEIS
OCHO
CERO
DOS
SIETE
SEIS
.
"*/T0179.lab"
MARCAR
LUIS
GUEVARA
.
"*/T0180.lab"
TELEFONO
SIETE
CUATRO
SEIS
CINCO
TRES
TRES
.
"*/T0181.lab"
LLAMAR
LUIS
.
"*/T0182.lab"
LLAMAR
JORGE
LUIS
.
"*/T0183.lab"
MARCAR
JORGE
LUIS
.
"*/T0184.lab"
MARCAR
GUEVARA
.
"*/T0185.lab"
TELEFONO
CUATRO
DOS
TRES
.
"*/T0186.lab"
LLAMAR
GUEVARA
.
"*/T0187.lab"
MARCAR
JORGE
LUIS
.
"*/T0188.lab"
TELEFONO
SIETE
CERO
UNO
CINCO
SEIS
CERO
SIETE
TRES
SEIS
SEIS
UNO
UNO
CINCO
.
"*/T0189.lab"
LLAMAR
JORGE
LUIS
.
"*/T0190.lab"
MARCAR
LEISSI
LEON
.
"*/T0191.lab"
LLAMAR
GUEVARA
.
"*/T0192.lab"
LLAMAR
LUIS
.
"*/T0193.lab"
TELEFONO
DOS
TRES
SIETE
.
"*/T0194.lab"
LLAMAR
CARLOS
DIAZ
.
"*/T0195.lab"
TELEFONO
CERO
.
"*/T0196.lab"
MARCAR
GUEVARA
.
"*/T0197.lab"
MARCAR
CARLOS
DIAZ
.
"*/T0198.lab"
TELEFONO
CINCO
CERO
OCHO
SEIS
SEIS
CERO
DOS
UNO
CERO
CINCO
OCHO
CUATRO
TRES
SEIS
CINCO
SIETE
TRES
DOS
CINCO
CUATRO
CERO
.
"*/T0199.lab"
LLAMAR
LEON
.
"*/T0200.lab"
TELEFONO
SEIS
CUATRO
CERO
DOS
UNO
.
```

A.4.1. Archivos de Transcripción a Nivel de Fonema

Cada archivo de transcripción es convertido a nivel fonema para entrenar los HMMs

```
#!MLF!#

"T0001.lab"
sil
t
eh
l
eh
f
oh
n
oh
dh
oh
s
s
eh
y
s
k
w
ah
t
r
oh
t
r
eh
s
uh
n
oh
oh
ch
oh
dh
oh
s
th
ih
ng
k
oh
th
ih
ng
k
oh
k
w
ah
t
r
oh
uh
n
oh
s
eh
y
s
k
w
ah
t
r
oh
s
eh
y
s
th
eh
r
oh
th
ih
ng
k
oh
s
y
eh
t
eh
k
w
ah
t
r
oh
n
w
eh
b
eh
th
eh
r
oh
sil
.
"T0002.lab"
sil
ll
ah
m
ah
r
l
uh
y
s
g
eh
b
ah
r
ah
sil
.
"T0003.lab"
sil
t
eh
l
eh
f
oh
n
oh
s
eh
y
s
s
y
eh
t
eh
s
y
eh
t
eh
uh
n
oh
uh
n
oh
dh
oh
s
dh
oh
s
uh
n
oh
s
y
eh
t
eh
th
ih
ng
k
oh
k
w
ah
t
r
oh
oh
ch
oh
sil
.
"T0004.lab"
sil
ll
ah
m
ah
r
dh
ih
ah
s
sil
.
"T0005.lab"
sil
ll
ah
m
ah
r
g
eh
b
ah
r
ah
sil
.
"T0006.lab"
sil
m
ah
r
k
ah
r
l
eh
oh
n
sil
.
"T0007.lab"
sil
m
ah
r
k
ah
r
l
uh
y
s
g
eh
b
ah
r
ah
sil
.
"T0008.lab"
sil
t
eh
l
eh
f
oh
n
oh
oh
ch
oh
s
y
eh
t
eh
s
y
eh
t
eh
dh
oh
s
th
eh
r
oh
t
r
eh
s
oh
ch
oh
th
eh
r
oh
k
w
ah
t
r
oh
sil
.
"T0009.lab"
sil
t
eh
l
eh
f
oh
n
oh
uh
n
oh
sil
.
"T0010.lab"
sil
ll
ah
m
ah
r
l
eh
oh
n
sil
.
"T0011.lab"
sil
ll
ah
m
ah
r
l
uh
y
s
sil
.
"T0012.lab"
sil
t
eh
l
eh
f
oh
n
oh
s
eh
y
s
t
r
eh
s
sil
.
"T0013.lab"
sil
ll
ah
m
ah
r
l
eh
y
s
ih
l
eh
oh
n
sil
.
"T0014.lab"
sil
t
eh
l
eh
f
oh
n
oh
dh
oh
s
oh
ch
oh
n
w
eh
b
eh
s
y
eh
t
eh
k
w
ah
t
r
oh
th
ih
ng
k
oh
th
eh
r
oh
dh
oh
s
th
eh
r
oh
t
r
eh
s
th
ih
ng
k
oh
th
ih
ng
k
oh
s
eh
y
s
n
w
eh
b
eh
n
w
eh
b
eh
th
ih
ng
k
oh
oh
ch
oh
dh
oh
s
sil
.
"T0015.lab"
sil
m
ah
r
k
ah
r
j
oh
r
j
eh
l
uh
y
s
sil
.
"T0016.lab"
sil
t
eh
l
eh
f
oh
n
oh
th
eh
r
oh
th
ih
ng
k
oh
dh
oh
s
oh
ch
oh
th
ih
ng
k
oh
k
w
ah
t
r
oh
oh
ch
oh
k
w
ah
t
r
oh
oh
ch
oh
t
r
eh
s
th
ih
ng
k
oh
s
y
eh
t
eh
th
ih
ng
k
oh
s
eh
y
s
t
r
eh
s
uh
n
oh
s
y
eh
t
eh
th
ih
ng
k
oh
th
ih
ng
k
oh
th
ih
ng
k
oh
t
r
eh
s
th
eh
r
oh
dh
oh
s
sil
.
"T0017.lab"
sil
ll
ah
m
ah
r
dh
ih
ah
s
sil
.
"T0018.lab"
sil
m
ah
r
k
ah
r
l
uh
y
s
g
eh
b
ah
r
ah
sil
.
"T0019.lab"
sil
t
eh
l
eh
f
oh
n
oh
uh
n
oh
uh
n
oh
s
y
eh
t
eh
oh
ch
oh
dh
oh
s
t
r
eh
s
k
w
ah
t
r
oh
th
eh
r
oh
th
eh
r
oh
dh
oh
s
s
eh
y
s
th
ih
ng
k
oh
k
w
ah
t
r
oh
s
y
eh
t
eh
s
eh
y
s
sil
.
"T0020.lab"
sil
ll
ah
m
ah
r
g
eh
b
ah
r
ah
sil
.
"T0021.lab"
sil
ll
ah
m
ah
r
l
eh
y
s
ih
l
eh
oh
n
sil
.
"T0022.lab"
sil
ll
ah
m
ah
r
l
eh
y
s
ih
l
eh
oh
n
sil
.
"T0023.lab"
sil
ll
ah
m
ah
r
l
eh
y
s
ih
l
eh
oh
n
sil
.
"T0024.lab"
sil
ll
ah
m
ah
r
j
oh
r
j
eh
l
uh
y
s
sil
.
"T0025.lab"
sil
ll
ah
m
ah
r
l
eh
y
s
ih
l
eh
oh
n
sil
.
"T0026.lab"
sil
t
eh
l
eh
f
oh
n
oh
s
eh
y
s
t
r
eh
s
oh
ch
oh
oh
ch
oh
ch
oh
th
ih
ng
k
oh
oh
ch
oh
oh
ch
oh
th
ih
ng
k
oh
uh
n
oh
sil
.
"T0027.lab"
sil
t
eh
l
eh
f
oh
n
oh
uh
n
oh
th
ih
ng
k
oh
n
w
eh
b
eh
dh
oh
s
th
ih
ng
k
oh
k
w
ah
t
r
oh
sil
.
"T0028.lab"
sil
t
eh
l
eh
f
oh
n
oh
dh
oh
s
k
w
ah
t
r
oh
th
eh
r
oh
s
eh
y
s
th
ih
ng
k
oh
s
y
eh
t
eh
n
w
eh
b
eh
n
w
eh
b
eh
s
eh
y
s
k
w
ah
t
r
oh
s
eh
y
s
```

```
sil
.
"T0029.lab"
sil
ll
ah
m
ah
r
l
uh
y
s
sil
.
"T0030.lab"
sil
t
eh
l
eh
f
oh
n
oh
s
y
eh
t
eh
sil
.
"T0031.lab"
sil
ll
ah
m
ah
r
l
uh
y
s
sil
.
"T0032.lab"
sil
ll
ah
m
ah
r
dh
ih
ah
s
sil
.
"T0033.lab"
sil
ll
ah
m
ah
r
j
oh
r
j
eh
l
uh
y
s
sil
.
"T0034.lab"
sil
ll
ah
m
ah
r
dh
ih
ah
s
sil
.
"T0035.lab"
sil
m
ah
r
k
ah
r
l
eh
oh
n
sil
.
"T0036.lab"
sil
m
ah
r
k
ah
r
l
uh
y
s
sil
.
"T0037.lab"
sil
m
ah
r
k
ah
r
l
eh
y
s
ih
l
eh
oh
n
sil
.
"T0038.lab"
sil
m
ah
r
k
ah
r
j
oh
r
j
eh
l
uh
y
s
sil
.
"T0039.lab"
sil
ll
ah
m
ah
r
j
oh
r
j
eh
l
uh
y
s
sil
.
"T0040.lab"
sil
m
ah
r
k
ah
r
l
eh
y
s
ih
l
eh
oh
n
sil
.
"T0041.lab"
sil
ll
ah
m
ah
r
dh
ih
ah
s
sil
.
"T0042.lab"
sil
m
ah
r
k
ah
r
l
uh
y
s
sil
.
"T0043.lab"
sil
m
ah
r
k
ah
r
j
oh
r
j
eh
l
uh
y
s
sil
.
"T0044.lab"
sil
ll
ah
m
ah
r
l
uh
y
s
g
eh
b
ah
r
ah
sil
.
"T0045.lab"
sil
ll
ah
m
ah
r
l
uh
y
s
g
eh
b
ah
r
ah
sil
.
"T0046.lab"
sil
ll
ah
m
ah
r
l
eh
oh
n
sil
.
"T0047.lab"
sil
ll
ah
m
ah
r
l
eh
y
s
ih
l
eh
oh
n
sil
.
"T0048.lab"
sil
ll
ah
m
ah
r
dh
ih
ah
s
sil
.
"T0049.lab"
sil
m
ah
r
k
ah
r
k
ah
r
l
oh
s
dh
ih
ah
s
sil
.
"T0050.lab"
sil
ll
ah
m
ah
r
g
eh
b
ah
r
ah
sil
.
"T0051.lab"
sil
t
eh
l
eh
f
oh
n
oh
s
eh
y
s
dh
oh
s
k
w
ah
t
r
oh
s
y
eh
t
eh
sil
.
"T0052.lab"
sil
ll
ah
m
ah
r
l
eh
oh
n
sil
.
"T0053.lab"
sil
t
eh
l
eh
f
oh
n
oh
uh
n
oh
uh
n
oh
th
ih
ng
k
oh
sil
.
"T0054.lab"
sil
m
ah
r
k
ah
r
dh
ih
ah
s
sil
.
"T0055.lab"
sil
m
ah
r
k
ah
r
l
eh
oh
n
sil
.
"T0056.lab"
sil
m
ah
r
k
ah
r
k
ah
r
l
oh
s
dh
ih
ah
s
sil
.
"T0057.lab"
sil
t
eh
l
eh
f
oh
n
oh
dh
oh
s
sil
.
"T0058.lab"
sil
t
eh
l
eh
f
oh
n
oh
th
eh
r
oh
dh
oh
s
s
y
eh
t
eh
uh
n
oh
k
w
ah
t
r
oh
n
w
eh
b
eh
n
w
eh
b
eh
th
eh
r
oh
t
r
eh
s
k
w
ah
t
r
oh
k
w
ah
t
r
oh
oh
ch
oh
n
w
eh
b
eh
oh
ch
oh
n
w
eh
b
eh
s
y
eh
t
eh
uh
n
oh
k
w
ah
t
r
oh
th
ih
ng
k
oh
k
w
ah
t
r
oh
dh
oh
s
s
eh
y
s
k
w
ah
t
r
oh
th
ih
ng
k
oh
s
eh
y
s
th
ih
ng
k
oh
th
ih
ng
k
oh
t
r
eh
s
s
y
eh
t
eh
dh
oh
s
k
w
ah
t
r
oh
s
eh
y
s
th
ih
ng
k
oh
s
eh
y
s
th
eh
r
oh
dh
oh
s
th
eh
r
oh
sil
.
"T0059.lab"
sil
m
ah
r
k
ah
r
dh
ih
ah
s
sil
.
"T0060.lab"
sil
ll
ah
m
ah
r
dh
ih
ah
s
sil
.
"T0061.lab"
sil
t
eh
l
eh
f
oh
n
oh
n
w
eh
b
eh
s
eh
y
s
sil
.
"T0062.lab"
sil
ll
ah
m
ah
r
dh
ih
ah
s
sil
.
"T0063.lab"
sil
ll
ah
m
ah
r
l
uh
y
s
sil
.
"T0064.lab"
sil
t
eh
l
eh
f
oh
n
oh
s
eh
y
s
s
y
eh
t
eh
th
eh
r
oh
n
w
eh
b
eh
sil
.
"T0065.lab"
sil
t
eh
l
eh
f
oh
n
oh
th
eh
r
oh
sil
.
"T0066.lab"
sil
t
eh
l
eh
f
oh
n
oh
s
eh
y
s
n
w
eh
b
eh
uh
n
oh
n
w
eh
b
eh
th
ih
ng
k
oh
n
w
eh
b
eh
oh
ch
oh
dh
oh
s
t
r
eh
s
s
y
eh
t
eh
s
y
eh
t
eh
th
eh
r
oh
th
eh
r
oh
n
w
eh
b
eh
uh
n
oh
uh
n
oh
t
r
eh
s
dh
oh
s
k
w
ah
t
r
oh
dh
oh
s
uh
n
oh
th
ih
ng
k
oh
th
ih
ng
k
oh
k
w
ah
t
r
oh
s
eh
y
s
sil
.
"T0067.lab"
sil
t
eh
l
eh
f
oh
n
oh
s
y
eh
t
eh
t
r
eh
s
s
eh
y
s
s
eh
y
s
s
y
eh
t
eh
t
r
eh
s
n
w
eh
b
eh
s
eh
y
s
n
w
eh
b
eh
t
r
eh
s
dh
oh
s
k
w
ah
t
r
oh
dh
oh
s
sil
.
"T0068.lab"
sil
m
ah
r
k
ah
r
l
eh
oh
n
sil
.
"T0069.lab"
sil
t
eh
l
eh
f
oh
n
oh
s
eh
y
s
uh
n
oh
th
eh
r
oh
uh
n
oh
sil
.
"T0070.lab"
sil
t
```

```
eh
l
eh
f
oh
n
oh
dh
oh
s
sil
.
"T0071.lab"
sil
t
eh
l
eh
f
oh
n
oh
s
y
eh
t
eh
dh
oh
s
oh
ch
oh
s
eh
y
s
th
eh
r
oh
n
w
eh
b
eh
s
y
eh
t
eh
s
y
eh
t
eh
oh
ch
oh
uh
n
oh
uh
n
oh
oh
ch
oh
s
eh
y
s
t
r
eh
s
th
ih
ng
k
oh
s
y
eh
t
eh
oh
ch
oh
oh
ch
oh
dh
oh
s
oh
ch
oh
uh
n
oh
dh
oh
s
th
ih
ng
k
oh
oh
ch
oh
sil
.
"T0072.lab"
sil
t
eh
l
eh
f
oh
n
oh
uh
n
oh
uh
n
oh
th
eh
r
oh
th
eh
r
oh
th
ih
ng
k
oh
uh
n
oh
sil
.
"T0073.lab"
sil
ll
ah
m
ah
r
l
eh
y
s
ih
l
eh
oh
n
sil
.
"T0074.lab"
sil
t
eh
l
eh
f
oh
n
oh
k
w
ah
t
r
oh
th
ih
ng
k
oh
uh
n
oh
s
eh
y
s
oh
ch
oh
dh
oh
s
sil
.
"T0075.lab"
sil
t
eh
l
eh
f
oh
n
oh
s
eh
y
s
s
y
eh
t
eh
th
eh
r
oh
t
r
eh
s
th
ih
ng
k
oh
t
r
eh
s
t
r
eh
s
t
r
eh
s
sil
.
"T0076.lab"
sil
m
ah
r
k
ah
r
l
eh
y
s
ih
l
eh
oh
n
sil
.
"T0077.lab"
sil
t
eh
l
eh
f
oh
n
oh
t
r
eh
s
t
r
eh
s
th
ih
ng
k
oh
uh
n
oh
sil
.
"T0078.lab"
sil
t
eh
l
eh
f
oh
n
oh
dh
oh
s
dh
oh
s
oh
ch
oh
th
ih
ng
k
oh
dh
oh
s
s
eh
y
s
sil
.
"T0079.lab"
sil
ll
ah
m
ah
r
k
ah
r
l
oh
s
dh
ih
ah
s
sil
.
"T0080.lab"
sil
ll
ah
m
ah
r
g
eh
b
ah
r
ah
sil
.
"T0081.lab"
sil
t
eh
l
eh
f
oh
n
oh
s
eh
y
s
sil
.
"T0082.lab"
sil
t
eh
l
eh
f
oh
n
oh
oh
ch
oh
s
y
eh
t
eh
th
eh
r
oh
oh
ch
oh
sil
.
"T0083.lab"
sil
m
ah
r
k
ah
r
l
uh
y
s
sil
.
"T0084.lab"
sil
t
eh
l
eh
f
oh
n
oh
s
y
eh
t
eh
uh
n
oh
th
eh
r
oh
th
ih
ng
k
oh
uh
n
oh
t
r
eh
s
s
y
eh
t
eh
th
eh
r
oh
th
ih
ng
k
oh
th
eh
r
oh
n
w
eh
b
eh
n
w
eh
b
eh
t
r
eh
s
th
eh
r
oh
dh
oh
s
t
r
eh
s
sil
.
"T0085.lab"
sil
t
eh
l
eh
f
oh
n
oh
t
r
eh
s
oh
ch
oh
th
eh
r
oh
sil
.
"T0086.lab"
sil
t
eh
l
eh
f
oh
n
oh
oh
ch
oh
t
r
eh
s
dh
oh
s
n
w
eh
b
eh
s
y
eh
t
eh
th
ih
ng
k
oh
k
w
ah
t
r
oh
th
eh
r
oh
s
y
eh
t
eh
oh
ch
oh
oh
ch
oh
k
w
ah
t
r
oh
n
w
eh
b
eh
uh
n
oh
dh
oh
s
sil
.
"T0087.lab"
sil
ll
ah
m
ah
r
dh
ih
ah
s
sil
.
"T0088.lab"
sil
ll
ah
m
ah
r
l
eh
y
s
ih
l
eh
oh
n
sil
.
"T0089.lab"
sil
ll
ah
m
ah
r
k
ah
r
l
oh
s
dh
ih
ah
s
sil
.
"T0090.lab"
sil
ll
ah
m
ah
r
l
uh
y
s
sil
.
"T0091.lab"
sil
ll
ah
m
ah
r
k
ah
r
l
oh
s
dh
ih
ah
s
sil
.
"T0092.lab"
sil
t
eh
l
eh
f
oh
n
oh
dh
oh
s
uh
n
oh
th
ih
ng
k
oh
dh
oh
s
t
r
eh
s
th
eh
r
oh
sil
.
"T0093.lab"
sil
t
eh
l
eh
f
oh
n
oh
oh
ch
oh
dh
oh
s
dh
oh
s
sil
.
"T0094.lab"
sil
m
ah
r
k
ah
r
l
eh
y
s
ih
l
eh
oh
n
sil
.
"T0095.lab"
sil
m
ah
r
k
ah
r
l
uh
y
s
g
eh
b
ah
r
ah
sil
.
"T0096.lab"
sil
t
eh
l
eh
f
oh
n
oh
uh
n
oh
sil
.
"T0097.lab"
sil
ll
ah
m
ah
r
k
ah
r
l
oh
s
dh
ih
ah
s
sil
.
"T0098.lab"
sil
t
eh
l
eh
f
oh
n
oh
t
r
eh
s
k
w
ah
t
r
oh
dh
oh
s
oh
ch
oh
th
ih
ng
k
oh
k
w
ah
t
r
oh
uh
n
oh
k
w
ah
t
r
oh
th
ih
ng
k
oh
dh
oh
s
th
eh
r
oh
s
y
eh
t
eh
th
eh
r
oh
t
r
eh
s
t
r
eh
s
s
y
eh
t
eh
th
eh
r
oh
th
ih
ng
k
oh
t
r
eh
s
s
y
eh
t
eh
th
ih
ng
k
oh
th
ih
ng
k
oh
s
y
eh
t
eh
th
eh
r
oh
uh
n
oh
s
eh
y
s
s
y
eh
t
eh
oh
ch
oh
dh
oh
s
uh
n
oh
uh
n
oh
oh
ch
oh
uh
n
oh
n
w
eh
b
eh
n
w
eh
b
eh
n
w
eh
b
eh
uh
n
oh
dh
oh
s
dh
oh
s
k
w
ah
t
r
oh
s
y
eh
t
eh
s
y
eh
t
eh
s
y
eh
t
eh
k
w
ah
t
r
oh
t
r
eh
s
th
eh
```

```
r
oh
k
w
ah
t
r
oh
dh
oh
s
n
w
eh
b
eh
t
r
eh
s
th
ih
ng
k
oh
t
r
eh
s
sil
.
"T0099.lab"
sil
m
ah
r
k
ah
r
l
eh
y
s
ih
l
eh
oh
n
sil
.
"T0100.lab"
sil
ll
ah
m
ah
r
l
uh
y
s
sil
.
"T0101.lab"
sil
ll
ah
m
ah
r
l
uh
y
s
sil
.
"T0102.lab"
sil
ll
ah
m
ah
r
dh
ih
ah
s
sil
.
"T0103.lab"
sil
t
eh
l
eh
f
oh
n
oh
uh
n
oh
sil
.
"T0104.lab"
sil
ll
ah
m
ah
r
g
eh
b
ah
r
ah
sil
.
"T0105.lab"
sil
m
ah
r
k
ah
r
l
oh
n
oh
dh
ih
ah
s
sil
.
"T0106.lab"
sil
t
eh
l
eh
f
oh
n
oh
dh
oh
k
w
eh
s
eh
r
oh
th
ih
s
t
r
oh
eh
s
dh
oh
s
eh
y
s
t
r
oh
n
w
s
uh
n
oh
s
n
eh
b
eh
t
r
eh
s
n
w
eh
b
eh
t
eh
oh
ch
oh
t
th
eh
r
oh
s
"T0107.lab"
sil
ll
ah
m
ah
r
j
oh
r
j
eh
l
uh
y
s
sil
.
"T0108.lab"
sil
ll
ah
m
ah
r
k
ah
r
l
oh
n
oh
dh
ih
ah
s
sil
.
"T0109.lab"
sil
t
eh
l
eh
f
oh
n
oh
t
eh
l
eh
f
oh
n
oh
s
k
w
ah
t
r
oh
sil
.
"T0110.lab"
sil
t
eh
l
eh
f
oh
n
oh
uh
n
oh
th
ih
ng
k
oh
s
eh
y
s
n
w
eh
b
eh
s
y
ah
t
r
eh
s
y
eh
t
eh
s
y
eh
t
eh
l
eh
ch
oh
k
w
ah
t
r
ah
m
ah
r
k
ah
r
l
eh
y
s
sil
.
"T0111.lab"
sil
t
eh
l
eh
f
oh
n
oh
th
ih
ng
k
oh
th
ih
ng
k
oh
th
ih
ng
k
oh
th
ih
ng
k
oh
r
j
eh
y
s
eh
t
eh
s
y
eh
t
eh
l
eh
ch
oh
k
w
ah
t
r
ah
m
ah
r
k
ah
r
l
eh
y
s
sil
.
"T0112.lab"
sil
ll
ah
m
ah
r
k
ah
r
l
eh
oh
ch
oh
t
th
eh
r
oh
s
eh
b
eh
s
ih
ah
s
sil
.
"T0113.lab"
sil
t
eh
l
eh
f
oh
n
oh
th
ih
ng
k
oh
th
ih
ng
k
oh
r
j
eh
y
s
eh
t
eh
s
y
eh
t
eh
l
eh
ch
oh
k
w
ah
t
r
ah
m
ah
r
k
ah
r
l
eh
y
s
sil
.
"T0114.lab"
sil
m
ah
r
k
ah
r
j
oh
r
j
eh
l
uh
y
s
sil
.
"T0115.lab"
sil
t
eh
l
eh
f
oh
n
oh
s
y
eh
t
eh
l
eh
ch
oh
k
w
ah
t
r
ah
m
ah
r
k
ah
r
l
eh
y
s
sil
.
"T0116.lab"
sil
ll
ah
m
ah
r
l
uh
y
s
g
eh
b
ah
r
ah
sil
.
"T0117.lab"
sil
ll
ah
m
ah
r
j
oh
r
j
eh
l
uh
y
s
sil
.
"T0118.lab"
sil
t
eh
l
eh
f
oh
n
oh
th
eh
r
oh
n
w
eh
b
eh
oh
ch
oh
uh
n
oh
th
ih
ng
k
oh
s
ah
r
k
ah
r
j
eh
l
eh
y
s
ih
l
eh
oh
n
"T0119.lab"
sil
m
ah
r
k
ah
r
l
eh
y
s
ih
l
eh
oh
n
sil
.
"T0120.lab"
sil
t
eh
l
eh
f
oh
n
oh
dh
oh
s
n
w
eh
b
eh
s
y
eh
t
eh
s
y
eh
t
eh
l
eh
ch
oh
k
w
ah
t
r
ah
m
ah
r
k
ah
r
l
eh
y
s
sil
.
"T0121.lab"
sil
m
ah
r
k
ah
r
g
eh
b
ah
dh
oh
s
n
w
eh
b
eh
oh
ch
oh
uh
n
oh
th
ih
ng
k
oh
s
ah
r
k
ah
r
j
eh
l
eh
y
s
ih
l
eh
oh
n
"T0122.lab"
sil
m
ah
r
k
ah
r
l
uh
y
s
ih
ah
s
sil
.
"T0123.lab"
sil
t
eh
l
eh
f
oh
n
oh
uh
n
oh
th
ih
ng
k
oh
th
ih
ng
k
oh
r
j
eh
l
uh
y
s
sil
.
"T0124.lab"
sil
ll
ah
m
ah
r
l
uh
y
s
sil
.
"T0125.lab"
sil
sil
m
ah
r
k
ah
r
l
uh
y
s
sil
.
"T0126.lab"
sil
t
eh
l
eh
f
oh
n
oh
dh
oh
s
n
w
eh
b
eh
s
y
eh
t
eh
s
y
eh
t
eh
l
eh
ch
oh
k
w
ah
t
r
ah
m
ah
r
k
ah
r
l
eh
y
s
sil
.
"T0127.lab"
sil
m
ah
r
k
ah
r
l
uh
y
s
sil
.
"T0128.lab"
sil
ll
ah
m
ah
r
j
oh
r
j
eh
l
uh
y
s
sil
.
"T0129.lab"
sil
t
eh
l
eh
f
oh
n
oh
th
eh
r
oh
n
y
eh
t
eh
s
y
eh
t
eh
l
eh
ch
oh
k
w
ah
t
r
ah
m
ah
r
k
ah
r
l
eh
y
s
sil
.
"T0130.lab"
sil
t
eh
l
eh
b
oh
dh
oh
s
n
w
eh
b
eh
oh
ch
oh
uh
n
oh
th
ih
ng
k
oh
s
ah
r
k
ah
r
j
eh
l
eh
y
s
sil
.
"T0131.lab"
sil
m
ah
r
k
ah
r
l
uh
y
s
ih
ah
s
sil
.
"T0132.lab"
sil
m
ah
r
k
ah
r
l
uh
y
s
ih
ah
s
sil
.
"T0133.lab"
sil
m
ah
r
k
ah
r
l
eh
y
s
ih
l
eh
oh
n
sil
.
"T0134.lab"
sil
m
ah
r
k
ah
r
l
eh
y
s
sil
.
"T0135.lab"
sil
m
ah
r
k
ah
r
l
uh
y
s
g
eh
b
```

```
ah
r
ah
sil
.
"T0136.lab"
sil
ll
ah
m
ah
r
l
eh
y
s
ih
l
eh
oh
n
sil
.
"T0137.lab"
sil
t
eh
l
eh
f
oh
n
oh
t
r
eh
s
k
w
ah
t
r
oh
uh
n
oh
uh
n
oh
dh
oh
s
uh
n
oh
oh
ch
oh
s
eh
y
s
s
y
eh
t
eh
th
eh
r
oh
uh
n
oh
t
r
eh
s
oh
ch
oh
dh
oh
s
s
y
eh
t
eh
s
eh
y
s
t
r
eh
s
dh
oh
s
n
w
eh
b
eh
th
eh
r
oh
th
ih
ng
k
oh
sil
.
"T0138.lab"
sil
ll
ah
m
ah
r
dh
ih
ah
s
sil
.
"T0139.lab"
sil
t
eh
l
eh
f
oh
n
oh
th
eh
r
oh
th
ih
ng
k
oh
oh
ch
oh
t
r
eh
s
sil
.
"T0140.lab"
sil
ll
ah
m
ah
r
j
oh
r
j
eh
l
uh
y
s
sil
.
"T0141.lab"
sil
t
eh
l
eh
f
oh
n
oh
th
eh
r
oh
oh
ch
oh
th
ih
ng
k
oh
n
w
eh
b
eh
th
ih
ng
k
oh
th
ih
ng
k
oh
dh
oh
s
t
r
eh
s
dh
oh
s
uh
n
oh
th
ih
ng
k
oh
t
r
eh
s
dh
oh
s
s
y
eh
t
eh
th
ih
ng
k
oh
uh
n
oh
s
eh
y
s
dh
oh
s
k
w
ah
t
r
oh
th
ih
ng
k
oh
n
w
eh
b
eh
n
w
eh
b
eh
s
eh
y
s
s
y
eh
t
eh
th
ih
ng
k
oh
t
r
eh
s
n
w
eh
b
eh
oh
ch
oh
oh
ch
oh
sil
.
"T0142.lab"
sil
m
ah
r
k
ah
r
j
oh
r
j
eh
l
uh
y
s
sil
.
"T0143.lab"
sil
m
ah
r
k
ah
r
l
uh
y
s
sil
.
"T0144.lab"
sil
ll
ah
m
ah
r
g
eh
b
ah
r
ah
sil
.
"T0145.lab"
sil
ll
ah
m
ah
r
dh
ih
ah
s
sil
.
"T0146.lab"
sil
m
ah
r
k
ah
r
dh
ih
ah
s
sil
.
"T0147.lab"
sil
m
ah
r
k
ah
r
g
eh
b
ah
r
ah
sil
.
"T0148.lab"
sil
m
ah
r
k
ah
r
l
uh
y
s
g
eh
b
ah
r
ah
sil
.
"T0149.lab"
sil
t
eh
l
eh
f
oh
n
oh
s
y
eh
t
eh
th
ih
ng
k
oh
sil
.
"T0150.lab"
sil
ll
ah
m
ah
r
l
uh
y
s
g
eh
b
ah
r
ah
sil
.
"T0151.lab"
sil
ll
ah
m
ah
r
l
eh
oh
n
sil
.
"T0152.lab"
sil
ll
ah
m
ah
r
l
uh
y
s
g
eh
b
ah
r
ah
sil
.
"T0153.lab"
sil
t
eh
l
eh
f
oh
n
oh
s
y
eh
t
eh
s
eh
y
s
th
eh
r
oh
th
eh
r
oh
s
eh
y
s
s
y
eh
t
eh
oh
ch
oh
s
eh
y
s
k
w
ah
t
r
oh
t
r
eh
s
s
eh
y
s
s
y
eh
t
eh
k
w
ah
t
r
oh
k
w
ah
t
r
oh
s
y
eh
t
eh
s
y
eh
t
eh
n
w
eh
b
eh
th
eh
r
oh
dh
oh
s
oh
ch
oh
uh
n
oh
th
eh
r
oh
t
r
eh
s
th
ih
ng
k
oh
oh
ch
oh
dh
oh
s
k
w
ah
t
r
oh
th
eh
r
oh
oh
ch
oh
t
r
eh
s
k
w
ah
t
r
oh
n
w
eh
b
eh
dh
oh
s
s
y
eh
t
eh
th
eh
r
oh
s
y
eh
t
eh
k
w
ah
t
r
oh
n
w
eh
b
eh
th
eh
r
oh
s
eh
y
s
oh
ch
oh
sil
.
"T0154.lab"
sil
ll
ah
m
ah
r
l
eh
oh
n
sil
.
"T0155.lab"
sil
ll
ah
m
ah
r
dh
ih
ah
s
sil
.
"T0156.lab"
sil
ll
ah
m
ah
r
k
ah
r
l
oh
s
dh
ih
ah
s
sil
.
"T0157.lab"
sil
t
eh
l
eh
f
oh
n
oh
uh
n
oh
n
w
eh
b
eh
th
eh
r
oh
t
r
eh
s
n
w
eh
b
eh
uh
n
oh
t
r
eh
s
n
w
eh
b
eh
oh
ch
oh
oh
ch
oh
sil
.
"T0158.lab"
sil
ll
ah
m
ah
r
j
oh
r
j
eh
l
uh
y
s
sil
.
"T0159.lab"
sil
t
eh
l
eh
f
oh
n
oh
n
w
eh
b
eh
t
r
eh
s
s
eh
y
s
oh
ch
oh
n
w
eh
b
eh
th
ih
ng
k
oh
n
w
eh
b
eh
n
w
eh
b
eh
k
w
ah
t
r
oh
uh
n
oh
s
y
eh
t
eh
s
y
eh
t
eh
s
eh
y
s
uh
n
oh
uh
n
oh
oh
ch
oh
s
y
eh
t
eh
oh
ch
oh
k
w
ah
t
r
oh
s
y
eh
t
eh
s
eh
y
s
dh
oh
s
th
eh
r
oh
th
ih
ng
k
oh
sil
.
"T0160.lab"
sil
t
eh
l
eh
f
oh
n
oh
s
y
eh
t
eh
uh
n
oh
sil
.
"T0161.lab"
sil
t
eh
l
eh
f
oh
n
oh
dh
oh
s
th
eh
r
oh
s
eh
y
s
uh
n
oh
s
y
eh
t
eh
dh
oh
s
th
ih
ng
k
oh
sil
.
"T0162.lab"
sil
ll
ah
m
ah
r
l
uh
y
s
g
eh
b
ah
r
ah
sil
.
"T0163.lab"
sil
t
eh
l
eh
f
oh
n
oh
th
ih
ng
k
oh
th
ih
ng
k
oh
```

```
k
w
ah
t
r
oh
k
w
ah
t
r
oh
s
y
eh
t
eh
k
w
ah
t
r
oh
n
w
eh
b
eh
oh
ch
oh
th
ih
ng
k
oh
n
w
eh
b
eh
k
w
ah
t
r
oh
n
w
eh
b
eh
sil
.
"T0164.lab"
sil
t
eh
l
eh
f
oh
n
oh
th
eh
r
oh
oh
ch
oh
t
r
eh
s
k
w
ah
t
r
oh
uh
n
oh
uh
n
oh
t
r
eh
s
s
y
eh
t
eh
sil
.
"T0165.lab"
sil
ll
ah
m
ah
r
j
oh
r
j
eh
l
uh
y
s
sil
.
"T0166.lab"
sil
ll
ah
m
ah
r
g
eh
b
ah
r
ah
sil
.
"T0167.lab"
sil
m
ah
r
k
ah
r
l
uh
y
s
g
eh
b
ah
r
ah
sil
.
"T0168.lab"
sil
t
eh
l
eh
f
oh
n
oh
s
y
eh
t
eh
k
w
ah
t
r
oh
s
eh
y
s
s
y
eh
t
eh
s
eh
y
s
n
w
eh
b
eh
sil
.
"T0169.lab"
sil
m
ah
r
k
ah
r
dh
ih
ah
s
sil
.
"T0170.lab"
sil
t
eh
l
eh
f
oh
n
oh
uh
n
oh
sil
.
"T0171.lab"
sil
t
eh
l
eh
f
oh
n
oh
s
y
eh
t
eh
uh
n
oh
t
r
eh
s
s
y
eh
t
eh
k
w
ah
t
r
oh
n
w
eh
b
eh
th
ih
ng
k
oh
dh
oh
s
th
ih
ng
k
oh
uh
n
oh
k
w
ah
t
r
oh
s
y
eh
t
eh
sil
.
"T0172.lab"
sil
ll
ah
m
ah
r
l
uh
y
s
sil
.
"T0173.lab"
sil
ll
ah
m
ah
r
j
oh
r
j
eh
l
uh
y
s
sil
.
"T0174.lab"
sil
m
ah
r
k
ah
r
dh
ih
ah
s
sil
.
"T0175.lab"
sil
t
eh
l
eh
f
oh
n
oh
th
eh
r
oh
s
eh
y
s
dh
oh
s
oh
ch
oh
s
eh
y
s
th
eh
r
oh
k
w
ah
t
r
oh
s
eh
y
s
th
ih
ng
k
oh
k
w
ah
t
r
oh
dh
oh
s
th
ih
ng
k
oh
s
y
eh
t
eh
uh
n
oh
s
eh
y
s
t
r
eh
s
s
y
eh
t
eh
sil
.
"T0176.lab"
sil
t
eh
l
eh
f
oh
n
oh
s
y
eh
t
eh
s
y
eh
t
eh
t
r
eh
s
th
eh
r
oh
th
ih
ng
k
oh
s
eh
y
s
dh
oh
s
uh
n
oh
n
w
eh
b
eh
s
y
eh
t
eh
t
r
eh
s
th
ih
ng
k
oh
k
w
ah
t
r
oh
th
eh
r
oh
th
ih
ng
k
oh
s
y
eh
t
eh
k
w
ah
t
r
oh
s
y
eh
t
eh
s
y
eh
t
eh
k
w
ah
t
r
oh
th
ih
ng
k
oh
uh
n
oh
oh
ch
oh
uh
n
oh
sil
.
"T0177.lab"
sil
m
ah
r
k
ah
r
l
eh
y
s
ih
l
eh
oh
n
sil
.
"T0178.lab"
sil
t
eh
l
eh
f
oh
n
oh
t
r
eh
s
s
y
eh
t
eh
th
ih
ng
k
oh
t
r
eh
s
s
eh
y
s
oh
ch
oh
th
eh
r
oh
dh
oh
s
s
y
eh
t
eh
s
eh
y
s
sil
.
"T0179.lab"
sil
m
ah
r
k
ah
r
l
uh
y
s
g
eh
b
ah
r
ah
sil
.
"T0180.lab"
sil
t
eh
l
eh
f
oh
n
oh
s
y
eh
t
eh
k
w
ah
t
r
oh
s
eh
y
s
th
ih
ng
k
oh
t
r
eh
s
t
r
eh
s
sil
.
"T0181.lab"
sil
ll
ah
m
ah
r
l
uh
y
s
sil
.
"T0182.lab"
sil
ll
ah
m
ah
r
j
oh
r
j
eh
l
uh
y
s
sil
.
"T0183.lab"
sil
m
ah
r
k
ah
r
j
oh
r
j
eh
l
uh
y
s
sil
.
"T0184.lab"
sil
m
ah
r
k
ah
r
g
eh
b
ah
r
ah
sil
.
"T0185.lab"
sil
t
eh
l
eh
f
oh
n
oh
k
w
ah
t
r
oh
dh
oh
s
t
r
eh
s
sil
.
"T0186.lab"
sil
ll
ah
m
ah
r
g
eh
b
ah
r
ah
sil
.
"T0187.lab"
sil
m
ah
r
k
ah
r
j
oh
r
j
eh
l
uh
y
s
sil
.
"T0188.lab"
sil
t
eh
l
eh
f
oh
n
oh
s
y
eh
t
eh
th
eh
r
oh
uh
n
oh
th
ih
ng
k
oh
s
eh
y
s
th
eh
r
oh
s
y
eh
t
eh
t
r
eh
s
s
eh
y
s
s
eh
y
s
uh
n
oh
uh
n
oh
uh
n
oh
th
ih
ng
k
oh
sil
.
"T0189.lab"
sil
ll
ah
m
ah
r
j
oh
r
j
eh
l
uh
y
s
sil
.
"T0190.lab"
sil
m
ah
r
k
ah
r
l
eh
y
s
ih
l
eh
oh
n
sil
.
"T0191.lab"
sil
ll
ah
m
ah
r
g
eh
b
ah
r
ah
sil
.
"T0192.lab"
sil
ll
ah
m
ah
r
l
uh
y
s
sil
.
"T0193.lab"
sil
t
eh
l
eh
f
oh
n
oh
dh
oh
s
t
r
eh
s
s
y
eh
t
eh
sil
.
"T0194.lab"
sil
ll
ah
m
ah
r
k
ah
r
l
oh
s
dh
ih
ah
s
sil
.
"T0195.lab"
sil
t
eh
l
eh
f
oh
n
oh
th
eh
r
oh
sil
.
"T0196.lab"
sil
m
ah
r
k
ah
r
g
eh
b
ah
r
ah
sil
.
"T0197.lab"
sil
m
ah
r
k
ah
r
```

```
k           l           ch          s           k           ng          ih          "T0199.lab" eh          oh
ah          eh          oh          uh          w           k           ng          sil         l           th
r           f           s           n           ah          oh          k           ll          eh          eh
l           oh          eh          oh          t           s           oh          ah          f           r
oh          n           y           th          r           y           k           m           oh          oh
s           oh          s           eh          oh          eh          w           ah          n           dh
dh          th          s           r           t           t           ah          r           oh          oh
ih          ih          eh          oh          r           eh          t           l           s           s
ah          ng          y           th          eh          t           r           eh          eh          uh
s           k           s           ih          s           r           oh          oh          y           n
sil         oh          th          ng          s           eh          th          n           s           oh
.           th          eh          k           eh          s           eh          sil         k           sil
"T0198.lab" eh          r           oh          y           dh          r           .           w           .
sil         r           oh          oh          s           oh          oh          "T0200.lab" ah
t           oh          dh          ch          th          s           sil         sil         t
eh          oh          oh          oh          ih          th          .           t           r
```

A.5. Definición del HMM prototipo

```
~o
<STREAMINFO> 1 39
<VECSIZE> 39<NULLD><MFCC_D_A_0><DIAGC>
~h "proto"
<BEGINHMM>
<NUMSTATES> 5
    <STATE> 2
        <MEAN> 39
            -1.359490e+01 2.970981e+00 ... 5.619141e-04
        <VARIANCE> 39
            4.940723e+01 1.735306e+01 ...7.193607e-02
        <GCONST> 1.027506e+02
    <STATE> 3
        <MEAN> 39
            -1.359490e+01 2.970981e+00 ... 5.619141e-04
        <VARIANCE> 39
            4.940723e+01 1.735306e+01 ...7.193607e-02
        <GCONST> 1.027506e+02
    <STATE> 4
        <MEAN> 39
            -1.359490e+01 2.970981e+00 ... 5.619141e-04
        <VARIANCE> 39
            4.940723e+01 1.735306e+01 ...7.193607e-02
        <GCONST> 1.027506e+02<TRANSP> 5
 0.000000e+00 1.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
 0.000000e+00 6.000000e-01 4.000000e-01 0.000000e+00 0.000000e+00
 0.000000e+00 0.000000e+00 6.000000e-01 4.000000e-01 0.000000e+00
 0.000000e+00 0.000000e+00 0.000000e+00 7.000000e-01 3.000000e-01
 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
<ENDHMM>
```

A.6. Trifonemas

Los trifonemas son construidos a partir de la información de los archivos de transcripción a nivel fonema, tomando en cuenta el contexto de un fonema. A continuación se muestra todos los trifonemas construidos.

```
#!MLF!#     uh-n+oh     s+eh        k+w         g-eh+b      t-eh        k+w         ah-r        ah-r        eh-t+eh
"*/T0001.lab" n-oh      s-eh+y      k-w+ah      eh-b+ah     sp          k-w+ah      sp          sp          t-eh
sil         sp          eh-y+s      w-ah+t      b-ah+r      uh+n        w-ah+t      g+eh        l+uh        sp
t+eh        oh+ch       y-s         ah-t+r      ah-r+ah     uh-n+oh     ah-t+r      g-eh+b      l-uh+y      s+y
t-eh+l      oh-ch+oh    sp          t-r+oh      r-ah        n-oh        t-r+oh      eh-b+ah     uh-y+s      s-y+eh
eh-l+eh     ch-oh       k+w         r-oh        sp          sp          r-oh        b-ah+r      y-s         y-eh+t
l-eh+f      sp          k-w+ah      sp          sil         uh+n        sp          ah-r+ah     sp          eh-t+eh
eh-f+oh     dh+oh       w-ah+t      n+w         .           uh-n+oh     oh+ch       r-ah        g+eh        t-eh
f-oh+n      dh-oh+s     ah-t+r      n-w+eh      "*/T0003.lab" n-oh      oh-ch+oh    sp          g-eh+b      sp
oh-n+oh     oh-s        t-r+oh      w-eh+b      sil         sp          ch-oh       sil         eh-b+ah     dh+oh
n-oh        sp          r-oh        eh-b+eh     t+eh        dh+oh       sp          .           b-ah+r      dh-oh+s
sp          th+ih       sp          b-eh        t-eh+l      dh-oh+s     sil         "*/T0006.lab" ah-r+ah   oh-s
dh+oh       th-ih+ng    s+eh        sp          eh-l+eh     oh-s        .           sil         r-ah        sp
dh-oh+s     ih-ng+k     s-eh+y      th+eh       l-eh+f      sp          "*/T0004.lab" m+ah        sp          th+eh
oh-s        ng-k+oh     eh-y+s      th-eh+r     eh-f+oh     dh+oh       sil         m-ah+r      sil         th-eh+r
sp          k-oh        y-s         eh-r+oh     f-oh+n      dh-oh+s     ll+ah       ah-r+k      .           eh-r+oh
s+eh        sp          sp          r-oh        oh-n+oh     oh-s        ll-ah+m     r-k+ah      "*/T0008.lab" r-oh
s-eh+y      th+ih       th+eh       sp          n-oh        sp          ah-m+ah     k-ah+r      sil         sp
eh-y+s      th-ih+ng    th-eh+r     sil         sp          uh+n        m-ah+r      ah-r        t+eh        t+r
y-s         ih-ng+k     eh-r+oh     .           s+eh        uh-n+oh     ah-r        sp          t-eh+l      t-r+eh
sp          ng-k+oh     r-oh        "*/T0002.lab" s-eh+y    n-oh        sp          l+eh        eh-l+eh     r-eh+s
k+w         k-oh        sp          sil         eh-y+s      sp          dh+ih       l-eh+oh     l-eh+f      eh-s
k-w+ah      sp          th+ih       ll+ah       y-s         s+y         dh-ih+ah    eh-oh+n     eh-f+oh     sp
w-ah+t      k+w         th-ih+ng    ll-ah+m     sp          s-y+eh      ih-ah+s     oh-n        f-oh+n      oh+ch
ah-t+r      k-w+ah      ih-ng+k     ah-m+ah     s+y         y-eh+t      ah-s        sp          oh-n+oh     oh-ch+oh
t-r+oh      w-ah+t      ng-k+oh     m-ah+r      s-y+eh      eh-t+eh     sp          sil         n-oh        ch-oh
r-oh        ah-t+r      k-oh        ah-r        y-eh+t      t-eh        sil         .           sp          sp
sp          t-r+oh      sp          sp          eh-t+eh     sp          .           "*/T0007.lab" oh+ch     th+eh
t+r         r-oh        s+y         l+uh        t-eh        th+ih       "*/T0005.lab" sil       oh-ch+oh    th-eh+r
t-r+eh      sp          s-y+eh      l-uh+y      sp          th-ih+ng    sil         m+ah        ch-oh       eh-r+oh
r-eh+s      uh+n        y-eh+t      uh-y+s      s+y         ih-ng+k     ll+ah       m-ah+r      sp          r-oh
eh-s        uh-n+oh     eh-t+eh     y-s         s-y+eh      ng-k+oh     ll-ah+m     ah-r+k      s+y         sp
sp          n-oh        t-eh        sp          y-eh+t      k-oh        ah-m+ah     r-k+ah      s-y+eh      k+w
uh+n        sp          sp          g+eh        eh-t+eh     sp          m-ah+r      k-ah+r      y-eh+t      k-w+ah
```

```
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
sil
.
"*/T0009.lab"
sil
t+eh
t-eh+l
eh-l+eh
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
uh+n
uh-n+oh
n-oh
sp
sil
.
"*/T0010.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
l+eh
l-eh+oh
eh-oh+n
oh-n
sp
sil
.
"*/T0011.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
l+uh
l-uh+y
uh-y+s
y-s
sp
sil
.
"*/T0012.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
sil
.
"*/T0013.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
l+eh
l-eh+y
eh-y+s
y-s+ih
s-ih
sp
l+eh
l-eh+oh
eh-oh+n
oh-n
sp
sil
.
"*/T0014.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
th+eh
th-eh+r
th-eh+r
eh-r+oh
r-oh
sp
dh+oh
dh-oh+s
oh-s
sp
th+eh
th-eh+r
r-oh
sp
oh+ch
oh-ch+oh
ch-oh
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
dh+oh
dh-oh+s
oh-s
sp
th+eh
th-eh+r
eh-r+oh
r-oh
sp
ih-ng+k
ng-k+oh
k-oh
sp
ah-r+k
r-k+ah
k-ah+r
ah-r
sp
j+oh
j-oh+r
oh-r+j
sp
uh+n
uh-n+oh
n-oh
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
sil
.
"*/T0015.lab"
sil
m+ah
m-ah+r
ah-r+k
y-s
sp
ah-r
t+r
t-r+eh
r-eh+s
eh-s
sp
uh+n
uh-n+oh
n-oh
sp
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
"*/T0016.lab"
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
th+eh
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
dh+oh
dh-oh+s
oh-s
sp
th+eh
th-eh+r
eh-r+oh
r-oh
sp
dh+oh
dh-oh+s
oh-s
sp
th+eh
th-eh+r
r-oh
sp
dh+oh
dh-oh+s
oh-s
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
"*/T0017.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
ng-k+oh
k-oh
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
dh+ih
dh-ih+ah
ih-ah+s
ah-s
sp
sil
.
"*/T0018.lab"
sil
m+ah
m-ah+r
ah-r+k
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
l+uh
l-uh+y
uh-y+s
.
"*/T0020.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
g+eh
g-eh+b
eh-b+ah
b-ah+r
ah-r+ah
r-ah
sp
g+eh
g-eh+b
eh-b+ah
b-ah+r
ah-r+ah
r-ah
sp
t-eh+l
eh-l+eh
l-eh+f
f-oh+n
oh-n+oh
n-oh
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
th+eh
th-eh+r
th-eh+r
eh-r+oh
r-oh
sp
dh+oh
dh-oh+s
oh-s
sp
th+eh
th-eh+r
r-oh
sp
dh+oh
dh-oh+s
oh-s
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
dh+oh
dh-oh+s
oh-s
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
l+eh
l-eh+oh
eh-oh+n
oh-n
sp
sil
.
"*/T0021.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
l+eh
l-eh+y
eh-y+s
y-s+ih
s-ih
sp
l+eh
l-eh+oh
eh-oh+n
oh-n
sp
sil
.
"*/T0022.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
l+eh
l-eh+y
eh-y+s
y-s+ih
s-ih
sp
l+eh
l-eh+oh
eh-oh+n
oh-n
sp
sil
.
"*/T0023.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
l+eh
l-eh+y
eh-y+s
y-s+ih
s-ih
sp
l+eh
l-eh+oh
eh-oh+n
oh-n
sp
sil
.
"*/T0024.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
j+oh
j-oh+r
oh-r+j
r-j+eh
j-eh
sp
l+uh
l-uh+y
uh-y+s
y-s
sp
sil
.
"*/T0025.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
l+eh
l-eh+y
eh-y+s
y-s+ih
s-ih
sp
l+eh
l-eh+oh
eh-oh+n
oh-n
sp
sil
.
"*/T0026.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
sil
.
"*/T0027.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
sil
.
"*/T0028.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
sil
.
"*/T0029.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
l+eh
l-eh+y
eh-y+s
y-s+ih
s-ih
sp
l+eh
l-eh+oh
eh-oh+n
oh-n
sp
sil
.
"*/T0030.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
l+uh
l-uh+y
uh-y+s
y-s
sp
sil
.
"*/T0031.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
l+uh
l-uh+y
uh-y+s
y-s
sp
sil
.
"*/T0032.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
dh+ih
dh-ih+ah
ih-ah+s
ah-s
sp
sil
.
"*/T0033.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
j+oh
j-oh+r
oh-r+j
r-j+eh
j-eh
sp
l+uh
l-uh+y
uh-y+s
y-s
sp
sil
.
"*/T0034.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
dh+ih
dh-ih+ah
ih-ah+s
ah-s
sp
sil
.
"*/T0035.lab"
sil
m+ah
m-ah+r
ah-r+k
sp
l+eh
l-eh+y
eh-y+s
y-s+ih
s-ih
sp
l+eh
l-eh+oh
eh-oh+n
oh-n
sp
sil
.
"*/T0036.lab"
sil
m+ah
m-ah+r
ah-r+k
r-k+ah
k-ah+r
ah-r
sp
l+uh
l-uh+y
uh-y+s
y-s
sp
sil
.
"*/T0037.lab"
sil
m+ah
m-ah+r
ah-r+k
r-k+ah
k-ah+r
ah-r
sp
l+eh
l-eh+y
eh-y+s
y-s+ih
s-ih
sp
l+eh
l-eh+oh
eh-oh+n
oh-n
sp
sil
.
"*/T0038.lab"
sil
m+ah
m-ah+r
ah-r+k
r-k+ah
k-ah+r
ah-r
sp
j+oh
j-oh+r
oh-r+j
r-j+eh
j-eh
sp
l+uh
l-uh+y
uh-y+s
y-s
sp
sil
.
"*/T0039.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
j+oh
j-oh+r
oh-r+j
r-j+eh
j-eh
sp
l+uh
l-uh+y
uh-y+s
y-s
sp
sil
.
"*/T0040.lab"
sil
m+ah
m-ah+r
ah-r+k
r-k+ah
k-ah+r
ah-r
sp
l+eh
l-eh+y
eh-y+s
y-s+ih
s-ih
sp
l+eh
l-eh+oh
eh-oh+n
oh-n
sp
sil
.
```

```
sil
.
"*/T0041.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
dh+ih
dh-ih+ah
ih-ah+s
ah-s
sp
sil
.
"*/T0042.lab"
sil
m+ah
m-ah+r
ah-r+k
r-k+ah
k-ah+r
ah-r
sp
l+uh
l-uh+y
uh-y+s
y-s
sp
sil
.
"*/T0043.lab"
sil
m+ah
m-ah+r
ah-r+k
r-k+ah
k-ah+r
ah-r
sp
j+oh
j-oh+r
oh-r+j
r-j+eh
j-eh
sp
l+uh
l-uh+y
uh-y+s
y-s
sp
sil
.
"*/T0044.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
l+uh
l-uh+y
uh-y+s
y-s
sp
g+eh
g-eh+b
eh-b+ah
b-ah+r
ah-r+ah
r-ah
sp
sil
.
"*/T0045.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
l+uh
l-uh+y
uh-y+s
y-s
sp
g+eh
g-eh+b
eh-b+ah
b-ah+r
ah-r+ah
r-ah
sp
sil
.
"*/T0046.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
l+eh
l-eh+oh
eh-oh+n
oh-n
sp
sil
.
"*/T0047.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
l+eh
l-eh+y
eh-y+s
y-s+ih
s-ih
sp
l+eh
l-eh+oh
eh-oh+n
oh-n
sp
sil
.
"*/T0048.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
dh+ih
dh-ih+ah
ih-ah+s
ah-s
sp
sil
.
"*/T0049.lab"
sil
m+ah
m-ah+r
ah-r+k
r-k+ah
k-ah+r
ah-r
sp
k+ah
k-ah+r
ah-r+l
r-l+oh
l-oh+s
oh-s
sp
dh+ih
dh-ih+ah
ih-ah+s
ah-s
sp
sil
.
"*/T0050.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
g+eh
g-eh+b
eh-b+ah
b-ah+r
ah-r+ah
r-ah
sp
sil
.
"*/T0051.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
dh+oh
dh-oh+s
oh-s
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
sil
.
"*/T0052.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
l+eh
l-eh+oh
eh-oh+n
oh-n
sp
sil
.
"*/T0053.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
uh+n
uh-n+oh
n-oh
sp
uh+n
uh-n+oh
n-oh
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
sil
.
"*/T0054.lab"
sil
m+ah
m-ah+r
ah-r+k
r-k+ah
k-ah+r
ah-r
sp
dh+ih
dh-ih+ah
ih-ah+s
ah-s
sp
sil
.
"*/T0055.lab"
sil
m+ah
m-ah+r
ah-r+k
r-k+ah
k-ah+r
ah-r
sp
l+eh
l-eh+oh
eh-oh+n
oh-n
sp
sil
.
"*/T0056.lab"
sil
m+ah
m-ah+r
ah-r+k
r-k+ah
k-ah+r
ah-r
sp
k+ah
k-ah+r
ah-r+l
r-l+oh
l-oh+s
oh-s
sp
dh+ih
dh-ih+ah
ih-ah+s
ah-s
sp
sil
.
"*/T0057.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
dh+oh
dh-oh+s
oh-s
sp
sil
.
"*/T0058.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
th+eh
th-eh+r
eh-r+oh
r-oh
sp
dh+oh
dh-oh+s
oh-s
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
uh+n
uh-n+oh
n-oh
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
n+w
n-w+eh
w-eh+b
eh-b+eh
b-eh
sp
n+w
n-w+eh
w-eh+b
eh-b+eh
b-eh
sp
th+eh
th-eh+r
eh-r+oh
r-oh
sp
t+r
t-r+eh
r-eh+s
eh-s
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
oh+ch
oh-ch+oh
ch-oh
sp
n+w
n-w+eh
w-eh+b
eh-b+eh
b-eh
sp
oh+ch
oh-ch+oh
ch-oh
sp
n+w
n-w+eh
w-eh+b
eh-b+eh
b-eh
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
uh+n
uh-n+oh
n-oh
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
dh+oh
dh-oh+s
oh-s
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
t+r
t-r+eh
r-eh+s
eh-s
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
dh+oh
dh-oh+s
oh-s
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
th+eh
th-eh+r
eh-r+oh
r-oh
sp
dh+oh
dh-oh+s
oh-s
sp
th+eh
th-eh+r
eh-r+oh
r-oh
sp
sil
.
"*/T0059.lab"
sil
m+ah
m-ah+r
ah-r+k
r-k+ah
k-ah+r
ah-r
sp
dh+ih
dh-ih+ah
ih-ah+s
ah-s
sp
sil
.
"*/T0060.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
dh+ih
dh-ih+ah
ih-ah+s
ah-s
sp
sil
.
"*/T0061.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
n+w
n-w+eh
w-eh+b
eh-b+eh
b-eh
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
sil
.
"*/T0062.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
dh+ih
dh-ih+ah
ih-ah+s
ah-s
sp
sil
.
"*/T0063.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
l+uh
l-uh+y
uh-y+s
y-s
sp
sil
.
"*/T0064.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
th+eh
th-eh+r
eh-r+oh
r-oh
sp
n+w
n-w+eh
w-eh+b
eh-b+eh
b-eh
sp
sil
.
"*/T0065.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
th+eh
th-eh+r
eh-r+oh
r-oh
sp
sil
.
"*/T0066.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
n+w
n-w+eh
w-eh+b
eh-b+eh
b-eh
sp
uh+n
uh-n+oh
n-oh
sp
n+w
n-w+eh
w-eh+b
eh-b+eh
b-eh
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
n+w
n-w+eh
w-eh+b
eh-b+eh
b-eh
sp
oh+ch
oh-ch+oh
ch-oh
dh+oh
dh-oh+s
oh-s
sp
t+r
t-r+eh
r-eh+s
eh-s
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
th+eh
th-eh+r
eh-r+oh
r-oh
sp
th+eh
th-eh+r
eh-r+oh
r-oh
sp
n+w
n-w+eh
w-eh+b
eh-b+eh
b-eh
sp
uh+n
uh-n+oh
n-oh
sp
uh+n
uh-n+oh
n-oh
sp
t+r
t-r+eh
r-eh+s
eh-s
sp
dh+oh
dh-oh+s
oh-s
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
dh+oh
dh-oh+s
oh-s
sp
uh+n
uh-n+oh
n-oh
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
sil
.
"*/T0067.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
t+r
t-r+eh
r-eh+s
eh-s
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
t+r
t-r+eh
r-eh+s
eh-s
sp
n+w
n-w+eh
w-eh+b
eh-b+eh
b-eh
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
n+w
n-w+eh
w-eh+b
eh-b+eh
b-eh
sp
t+r
t-r+eh
r-eh+s
eh-s
sp
dh+oh
dh-oh+s
oh-s
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
dh+oh
dh-oh+s
oh-s
sp
sil
.
"*/T0068.lab"
sil
m+ah
m-ah+r
ah-r+k
r-k+ah
k-ah+r
ah-r
sp
l+eh
l-eh+oh
eh-oh+n
oh-n
sp
sil
.
"*/T0069.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
uh+n
uh-n+oh
n-oh
sp
th+eh
th-eh+r
eh-r+oh
r-oh
sp
uh+n
uh-n+oh
n-oh
sp
sil
.
"*/T0070.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
dh+oh
dh-oh+s
oh-s
sp
sil
.
"*/T0071.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
dh+oh
dh-oh+s
oh-s
sp
oh+ch
oh-ch+oh
ch-oh
sp
s+eh
s-eh+y
eh-y+s
```

```
y-s          t+eh         sil          th-ih+ng     s+eh         sp           n+w          sil          .            oh-s
sp           t-eh+l       .            ih-ng+k      s-eh+y       th+eh        n-w+eh       .            "*/T0093.lab" sp
th+eh        eh-l+eh      "*/T0075.lab" ng-k+oh     eh-y+s       th-eh+r      w-eh+b       "*/T0089.lab" sil         dh+ih
th-eh+r      l-eh+f       sil          k-oh         y-s          eh-r+oh      eh-b+eh      sil          t+eh         dh-ih+ah
eh-r+oh      eh-f+oh      t+eh         sp           sp           r-oh         b-eh         ll+ah        t-eh+l       ih-ah+s
r-oh         f-oh+n       t-eh+l       uh+n         sil          sp           sp           ll-ah+m      eh-l+eh      ah-s
sp           oh-n+oh      eh-l+eh      uh-n+oh      .            th+ih        s+y          ah-m+ah      l-eh+f       sp
n+w          n-oh         l-eh+f       n-oh         "*/T0082.lab" th-ih+ng    s-y+eh       m-ah+r       eh-f+oh      sil
n-w+eh       sp           eh-f+oh      sp           sil          ih-ng+k      y-eh+t       ah-r         f-oh+n       .
w-eh+b       uh+n         f-oh+n       sil          t+eh         ng-k+oh      eh-t+eh      sp           oh-n+oh      "*/T0098.lab"
eh-b+eh      uh-n+oh      oh-n+oh      .            t-eh+l       k-oh         t-eh         k+ah         n-oh         sil
b-eh         n-oh         n-oh         "*/T0078.lab" eh-l+eh     sp           sp           k-ah+r       sp           t+eh
sp           sp           sp           sil          l-eh+f       th+eh        th+ih        ah-r+l       oh+ch        t-eh+l
s+y          uh+n         s+eh         t+eh         eh-f+oh      th-eh+r      th-ih+ng     r-l+oh       oh-ch+oh     eh-l+eh
s-y+eh       uh-n+oh      s-eh+y       t-eh+l       f-oh+n       eh-r+oh      ih-ng+k      l-oh+s       ch-oh        l-eh+f
y-eh+t       n-oh         eh-y+s       eh-l+eh      oh-n+oh      r-oh         ng-k+oh      oh-s         sp           eh-f+oh
eh-t+eh      sp           y-s          l-eh+f       n-oh         sp           k-oh         sp           dh+oh        f-oh+n
t-eh         th+eh        sp           eh-f+oh      sp           n+w          sp           dh+ih        dh-oh+s      oh-n+oh
sp           th-eh+r      s+y          f-oh+n       oh+ch        n-w+eh       k+w          dh-ih+ah     oh-s         n-oh
s+y          eh-r+oh      s-y+eh       oh-n+oh      oh-ch+oh     w-eh+b       k-w+ah       ih-ah+s      sp           sp
s-y+eh       r-oh         y-eh+t       n-oh         ch-oh        eh-b+eh      w-ah+t       ah-s         dh+oh        t+r
y-eh+t       sp           eh-t+eh      sp           sp           b-eh         ah-t+r       sp           dh-oh+s      t-r+eh
eh-t+eh      th+eh        t-eh         dh+oh        s+y          sp           t-r+oh       sil          oh-s         r-eh+s
t-eh         th-eh+r      sp           dh-oh+s      s-y+eh       n+w          r-oh         .            sp           eh-s
sp           eh-r+oh      th+eh        oh-s         y-eh+t       n-w+eh       sp           "*/T0090.lab" sil         sp
oh+ch        r-oh         th-eh+r      sp           eh-t+eh      w-eh+b       th+eh        sil          .            k+w
oh-ch+oh     sp           eh-r+oh      dh+oh        t-eh         eh-b+eh      th-eh+r      ll+ah        "*/T0094.lab" k-w+ah
ch-oh        th+ih        r-oh         dh-oh+s      sp           b-eh         eh-r+oh      ll-ah+m      sil          w-ah+t
sp           th-ih+ng     sp           oh-s         th+eh        sp           r-oh         ah-m+ah      m+ah         ah-t+r
uh+n         ih-ng+k      t+r          sp           th-eh+r      t+r          sp           m-ah+r       m-ah+r       t-r+oh
uh-n+oh      ng-k+oh      t-r+eh       oh+ch        eh-r+oh      t-r+eh       s+y          ah-r         ah-r+k       r-oh
n-oh         k-oh         r-eh+s       oh-ch+oh     r-oh         r-eh+s       s-y+eh       sp           r-k+ah       sp
sp           sp           eh-s         ch-oh        sp           eh-s         y-eh+t       l+uh         k-ah+r       dh+oh
uh+n         uh+n         sp           sp           oh+ch        sp           eh-t+eh      l-uh+y       ah-r         dh-oh+s
uh-n+oh      uh-n+oh      th+ih        th+ih        oh-ch+oh     th+eh        t-eh         uh-y+s       sp           oh-s
n-oh         n-oh         th-ih+ng     th-ih+ng     ch-oh        th-eh+r      sp           y-s          l+eh         sp
sp           sp           ih-ng+k      ih-ng+k      sp           eh-r+oh      oh+ch        sp           l-eh+y       oh+ch
oh+ch        sil          ng-k+oh      ng-k+oh      sil          r-oh         oh-ch+oh     sil          eh-y+s       oh-ch+oh
oh-ch+oh     .            k-oh         k-oh         .            sp           ch-oh        .            y-s+ih       ch-oh
ch-oh        "*/T0073.lab" sp          sp           "*/T0083.lab" dh+oh       sp           "*/T0091.lab" s-ih        sp
sp           sil          t+r          dh+oh        sil          dh-oh+s      oh+ch        sil          sp           th+ih
s+eh         ll+ah        t-r+eh       dh-oh+s      m+ah         oh-s         oh-ch+oh     ll+ah        l+eh         th-ih+ng
s-eh+y       ll-ah+m      r-eh+s       oh-s         m-ah+r       sp           ch-oh        ll-ah+m      l-eh+oh      ih-ng+k
eh-y+s       ah-m+ah      eh-s         sp           ah-r+k       t+r          sp           ah-m+ah      eh-oh+n      ng-k+oh
y-s          m-ah+r       sp           s+eh         r-k+ah       t-r+eh       k+w          m-ah+r       oh-n         k-oh
sp           ah-r         t+r          s-eh+y       k-ah+r       r-eh+s       k-w+ah       ah-r         sp           sp
t+r          sp           t-r+eh       eh-y+s       ah-r         eh-s         w-ah+t       sp           sil          k+w
t-r+eh       l+eh         r-eh+s       y-s          sp           sp           ah-t+r       k+ah         .            k-w+ah
r-eh+s       l-eh+y       eh-s         sp           l+uh         sil          t-r+oh       k-ah+r       "*/T0095.lab" w-ah+t
eh-s         eh-y+s       sp           sil          l-uh+y       .            r-oh         ah-r+l       sil          ah-t+r
sp           y-s+ih       t+r          .            uh-y+s       "*/T0085.lab" sp          r-l+oh       m+ah         t-r+oh
th+ih        s-ih         t-r+eh       "*/T0079.lab" y-s         sil          n+w          l-oh+s       m-ah+r       r-oh
th-ih+ng     sp           r-eh+s       sil          sp           t+eh         n-w+eh       oh-s         ah-r+k       sp
ih-ng+k      l+eh         eh-s         ll+ah        sil          t-eh+l       w-eh+b       sp           r-k+ah       uh+n
ng-k+oh      l-eh+oh      sp           ll-ah+m      .            eh-l+eh      eh-b+eh      dh+ih        k-ah+r       uh-n+oh
k-oh         eh-oh+n      sil          ah-m+ah      "*/T0084.lab" l-eh+f      b-eh         dh-ih+ah     ah-r         n-oh
sp           oh-n         .            m-ah+r       sil          eh-f+oh      sp           ih-ah+s      sp           sp
s+y          sp           "*/T0076.lab" ah-r        t+eh         f-oh+n       uh+n         ah-s         l+uh         k+w
s-y+eh       sil          sil          sp           t-eh+l       oh-n+oh      uh-n+oh      sp           l-uh+y       k-w+ah
y-eh+t       .            m+ah         k+ah         eh-l+eh      n-oh         n-oh         sil          uh-y+s       w-ah+t
eh-t+eh      "*/T0074.lab" m-ah+r       k-ah+r       l-eh+f       sp           sp           .            y-s          ah-t+r
t-eh         sil          ah-r+k       ah-r+l       eh-f+oh      t+r          dh+oh        "*/T0092.lab" sp          t-r+oh
sp           t+eh         r-k+ah       r-l+oh       f-oh+n       t-r+eh       dh-oh+s      sil          g+eh         r-oh
oh+ch        t-eh+l       k-ah+r       l-oh+s       oh-n+oh      r-eh+s       oh-s         t+eh         g-eh+b       sp
oh-ch+oh     eh-l+eh      ah-r         oh-s         n-oh         eh-s         sp           t-eh+l       eh-b+ah      th+ih
ch-oh        l-eh+f       sp           sp           sp           sp           sil          eh-l+eh      b-ah+r       th-ih+ng
sp           eh-f+oh      l+eh         dh+ih        s+y          oh+ch        .            l-eh+f       ah-r+ah      ih-ng+k
oh+ch        f-oh+n       l-eh+y       dh-ih+ah     s-y+eh       oh-ch+oh     "*/T0087.lab" eh-f+oh     r-ah         ng-k+oh
oh-ch+oh     oh-n+oh      eh-y+s       ih-ah+s      y-eh+t       ch-oh        sil          f-oh+n       sp           k-oh
ch-oh        n-oh         y-s+ih       ah-s         eh-t+eh      sp           ll+ah        oh-n+oh      sil          sp
sp           sp           s-ih         sp           t-eh         th+eh        ll-ah+m      n-oh         .            dh+oh
dh+oh        k+w          sp           sil          sp           th-eh+r      ah-m+ah      sp           "*/T0096.lab" dh-oh+s
dh-oh+s      k-w+ah       l+eh         .            uh+n         eh-r+oh      m-ah+r       dh+oh        sil          oh-s
oh-s         w-ah+t       l-eh+oh      "*/T0080.lab" uh-n+oh     r-oh         ah-r         dh-oh+s      t+eh         sp
sp           ah-t+r       eh-oh+n      sil          n-oh         sp           sp           oh-s         t-eh+l       th+eh
oh+ch        t-r+oh       oh-n         ll+ah        sp           sil          dh+ih        sp           eh-l+eh      th-eh+r
oh-ch+oh     r-oh         sp           ll-ah+m      th+eh        .            dh-ih+ah     uh+n         l-eh+f       eh-r+oh
ch-oh        sp           sil          ah-m+ah      th-eh+r      "*/T0086.lab" ih-ah+s     uh-n+oh      eh-f+oh      r-oh
sp           th+ih        .            m-ah+r       eh-r+oh      sil          ah-s         n-oh         f-oh+n       sp
uh+n         th-ih+ng     "*/T0077.lab" ah-r        r-oh         t+eh         sp           sp           oh-n+oh      s+y
uh-n+oh      ih-ng+k      sil          sp           sp           t-eh+l       sil          th+ih        n-oh         s-y+eh
n-oh         ng-k+oh      t+eh         g+eh         th+ih        eh-l+eh      .            th-ih+ng     sp           y-eh+t
sp           k-oh         t-eh+l       g-eh+b       th-ih+ng     l-eh+f       "*/T0088.lab" ih-ng+k     uh+n         eh-t+eh
dh+oh        sp           eh-l+eh      eh-b+ah      ih-ng+k      eh-f+oh      sil          ng-k+oh      uh-n+oh      t-eh
dh-oh+s      uh+n         l-eh+f       b-ah+r       ng-k+oh      f-oh+n       ll+ah        k-oh         n-oh         sp
oh-s         uh-n+oh      eh-f+oh      ah-r+ah      k-oh         oh-n+oh      ll-ah+m      sp           sp           th+eh
sp           n-oh         f-oh+n       r-ah         sp           n-oh         ah-m+ah      dh+oh        sil          th-eh+r
th+ih        sp           oh-n+oh      sp           uh+n         sp           m-ah+r       dh-oh+s      .            eh-r+oh
th-ih+ng     s+eh         n-oh         sil          uh-n+oh      oh+ch        ah-r         oh-s         "*/T0097.lab" r-oh
ih-ng+k      s-eh+y       sp           .            n-oh         oh-ch+oh     sp           sp           sil          sp
ng-k+oh      eh-y+s       t+r          "*/T0081.lab" sp          ch-oh        l+eh         t+r          ll+ah        t+r
k-oh         y-s          t-r+eh       sil          t+r          sp           l-eh+y       t-r+eh       ll-ah+m      t-r+eh
sp           sp           r-eh+s       t+eh         t-r+eh       t+r          eh-y+s       r-eh+s       ah-m+ah      r-eh+s
oh+ch        oh+ch        eh-s         t-eh+l       r-eh+s       t-r+eh       y-s+ih       eh-s         m-ah+r       eh-s
oh-ch+oh     oh-ch+oh     sp           eh-l+eh      eh-s         r-eh+s       s-ih         sp           ah-r         sp
ch-oh        ch-oh        t+r          l-eh+f       sp           eh-s         sp           th+eh        sp           t+r
sp           sp           t-r+eh       eh-f+oh      s+y          sp           l+eh         th-eh+r      k+ah         t-r+eh
sil          dh+oh        r-eh+s       f-oh+n       s-y+eh       dh+oh        l-eh+oh      eh-r+oh      k-ah+r       r-eh+s
.            dh-oh+s      eh-s         oh-n+oh      y-eh+t       dh-oh+s      eh-oh+n      r-oh         ah-r+l       eh-s
"*/T0072.lab" oh-s        sp           n-oh         eh-t+eh      oh-s         oh-n         sp           r-l+oh       sp
sil          sp           th+ih        sp           t-eh         sp           sp           sil          l-oh+s       s+y
```

```
s-y+eh      n+w         k-ah+r      k-ah+r      .           oh-ch+oh    uh-n+oh     sp          l-uh+y      ng-k+oh
y-eh+t      n-w+eh      ah-r        ah-r        "*/T0107.lab" ch-oh     n-oh        oh+ch       uh-y+s      k-oh
eh-t+eh     w-eh+b      sp          sp          sil         sp          sp          oh-ch+oh    y-s         sp
t-eh        eh-b+eh     l+eh        k+ah        ll+ah       th+eh       th+ih       ch-oh       sp          sil
sp          b-eh        l-eh+y      k-ah+r      ll-ah+m     th-eh+r     th-ih+ng    sp          sil         .
th+eh       sp          eh-y+s      ah-r+l      ah-m+ah     eh-r+oh     ih-ng+k     uh+n        .           "*/T0119.lab"
th-eh+r     uh+n        y-s+ih      r-l+oh      m-ah+r      r-oh        ng-k+oh     uh-n+oh     "*/T0115.lab" sil
eh-r+oh     uh-n+oh     s-ih        l-oh+s      ah-r        sp          k-oh        n-oh        sil         m+ah
r-oh        n-oh        sp          oh-s        sp          dh+oh       sp          sp          t+eh        m-ah+r
sp          sp          l+eh        sp          j+oh        dh-oh+s     s+eh        th+ih       t-eh+l      ah-r+k
th+ih       dh+oh       l-eh+oh     dh+ih       j-oh+r      oh-s        s-eh+y      th-ih+ng    eh-l+eh     r-k+ah
th-ih+ng    dh-oh+s     eh-oh+n     dh-ih+ah    oh-r+j      sp          eh-y+s      ih-ng+k     l-eh+f      k-ah+r
ih-ng+k     oh-s        oh-n        ih-ah+s     r-j+eh      oh+ch       y-s         ng-k+oh     eh-f+oh     ah-r
ng-k+oh     sp          sp          ah-s        j-eh        oh-ch+oh    sp          k-oh        f-oh+n      sp
k-oh        dh+oh       sil         sp          sp          ch-oh       n+w         sp          oh-n+oh     l+eh
sp          dh-oh+s     .           sil         l+uh        sp          n-w+eh      th+ih       n-oh        l-eh+y
t+r         oh-s        "*/T0100.lab" .         l-uh+y      th+ih       w-eh+b      th-ih+ng    sp          eh-y+s
t-r+eh      sp          sil         "*/T0106.lab" uh-y+s    th-ih+ng    eh-b+eh     ih-ng+k     s+y         y-s+ih
r-eh+s      k+w         ll+ah       sil         y-s         ih-ng+k     b-eh        ng-k+oh     s-y+eh      s-ih
eh-s        k-w+ah      ll-ah+m     t+eh        sp          ng-k+oh     sp          k-oh        y-eh+t      sp
sp          w-ah+t      ah-m+ah     t-eh+l      sil         k-oh        s+y         sp          eh-t+eh     l+eh
s+y         ah-t+r      m-ah+r      eh-l+eh     .           sp          s-y+eh      sil         t-eh        l-eh+oh
s-y+eh      t-r+oh      ah-r        l-eh+f      "*/T0108.lab" dh+oh     y-eh+t      .           sp          eh-oh+n
y-eh+t      r-oh        sp          eh-f+oh     sil         dh-oh+s     eh-t+eh     "*/T0112.lab" sil       oh-n
eh-t+eh     sp          l+uh        f-oh+n      ll+ah       oh-s        t-eh        sil         .           sp
t-eh        s+y         l-uh+y      oh-n+oh     ll-ah+m     sp          sp          ll+ah       "*/T0116.lab" sil
sp          s-y+eh      uh-y+s      n-oh        ah-m+ah     th+eh       s+y         ll-ah+m     sil         .
th+ih       y-eh+t      y-s         sp          m-ah+r      th-eh+r     s-y+eh      ah-m+ah     ll+ah       "*/T0120.lab"
th-ih+ng    eh-t+eh     sp          dh+oh       ah-r        eh-r+oh     y-eh+t      m-ah+r      ll-ah+m     sil
ih-ng+k     t-eh        sil         dh-oh+s     sp          r-oh        eh-t+eh     ah-r        ah-m+ah     t+eh
ng-k+oh     sp          .           oh-s        k+ah        sp          t-eh        sp          m-ah+r      t-eh+l
k-oh        s+y         "*/T0101.lab" sp        k-ah+r      th+eh       sp          k+ah        ah-r        eh-l+eh
sp          s-y+eh      sil         k+w         ah-r+l      th-eh+r     oh+ch       k-ah+r      sp          l-eh+f
th+ih       y-eh+t      ll+ah       k-w+ah      r-l+oh      eh-r+oh     oh-ch+oh    ah-r+l      l+uh        eh-f+oh
th-ih+ng    eh-t+eh     ll-ah+m     w-ah+t      l-oh+s      r-oh        ch-oh       r-l+oh      l-uh+y      f-oh+n
ih-ng+k     t-eh        ah-m+ah     ah-t+r      oh-s        sp          sp          l-oh+s      uh-y+s      oh-n+oh
ng-k+oh     sp          m-ah+r      t-r+oh      sp          n+w         k+w         oh-s        y-s         n-oh
k-oh        s+y         ah-r        r-oh        dh+ih       n-w+eh      k-w+ah      sp          sp          sp
sp          s-y+eh      sp          sp          dh-ih+ah    w-eh+b      w-ah+t      dh+ih       g+eh        dh+oh
s+y         y-eh+t      l+uh        th+ih       ih-ah+s     eh-b+eh     ah-t+r      dh-ih+ah    g-eh+b      dh-oh+s
s-y+eh      eh-t+eh     l-uh+y      th-ih+ng    ah-s        b-eh        t-r+oh      ih-ah+s     eh-b+ah     oh-s
y-eh+t      t-eh        uh-y+s      ih-ng+k     sp          sp          r-oh        ah-s        b-ah+r      sp
eh-t+eh     sp          y-s         ng-k+oh     sil         k+w         sp          sp          ah-r+ah     n+w
t-eh        k+w         sp          k-oh        .           k-w+ah      k+w         sil         r-ah        n-w+eh
sp          k-w+ah      sil         sp          "*/T0109.lab" w-ah+t    k-w+ah      .           sp          w-eh+b
th+eh       w-ah+t      .           k+w         sil         ah-t+r      w-ah+t      "*/T0113.lab" sil       eh-b+eh
th-eh+r     ah-t+r      "*/T0102.lab" k-w+ah    t+eh        t-r+oh      ah-t+r      sil         .           b-eh
eh-r+oh     t-r+oh      sil         w-ah+t      t-eh+l      r-oh        t-r+oh      t+eh        "*/T0117.lab" sp
r-oh        r-oh        ll+ah       ah-t+r      eh-l+eh     sp          r-oh        t-eh+l      sil         s+y
sp          sp          ll-ah+m     t-r+oh      l-eh+f      s+y         sp          eh-l+eh     ll+ah       s-y+eh
uh+n        t+r         ah-m+ah     r-oh        eh-f+oh     s-y+eh      oh+ch       l-eh+f      ll-ah+m     y-eh+t
uh-n+oh     t-r+eh      m-ah+r      sp          f-oh+n      y-eh+t      oh-ch+oh    eh-f+oh     ah-m+ah     eh-t+eh
n-oh        r-eh+s      ah-r        th+eh       oh-n+oh     eh-t+eh     ch-oh       f-oh+n      m-ah+r      t-eh
sp          eh-s        sp          th-eh+r     n-oh        t-eh        sp          oh-n+oh     ah-r        sp
s+eh        sp          dh+ih       eh-r+oh     sp          sp          th+ih       n-oh        sp          th+ih
s-eh+y      th+eh       dh-ih+ah    r-oh        s+eh        s+y         th-ih+ng    sp          j+oh        th-ih+ng
eh-y+s      th-eh+r     ih-ah+s     sp          s-eh+y      s-y+eh      ih-ng+k     s+eh        j-oh+r      ih-ng+k
y-s         eh-r+oh     ah-s        n+w         eh-y+s      y-eh+t      ng-k+oh     s-eh+y      oh-r+j      ng-k+oh
sp          r-oh        sp          n-w+eh      y-s         eh-t+eh     k-oh        eh-y+s      r-j+eh      k-oh
s+y         sp          sil         w-eh+b      sp          t-eh        sp          y-s         j-eh        sp
s-y+eh      k+w         .           eh-b+eh     t+r         sp          sil         sp          sp          dh+oh
y-eh+t      k-w+ah      "*/T0103.lab" b-eh      t-r+eh      th+eh       .           s+eh        l+uh        dh-oh+s
eh-t+eh     w-ah+t      sil         sp          r-eh+s      th-eh+r     "*/T0111.lab" s-eh+y    l-uh+y      oh-s
t-eh        ah-t+r      t+eh        th+eh       eh-s        eh-r+oh     sil         eh-y+s      uh-y+s      sp
sp          t-r+oh      t-eh+l      th-eh+r     sp          r-oh        t+eh        y-s         y-s         oh+ch
oh+ch       r-oh        eh-l+eh     eh-r+oh     dh+oh       sp          t-eh+l      sp          sp          oh-ch+oh
oh-ch+oh    sp          l-eh+f      r-oh        dh-oh+s     uh+n        eh-l+eh     t+r         sil         ch-oh
ch-oh       dh+oh       eh-f+oh     sp          oh-s        uh-n+oh     l-eh+f      t-r+eh      .           sp
sp          dh-oh+s     f-oh+n      t+r         sp          n-oh        eh-f+oh     r-eh+s      "*/T0118.lab" th+eh
dh+oh       oh-s        oh-n+oh     t-r+eh      t+r         sp          f-oh+n      eh-s        sil         th-eh+r
dh-oh+s     sp          n-oh        r-eh+s      t-r+eh      s+eh        oh-n+oh     sp          t+eh        eh-r+oh
oh-s        n+w         sp          eh-s        r-eh+s      s-eh+y      n-oh        th+ih       t-eh+l      r-oh
sp          n-w+eh      uh+n        sp          eh-s        eh-y+s      sp          th-ih+ng    eh-l+eh     sp
uh+n        w-eh+b      uh-n+oh     t+r         sp          y-s         th+ih       ih-ng+k     l-eh+f      sil
uh-n+oh     eh-b+eh     n-oh        t-r+eh      s+eh        sp          th-ih+ng    ng-k+oh     eh-f+oh     .
n-oh        b-eh        sp          r-eh+s      s-eh+y      t+r         ih-ng+k     k-oh        f-oh+n      "*/T0121.lab"
sp          sp          sil         eh-s        eh-y+s      t-r+eh      ng-k+oh     sp          oh-n+oh     sil
uh+n        t+r         .           sp          y-s         r-eh+s      k-oh        th+ih       n-oh        m+ah
uh-n+oh     t-r+eh      "*/T0104.lab" n+w       sp          eh-s        sp          th-ih+ng    sp          m-ah+r
n-oh        r-eh+s      sil         n-w+eh      uh+n        sp          s+eh        ih-ng+k     th+eh       ah-r+k
sp          eh-s        ll+ah       w-eh+b      uh-n+oh     k+w         s-eh+y      ng-k+oh     th-eh+r     r-k+ah
oh+ch       sp          ll-ah+m     eh-b+eh     n-oh        k-w+ah      eh-y+s      k-oh        eh-r+oh     k-ah+r
oh-ch+oh    th+ih       ah-m+ah     b-eh        sp          w-ah+t      y-s         sp          r-oh        ah-r
ch-oh       th-ih+ng    m-ah+r      sp          s+y         ah-t+r      sp          sil         sp          sp
sp          ih-ng+k     ah-r        t+r         s-y+eh      t-r+oh      s+y         .           n+w         g+eh
uh+n        ng-k+oh     sp          t-r+eh      y-eh+t      r-oh        s-y+eh      "*/T0114.lab" n-w+eh    g-eh+b
uh-n+oh     k-oh        g+eh        r-eh+s      eh-t+eh     sp          y-eh+t      sil         w-eh+b      eh-b+ah
n-oh        sp          g-eh+b      eh-s        t-eh        sil         eh-t+eh     m+ah        eh-b+eh     b-ah+r
sp          t+r         eh-b+ah     sp          sp          .           t-eh        m-ah+r      b-eh        ah-r+ah
n+w         t-r+eh      b-ah+r      t+r         t+r         "*/T0110.lab" sp        ah-r+k      sp          r-ah
n-w+eh      r-eh+s      ah-r+ah     t-r+eh      t-r+eh      sil         s+y         r-k+ah      oh+ch       sp
w-eh+b      eh-s        r-ah        r-eh+s      r-eh+s      t+eh        s-y+eh      k-ah+r      oh-ch+oh    sil
eh-b+eh     sp          sp          eh-s        eh-s        t-eh+l      y-eh+t      ah-r        ch-oh       .
b-eh        sil         sil         sp          sp          eh-l+eh     eh-t+eh     sp          sp          "*/T0122.lab"
sp          .           .           n+w         s+y         l-eh+f      t-eh        j+oh        uh+n        sil
n+w         "*/T0099.lab" "*/T0105.lab" n-w+eh  s-y+eh      eh-f+oh     sp          j-oh+r      uh-n+oh     m+ah
n-w+eh      sil         sil         w-eh+b      y-eh+t      f-oh+n      th+ih       oh-r+j      n-oh        m-ah+r
w-eh+b      m+ah        m+ah        eh-b+eh     eh-t+eh     oh-n+oh     th-ih+ng    r-j+eh      sp          ah-r+k
eh-b+eh     m-ah+r      m-ah+r      b-eh        t-eh        n-oh        ih-ng+k     j-eh        th+ih       r-k+ah
b-eh        ah-r+k      ah-r+k      sp          sp          sp          ng-k+oh     sp          th-ih+ng    k-ah+r
sp          r-k+ah      r-k+ah      sil         oh+ch       uh+n        k-oh        l+uh        ih-ng+k     ah-r
```

```
sp          oh-ch+oh    sp          sil         oh+ch       sp          uh+n        oh+ch       g+eh        sp
l+uh        ch-oh       s+eh        .           oh-ch+oh    th+eh       uh-n+oh     oh-ch+oh    g-eh+b      l+uh
l-uh+y      sp          s-eh+y      "*/T0134.lab" ch-oh     th-eh+r     n-oh        ch-oh       eh-b+ah     l-uh+y
uh-y+s      dh+oh       eh-y+s      sil         sp          eh-r+oh     sp          sp          b-ah+r      uh-y+s
y-s         dh-oh+s     y-s         m+ah        s+eh        r-oh        th+ih       sil         ah-r+ah     y-s
sp          oh-s        sp          m-ah+r      s-eh+y      sp          th-ih+ng    .           r-ah        sp
sil         sp          t+r         ah-r+k      eh-y+s      th+ih       ih-ng+k     "*/T0142.lab" sp        g+eh
.           dh+oh       t-r+eh      r-k+ah      y-s         th-ih+ng    ng-k+oh     sil         sil         g-eh+b
"*/T0123.lab" dh-oh+s   r-eh+s      k-ah+r      sp          ih-ng+k     k-oh        m+ah        .           eh-b+ah
sil         oh-s        eh-s        ah-r        s+y         ng-k+oh     sp          m-ah+r      "*/T0148.lab" b-ah+r
t+eh        sp          sp          sp          s-y+eh      k-oh        t+r         ah-r+k      sil         ah-r+ah
t-eh+l      sil         sil         l+eh        y-eh+t      sp          t-r+eh      r-k+ah      m+ah        r-ah
eh-l+eh     .           .           l-eh+oh     eh-t+eh     oh+ch       r-eh+s      k-ah+r      m-ah+r      sp
l-eh+f      "*/T0127.lab" "*/T0130.lab" eh-oh+n  t-eh        oh-ch+oh    eh-s        ah-r        ah-r+k      sil
eh-f+oh     sil         sil         oh-n        sp          ch-oh       sp          sp          r-k+ah      .
f-oh+n      m+ah        t+eh        sp          th+eh       sp          dh+oh       j+oh        k-ah+r      "*/T0153.lab"
oh-n+oh     m-ah+r      t-eh+l      sil         th-eh+r     t+r         dh-oh+s     j-oh+r      ah-r        sil
n-oh        ah-r+k      eh-l+eh     .           eh-r+oh     t-r+eh      oh-s        oh-r+j      sp          t+eh
sp          r-k+ah      l-eh+f      "*/T0135.lab" r-oh      r-eh+s      sp          r-j+eh      l+uh        t-eh+l
uh+n        k-ah+r      eh-f+oh     sil         sp          eh-s        s+y         j-eh        l-uh+y      eh-l+eh
uh-n+oh     ah-r        f-oh+n      m+ah        uh+n        sp          s-y+eh      sp          uh-y+s      l-eh+f
n-oh        sp          oh-n+oh     m-ah+r      uh-n+oh     sil         y-eh+t      l+uh        y-s         eh-f+oh
sp          k+ah        n-oh        ah-r+k      n-oh        .           eh-t+eh     l-uh+y      sp          f-oh+n
sil         k-ah+r      sp          r-k+ah      sp          "*/T0140.lab" t-eh      uh-y+s      g+eh        oh-n+oh
.           ah-r+l      s+y         k-ah+r      t+r         sil         sp          y-s         g-eh+b      n-oh
"*/T0124.lab" r-l+oh    s-y+eh      ah-r        t-r+eh      ll+ah       th+ih       sp          eh-b+ah     sp
sil         l-oh+s      y-eh+t      sp          r-eh+s      ll-ah+m     th-ih+ng    sil         b-ah+r      s+y
ll+ah       oh-s        eh-t+eh     l+uh        eh-s        ah-m+ah     ih-ng+k     .           ah-r+ah     s-y+eh
ll-ah+m     sp          t-eh        l-uh+y      sp          m-ah+r      ng-k+oh     "*/T0143.lab" r-ah      y-eh+t
ah-m+ah     dh+ih       sp          uh-y+s      oh+ch       ah-r        k-oh        sil         sp          eh-t+eh
m-ah+r      dh-ih+ah    dh+oh       y-s         oh-ch+oh    sp          sp          m+ah        sil         t-eh
ah-r        ih-ah+s     dh-oh+s     sp          ch-oh       j+oh        uh+n        m-ah+r      .           sp
sp          ah-s        oh-s        g+eh        sp          j-oh+r      uh-n+oh     ah-r+k      "*/T0149.lab" s+eh
dh+ih       sp          sp          g-eh+b      dh+oh       oh-r+j      n-oh        r-k+ah      sil         s-eh+y
dh-ih+ah    sil         sil         eh-b+ah     dh-oh+s     r-j+eh      sp          k-ah+r      t+eh        eh-y+s
ih-ah+s     .           .           b-ah+r      oh-s        j-eh        s+eh        ah-r        t-eh+l      y-s
ah-s        "*/T0128.lab" "*/T0131.lab" ah-r+ah  sp         sp          s-eh+y      sp          eh-l+eh     sp
sp          sil         sil         r-ah        s+y         l+uh        eh-y+s      l+uh        l-eh+f      th+eh
sil         ll+ah       m+ah        sp          s-y+eh      l-uh+y      y-s         l-uh+y      eh-f+oh     th-eh+r
.           ll-ah+m     m-ah+r      sil         y-eh+t      uh-y+s      sp          uh-y+s      f-oh+n      eh-r+oh
"*/T0125.lab" ah-m+ah   ah-r+k      .           eh-t+eh     y-s         dh+oh       y-s         oh-n+oh     r-oh
sil         m-ah+r      r-k+ah      "*/T0136.lab" t-eh      sp          dh-oh+s     sp          n-oh        sp
m+ah        ah-r        k-ah+r      sil         sp          sil         oh-s        sil         sp          th+eh
m-ah+r      sp          ah-r        ll+ah       s+eh        .           sp          .           s+y         th-eh+r
ah-r+k      j+oh        sp          ll-ah+m     s-eh+y      "*/T0141.lab" k+w       "*/T0144.lab" s-y+eh    eh-r+oh
r-k+ah      j-oh+r      l+uh        ah-m+ah     eh-y+s      sil         k-w+ah      sil         y-eh+t      r-oh
k-ah+r      oh-r+j      l-uh+y      m-ah+r      y-s         t+eh        w-ah+t      ll+ah       eh-t+eh     sp
ah-r        r-j+eh      uh-y+s      ah-r        sp          t-eh+l      ah-t+r      ll-ah+m     t-eh        s+eh
sp          j-eh        y-s         sp          t+r         eh-l+eh     t-r+oh      ah-m+ah     sp          s-eh+y
l+uh        sp          sp          l+eh        t-r+eh      l-eh+f      r-oh        m-ah+r      th+ih       eh-y+s
l-uh+y      l+uh        g+eh        l-eh+y      r-eh+s      eh-f+oh     sp          ah-r        th-ih+ng    y-s
uh-y+s      l-uh+y      g-eh+b      eh-y+s      eh-s        f-oh+n      th+ih       sp          ih-ng+k     sp
y-s         uh-y+s      eh-b+ah     y-s+ih      sp          oh-n+oh     th-ih+ng    g+eh        ng-k+oh     s+y
sp          y-s         b-ah+r      s-ih        dh+oh       n-oh        ih-ng+k     g-eh+b      k-oh        s-y+eh
sil         sp          ah-r+ah     sp          dh-oh+s     sp          ng-k+oh     eh-b+ah     sp          y-eh+t
.           sil         r-ah        l+eh        oh-s        th+eh       k-oh        b-ah+r      sil         eh-t+eh
"*/T0126.lab" .         sp          l-eh+oh     sp          th-eh+r     sp          ah-r+ah     .           t-eh
sil         "*/T0129.lab" sil       eh-oh+n     th+w        eh-r+oh     n+w         r-ah        "*/T0150.lab" sp
t+eh        sil         .           oh-n        n-w+eh      r-oh        n-w+eh      sp          sil         oh+ch
t-eh+l      t+eh        "*/T0132.lab" sp        w-eh+b      sp          w-eh+b      sil         ll+ah       oh-ch+oh
eh-l+eh     t-eh+l      sil         sil         eh-b+eh     oh+ch       eh-b+eh     .           ll-ah+m     ch-oh
l-eh+f      eh-l+eh     m+ah        .           b-eh        oh-ch+oh    b-eh        "*/T0145.lab" ah-m+ah   sp
eh-f+oh     l-eh+f      m-ah+r      "*/T0137.lab" sp        ch-oh       sp          sil         m-ah+r      s+eh
f-oh+n      eh-f+oh     ah-r+k      sil         th+eh       sp          n+w         ll+ah       ah-r        s-eh+y
oh-n+oh     f-oh+n      r-k+ah      t+eh        th-eh+r     th+ih       n-w+eh      ll-ah+m     sp          eh-y+s
n-oh        oh-n+oh     k-ah+r      t-eh+l      eh-r+oh     th-ih+ng    w-eh+b      ah-m+ah     l+uh        y-s
sp          n-oh        ah-r        eh-l+eh     r-oh        ih-ng+k     eh-b+eh     m-ah+r      l-uh+y      sp
k+w         sp          sp          l-eh+f      sp          ng-k+oh     b-eh        ah-r        uh-y+s      k+w
k-w+ah      uh+n        j+oh        eh-f+oh     th+ih       k-oh        sp          sp          y-s         k-w+ah
w-ah+t      uh-n+oh     j-oh+r      f-oh+n      th-ih+ng    sp          s+eh        dh+ih       sp          w-ah+t
ah-t+r      n-oh        oh-r+j      oh-n+oh     ih-ng+k     n+w         s-eh+y      dh-ih+ah    g+eh        ah-t+r
t-r+oh      sp          r-j+eh      n-oh        ng-k+oh     n-w+eh      eh-y+s      ih-ah+s     g-eh+b      t-r+oh
r-oh        th+eh       j-eh        sp          k-oh        w-eh+b      y-s         ah-s        eh-b+ah     r-oh
sp          th-eh+r     sp          t+r         sp          eh-b+eh     sp          sp          b-ah+r      sp
t+r         eh-r+oh     l+uh        t-r+eh      sil         b-eh        s+y         sil         ah-r+ah     t+r
t-r+eh      r-oh        l-uh+y      r-eh+s      .           sp          s-y+eh      .           r-ah        t-r+eh
r-eh+s      sp          uh-y+s      eh-s        "*/T0138.lab" th+ih     y-eh+t      "*/T0146.lab" sp        r-eh+s
eh-s        s+y         y-s         sp          sil         th-ih+ng    eh-t+eh     sil         sil         eh-s
sp          s-y+eh      sp          k+w         ll+ah       ih-ng+k     t-eh        m+ah        .           sp
k+w         y-eh+t      sil         k-w+ah      ll-ah+m     ng-k+oh     sp          m-ah+r      "*/T0151.lab" s+eh
k-w+ah      eh-t+eh     .           w-ah+t      ah-m+ah     k-oh        th+ih       ah-r+k      sil         s-eh+y
w-ah+t      t-eh        "*/T0133.lab" ah-t+r    m-ah+r      sp          th-ih+ng    r-k+ah      ll+ah       eh-y+s
ah-t+r      sp          sil         t-r+oh      ah-r        th+ih       ih-ng+k     k-ah+r      ll-ah+m     y-s
t-r+oh      s+y         m+ah        r-oh        sp          th-ih+ng    ng-k+oh     ah-r        ah-m+ah     sp
r-oh        s-y+eh      m-ah+r      sp          dh+ih       ih-ng+k     k-oh        sp          m-ah+r      s+y
sp          y-eh+t      ah-r+k      uh+n        dh-ih+ah    ng-k+oh     sp          dh+ih       ah-r        s-y+eh
dh+oh       eh-t+eh     r-k+ah      uh-n+oh     ih-ah+s     k-oh        t+r         dh-ih+ah    sp          y-eh+t
dh-oh+s     t-eh        k-ah+r      n-oh        ah-s        sp          t-r+eh      ih-ah+s     l+eh        eh-t+eh
oh-s        sp          ah-r        sp          sp          dh+oh       r-eh+s      ah-s        l-eh+oh     t-eh
sp          k+w         sp          uh+n        sil         dh-oh+s     eh-s        sp          eh-oh+n     sp
n+w         k-w+ah      l+eh        uh-n+oh     .           oh-s        sp          sil         oh-n        k+w
n-w+eh      w-ah+t      l-eh+y      n-oh        "*/T0139.lab" sp        n+w         .           sp          k-w+ah
w-eh+b      ah-t+r      eh-y+s      sp          sil         t+r         n-w+eh      "*/T0147.lab" sil       w-ah+t
eh-b+eh     t-r+oh      y-s+ih      dh+oh       t+eh        t-r+eh      w-eh+b      sil         .           ah-t+r
b-eh        r-oh        s-ih        dh-oh+s     t-eh+l      r-eh+s      eh-b+eh     m+ah        "*/T0152.lab" t-r+oh
sp          sp          sp          oh-s        eh-l+eh     eh-s        b-eh        m-ah+r      sil         r-oh
uh+n        s+y         l+eh        sp          l-eh+f      sp          sp          ah-r+k      ll+ah       sp
uh-n+oh     s-y+eh      l-eh+oh     uh+n        eh-f+oh     dh+oh       oh+ch       r-k+ah      ll-ah+m     k+w
n-oh        y-eh+t      eh-oh+n     uh-n+oh     f-oh+n      dh-oh+s     oh-ch+oh    k-ah+r      ah-m+ah     k-w+ah
sp          eh-t+eh     oh-n        n-oh        oh-n+oh     oh-s        ch-oh       ah-r        m-ah+r      w-ah+t
oh+ch       t-eh        sp          sp          n-oh        sp          sp          sp          ah-r        ah-t+r
```

```
t-r+oh
r-oh
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
n+w
n-w+eh
w-eh+b
eh-b+eh
b-eh
sp
th+eh
th-eh+r
eh-r+oh
r-oh
sp
dh+oh
dh-oh+s
oh-s
sp
oh+ch
oh-ch+oh
ch-oh
sp
uh+n
uh-n+oh
n-oh
sp
th+eh
th-eh+r
eh-r+oh
r-oh
sp
t+r
t-r+eh
r-eh+s
eh-s
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
oh+ch
oh-ch+oh
ch-oh
sp
dh+oh
dh-oh+s
oh-s
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
th+eh
th-eh+r
eh-r+oh
r-oh
sp
oh+ch
oh-ch+oh
ch-oh
sp
t+r
t-r+eh
r-eh+s
eh-s
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
n+w
n-w+eh
w-eh+b
eh-b+eh
b-eh
sp
dh+oh
dh-oh+s
oh-s
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
th+eh
th-eh+r
eh-r+oh
r-oh
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
n+w
n-w+eh
w-eh+b
eh-b+eh
b-eh
sp
th+eh
th-eh+r
eh-r+oh
r-oh
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
oh+ch
oh-ch+oh
ch-oh
sp
sil
.
"*/T0154.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
l+eh
l-eh+oh
eh-oh+n
oh-n
sp
sil
.
"*/T0155.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
dh+ih
dh-ih+ah
ih-ah+s
ah-s
sp
sil
.
"*/T0156.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
k+ah
k-ah+r
ah-r+l
r-l+oh
l-oh+s
oh-s
sp
dh+ih
dh-ih+ah
ih-ah+s
ah-s
sp
sil
.
"*/T0157.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
uh+n
uh-n+oh
n-oh
sp
n+w
n-w+eh
w-eh+b
eh-b+eh
b-eh
sp
th+eh
th-eh+r
eh-r+oh
r-oh
sp
t+r
t-r+eh
r-eh+s
eh-s
sp
n+w
n-w+eh
w-eh+b
eh-b+eh
b-eh
sp
uh+n
uh-n+oh
n-oh
sp
t+r
t-r+eh
r-eh+s
eh-s
sp
n+w
n-w+eh
w-eh+b
eh-b+eh
b-eh
sp
oh+ch
oh-ch+oh
ch-oh
sp
oh+ch
oh-ch+oh
ch-oh
sp
sil
.
"*/T0158.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
j+oh
j-oh+r
oh-r+j
r-j+eh
j-eh
sp
l+uh
l-uh+y
uh-y+s
y-s
sp
sil
.
"*/T0159.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
n+w
n-w+eh
w-eh+b
eh-b+eh
b-eh
sp
t+r
t-r+eh
r-eh+s
eh-s
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
oh+ch
oh-ch+oh
ch-oh
sp
n+w
n-w+eh
w-eh+b
eh-b+eh
b-eh
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
n+w
n-w+eh
w-eh+b
eh-b+eh
b-eh
sp
n+w
n-w+eh
w-eh+b
eh-b+eh
b-eh
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
uh+n
uh-n+oh
n-oh
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
uh+n
uh-n+oh
n-oh
sp
uh+n
uh-n+oh
n-oh
sp
oh+ch
oh-ch+oh
ch-oh
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
oh+ch
oh-ch+oh
ch-oh
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
dh+oh
dh-oh+s
oh-s
sp
th+eh
th-eh+r
eh-r+oh
r-oh
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
sil
.
"*/T0160.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
uh+n
uh-n+oh
n-oh
sp
sil
.
"*/T0161.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
dh+oh
dh-oh+s
oh-s
sp
th+eh
th-eh+r
eh-r+oh
r-oh
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
uh+n
uh-n+oh
n-oh
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
dh+oh
dh-oh+s
oh-s
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
sil
.
"*/T0162.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
l+uh
l-uh+y
uh-y+s
y-s
sp
g+eh
g-eh+b
eh-b+ah
b-ah+r
ah-r+ah
r-ah
sp
sil
.
"*/T0163.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
n+w
n-w+eh
w-eh+b
eh-b+eh
b-eh
sp
oh+ch
oh-ch+oh
ch-oh
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
n+w
n-w+eh
w-eh+b
eh-b+eh
b-eh
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
n+w
n-w+eh
w-eh+b
eh-b+eh
b-eh
sp
sil
.
"*/T0164.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
th+eh
th-eh+r
eh-r+oh
r-oh
sp
oh+ch
oh-ch+oh
ch-oh
sp
t+r
t-r+eh
r-eh+s
eh-s
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
uh+n
uh-n+oh
n-oh
sp
uh+n
uh-n+oh
n-oh
sp
t+r
t-r+eh
r-eh+s
eh-s
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
sil
.
"*/T0165.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
j+oh
j-oh+r
oh-r+j
r-j+eh
j-eh
sp
l+uh
l-uh+y
uh-y+s
y-s
sp
sil
.
"*/T0166.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
g+eh
g-eh+b
eh-b+ah
b-ah+r
ah-r+ah
r-ah
sp
sil
.
"*/T0167.lab"
sil
m+ah
m-ah+r
ah-r+k
r-k+ah
k-ah+r
ah-r
sp
l+uh
l-uh+y
uh-y+s
y-s
sp
g+eh
g-eh+b
eh-b+ah
b-ah+r
ah-r+ah
r-ah
sp
sil
.
"*/T0168.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
n+w
n-w+eh
w-eh+b
eh-b+eh
b-eh
sp
sil
.
"*/T0169.lab"
sil
m+ah
m-ah+r
ah-r+k
r-k+ah
k-ah+r
ah-r
sp
dh+ih
dh-ih+ah
ih-ah+s
ah-s
sp
sil
.
"*/T0170.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
uh+n
uh-n+oh
n-oh
sp
sil
.
"*/T0171.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
uh+n
uh-n+oh
n-oh
sp
t+r
t-r+eh
r-eh+s
eh-s
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
n+w
n-w+eh
w-eh+b
eh-b+eh
b-eh
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
dh+oh
dh-oh+s
oh-s
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
uh+n
uh-n+oh
n-oh
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
sil
.
"*/T0172.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
l+uh
l-uh+y
uh-y+s
y-s
sp
sil
.
"*/T0173.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
j+oh
j-oh+r
oh-r+j
r-j+eh
j-eh
sp
l+uh
l-uh+y
uh-y+s
y-s
sp
sil
.
"*/T0174.lab"
sil
m+ah
m-ah+r
ah-r+k
r-k+ah
k-ah+r
ah-r
sp
dh+ih
dh-ih+ah
ih-ah+s
ah-s
sp
sil
.
"*/T0175.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
th+eh
th-eh+r
eh-r+oh
r-oh
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
dh+oh
dh-oh+s
oh-s
sp
oh+ch
oh-ch+oh
ch-oh
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
th+eh
th-eh+r
eh-r+oh
r-oh
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
dh+oh
dh-oh+s
oh-s
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
uh+n
uh-n+oh
n-oh
```

```
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
t+r
t-r+eh
r-eh+s
eh-s
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
sil
.
"*/T0176.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
t+r
t-r+eh
r-eh+s
eh-s
sp
th+eh
th-eh+r
eh-r+oh
r-oh
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
dh+oh
dh-oh+s
oh-s
sp
uh+n
uh-n+oh
n-oh
sp
n+w
n-w+eh
w-eh+b
eh-b+eh
b-eh
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
t+r
t-r+eh
r-eh+s
eh-s
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
th+eh
th-eh+r
eh-r+oh
r-oh
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
uh+n
uh-n+oh
n-oh
sp
oh+ch
oh-ch+oh
ch-oh
sp
uh+n
uh-n+oh
n-oh
sp
sil
.
"*/T0177.lab"
sil
m+ah
m-ah+r
ah-r+k
r-k+ah
k-ah+r
ah-r
sp
l+eh
l-eh+y
eh-y+s
y-s+ih
s-ih
sp
l+eh
l-eh+oh
eh-oh+n
oh-n
sp
sil
.
"*/T0178.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
t+r
t-r+eh
r-eh+s
eh-s
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
t+r
t-r+eh
r-eh+s
eh-s
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
oh+ch
oh-ch+oh
ch-oh
sp
th+eh
th-eh+r
eh-r+oh
r-oh
sp
dh+oh
dh-oh+s
oh-s
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
sil
.
"*/T0179.lab"
sil
m+ah
m-ah+r
ah-r+k
r-k+ah
k-ah+r
ah-r
sp
l+uh
l-uh+y
uh-y+s
y-s
sp
g+eh
g-eh+b
eh-b+ah
b-ah+r
ah-r+ah
r-ah
sp
sil
.
"*/T0180.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
t+r
t-r+eh
r-eh+s
eh-s
sp
t+r
t-r+eh
r-eh+s
eh-s
sp
sil
.
"*/T0181.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
l+uh
l-uh+y
uh-y+s
y-s
sp
sil
.
"*/T0182.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
j+oh
j-oh+r
oh-r+j
r-j+eh
j-eh
sp
l+uh
l-uh+y
uh-y+s
y-s
sp
sil
.
"*/T0183.lab"
sil
m+ah
m-ah+r
ah-r+k
r-k+ah
k-ah+r
ah-r
sp
j+oh
j-oh+r
oh-r+j
r-j+eh
j-eh
sp
l+uh
l-uh+y
uh-y+s
y-s
sp
sil
.
"*/T0184.lab"
sil
m+ah
m-ah+r
ah-r+k
r-k+ah
k-ah+r
ah-r
sp
g+eh
g-eh+b
eh-b+ah
b-ah+r
ah-r+ah
r-ah
sp
sil
.
"*/T0185.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
dh+oh
dh-oh+s
oh-s
sp
t+r
t-r+eh
r-eh+s
eh-s
sp
sil
.
"*/T0186.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
g+eh
g-eh+b
eh-b+ah
b-ah+r
ah-r+ah
r-ah
sp
sil
.
"*/T0187.lab"
sil
m+ah
m-ah+r
ah-r+k
r-k+ah
k-ah+r
ah-r
sp
j+oh
j-oh+r
oh-r+j
r-j+eh
j-eh
sp
l+uh
l-uh+y
uh-y+s
y-s
sp
sil
.
"*/T0188.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
th+eh
th-eh+r
eh-r+oh
r-oh
sp
uh+n
uh-n+oh
n-oh
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
th+eh
th-eh+r
eh-r+oh
r-oh
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
t+r
t-r+eh
r-eh+s
eh-s
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
uh+n
uh-n+oh
n-oh
sp
uh+n
uh-n+oh
n-oh
sp
uh+n
uh-n+oh
n-oh
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
sil
.
"*/T0189.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
j+oh
j-oh+r
oh-r+j
r-j+eh
j-eh
sp
l+uh
l-uh+y
uh-y+s
y-s
sp
sil
.
"*/T0190.lab"
sil
m+ah
m-ah+r
ah-r+k
r-k+ah
k-ah+r
ah-r
sp
l+eh
l-eh+y
eh-y+s
y-s+ih
s-ih
sp
l+eh
l-eh+oh
eh-oh+n
oh-n
sp
sil
.
"*/T0191.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
g+eh
g-eh+b
eh-b+ah
b-ah+r
ah-r+ah
r-ah
sp
sil
.
"*/T0192.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
l+uh
l-uh+y
uh-y+s
y-s
sp
sil
.
"*/T0193.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
dh+oh
dh-oh+s
oh-s
sp
t+r
t-r+eh
r-eh+s
eh-s
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
sil
.
"*/T0194.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
k+ah
k-ah+r
ah-r+l
r-l+oh
l-oh+s
oh-s
sp
dh+ih
dh-ih+ah
ih-ah+s
ah-s
sp
sil
.
"*/T0195.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
th+eh
th-eh+r
eh-r+oh
r-oh
sp
sil
.
"*/T0196.lab"
sil
m+ah
m-ah+r
ah-r+k
r-k+ah
k-ah+r
ah-r
sp
g+eh
g-eh+b
eh-b+ah
b-ah+r
ah-r+ah
r-ah
sp
sil
.
"*/T0197.lab"
sil
m+ah
m-ah+r
ah-r+k
r-k+ah
k-ah+r
ah-r
sp
k+ah
k-ah+r
ah-r+l
r-l+oh
l-oh+s
oh-s
sp
dh+ih
dh-ih+ah
ih-ah+s
ah-s
sp
sil
.
"*/T0198.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
th+eh
th-eh+r
eh-r+oh
r-oh
sp
oh+ch
oh-ch+oh
ch-oh
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
th+eh
th-eh+r
eh-r+oh
r-oh
sp
dh+oh
dh-oh+s
oh-s
sp
uh+n
uh-n+oh
n-oh
sp
th+eh
th-eh+r
eh-r+oh
r-oh
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
oh+ch
oh-ch+oh
ch-oh
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
t+r
t-r+eh
r-eh+s
eh-s
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
s+y
s-y+eh
y-eh+t
eh-t+eh
t-eh
sp
t+r
t-r+eh
r-eh+s
eh-s
sp
dh+oh
dh-oh+s
oh-s
sp
th+ih
th-ih+ng
ih-ng+k
ng-k+oh
k-oh
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
th+eh
th-eh+r
eh-r+oh
r-oh
sp
sil
.
"*/T0199.lab"
sil
ll+ah
ll-ah+m
ah-m+ah
m-ah+r
ah-r
sp
l+eh
l-eh+oh
eh-oh+n
oh-n
sp
sil
.
"*/T0200.lab"
sil
t+eh
t-eh+l
eh-l+eh
l-eh+f
eh-f+oh
f-oh+n
oh-n+oh
n-oh
sp
s+eh
s-eh+y
eh-y+s
y-s
sp
k+w
k-w+ah
w-ah+t
ah-t+r
t-r+oh
r-oh
sp
th+eh
th-eh+r
eh-r+oh
r-oh
sp
dh+oh
dh-oh+s
oh-s
sp
uh+n
uh-n+oh
n-oh
sp
sil
.
```

A.7. Código Fuente

A.7.1. Código de Inicialización de HMM's

A continuación el código fuente de los algoritmos de inicialización del HMM en Java

```
package modeloocultodemarkov;

/**
 *
 * @author jorge
 */
public class Algoritmos {
    //Algoritmo k means
    //referencia: Machine Learning Tom Mitchell Pag 191
    //@param Y  observaciones dxn (n datos d dimesionales)
    public static Cluster[] kMeans(double [][] Y, int nroCluster)
    {
        Cluster [] cluster=new Cluster[nroCluster];
        int d=Y.length; // d= dimension del vector
        int n=Y[0].length;//n=cantidad de vectores de observacion
        int i,j,k;
        double distancia;
        double min;
        int pos=0;

        //criterios para evaluar el error cuadratico medio
        double umbralError=0.000000001;
        double error=0;//error actual
        double errorAnterior=0;;//error anterior

        System.out.println(" ");
        System.out.println(".................");
        System.out.println("Algoritmo Kmeans");
        System.out.println(".................");
        //inicializar clusters
        for(i=0;i<nroCluster;i++)
        {
            cluster[i]=new Cluster(d);
        }

        //KMedias es una matriz con las medias, una media por cluster
        double [][] kMedias =new double[d][nroCluster];

        //escoger aleatoriamente k medias

        for(k=0;k<nroCluster;k++)
        {
            j=(int)Math.round(Util.T(Math.random(), 0, 1, 0, n-1));
             //System.out.println(""+Math.round(Util.T(Math.random(), 0, 1, 0, Y[0].length-1)));
            for(i=0;i<d;i++)//para cada dimension
            {
                kMedias[i][k]=Y[i][j];
            }
        }

        System.out.println("Medias Iniciales");
        Matrices.mostrar(kMedias);

        //error cuadratico medio inicial

        //proceso de clustering

        do{
            errorAnterior=error;
            //para todos los vectores de observacion Y
            for(j=0;j<n;j++)
            {
                min=100000000;
                pos=0;
                for(k=0;k<nroCluster;k++)
                {
                    //calculo de la distancia
                    distancia=0;
                    //System.out.println("vectores");
                    for(i=0;i<d;i++)
                    {
                        distancia=distancia+Math.pow(Y[i][j]-kMedias[i][k],2);
                      //  System.out.println(""+Y[i][j]+" "+kMedias[i][k]);
                    }
                    distancia=Math.sqrt(distancia);
                   // System.out.println(""+distancia);
                    if(distancia<min)
                    {
                        min=distancia;
                        pos=k;
                    }
                }

            //guardar datos en el cluster
                double [][]transpuestaY=Matrices.traspuesta(Y);

                cluster[pos].addDatos(transpuestaY[j]);

            }

            //actualizar medias
            kMedias=Matrices.traspuesta(kMedias);
```

```
            for(i=0;i<nroCluster;i++)
            {
                cluster[i].actualizarMedia();
                cluster[i].actualizarCoovarianzaDiagona();
                kMedias[i]=cluster[i].getMedia();
            }
            kMedias=Matrices.traspuesta(kMedias);

            for(i=0;i<nroCluster;i++)
            {
                System.out.println(" ");
                System.out.println("------------------------ ");
                System.out.println("cluster "+i);
                cluster[i].imprimir();
            }

   //          System.out.println("kMedias");
   //          Matrices.mostrar(kMedias);

            //calcular el error cuadratico medio
            error=0;
            for(i=0;i<nroCluster;i++)
            {
             error=error+cluster[i].getErrorCuadraticoMedio();
            }
            System.out.println("");
            System.out.println("---------------");
            System.out.println("\nError "+error);
            System.out.println("---------------");

            //limpiar los clusters
            if(Math.abs(error-errorAnterior)>umbralError)
            {
                for(i=0;i<nroCluster;i++)
                {
                 cluster[i].clear();
                }
            }

        }while(Math.abs(error-errorAnterior)>umbralError);
        System.out.println("---------------");
        System.out.println("Fin kMeans");
        System.out.println("---------------");
        System.out.println("");
        return cluster;

    }
}

package modeloocultodemarkov;

/**
 *
 * @author usuario
 */
public class Cluster {

    double[][] datos;   //datos dxn n datos d dimensionales
    double [] media;
    double [] coovarianzaDiagonal;
    int datosPorCluster; //
    int dim;

    //ingresa la dimension de los vectores que almacena este cluster
    public Cluster(int dim)
    {
        this.dim=dim;
        this.datos=new double[dim][0];
        datosPorCluster=0;
        media=generarMedia();
    }

    public void clear()
    {
        this.datos=new double[dim][0];
        datosPorCluster=0;
    }
    //obtiene el error cuadratico medio de los datos con respecto a la media
    public double getErrorCuadraticoMedio()
    {
        if(datos[0].length!=0)
        {
        double errorCM=0;//error cuadratico medio
        datos=Matrices.traspuesta(datos);
        for(int i=0;i<datos.length;i++)
        {
            errorCM=errorCM+Math.pow(Util.distancia(datos[i], media), 2);
        }
        datos=Matrices.traspuesta(datos);
        return errorCM;
        }
        return 0.0;
    }
    public void addDatos(double []Y)
    {
        datos=Matrices.redimensionar(datos, Y);
        datosPorCluster++;
    }

    public double[] generarMedia()
```

```
    {
        double []u=new double[dim];
        for(int i=0;i<dim;i++)
        {
            u[i]=0;
        }
        return u;
    }
    public void addMedia(double [] media)
    {
        this.media=media;
    }

    public void actualizarMedia()
    {
        if(datos[0].length!=0)
        {
            media=Matrices.promedio(datos);
        }

    }

    public void actualizarCoovarianzaDiagona()
    {
        coovarianzaDiagonal=Matrices.coovarianzaDiagonal(datos, media);
    }

    public double[] getMedia()
    {
        return media;
    }

    public double[] getMatrizCoovarianzaDiagonal()
    {
        return coovarianzaDiagonal;
    }
    public int getNumeroDatosPorCluster()
    {
        return datosPorCluster;
    }

    public void imprimir()
    {
        System.out.println("    Datos");
        Matrices.mostrar(datos);
        System.out.println("    Media");
        Matrices.mostrar(media);
        System.out.println("    Coovarianza Diagonal");
        Matrices.mostrar(coovarianzaDiagonal);
        System.out.println("    Datos por cluster");
        System.out.print("      ");
        System.out.println(datosPorCluster);
    }

}
```

A.7.2. Código de los Modelos Ocultos de Markov HMM

A continuación el código fuente de los modelos ocultos de Markov HMM Java

```
package modeloocultodemarkov;

/**ModeloOcultoDeMarkov
 *
 * @author usuario
 */
public class Estado {

    private String id;
    //probabilidades
   // private Observacion b;
    public double[] alfaLog;  //arreglo de logaritmo de alfa, cada posicion indica un instante de tiempo
    public double[] betaLog;  //arreglo de logaritmo de beta, cada posicion indica un instante de tiempo
    public double[] alfaViterbiLog;//arreglo de logaritmo de alfaViterbi, cada posicion indica un instante de tiempo
    public double[][] gamma_jm_Log; //arreglo de logaritmo de "m".gamma, cada posicion indica un instante de tiempo, ver ec 9.43
    public double [] GammaLog;
    private double pi;

    //almacena las observaciones para el entrenamiento, luego son borradas
    public double[][] observaciones;
    //log prob de la observacion es decir se obtiene llamando a getLopProb de su GMM respectivo
    private double logProb;
    //numero de gaussiamas para el GMM de este estado
    private int nroGaussianas;
    //GMM para el estado
    public GMM gmm;


    public Estado(String id)
    {
        this.id=id;
    }
    /*Agrega todas los vectores de observacion resultado de la segmentacion inicial
      a dicho estado
      */
    public void addObservacion(double[]y)
    {
```

```
        if(observaciones==null)
        {
            observaciones = new double[y.length][0];
        }
        this.observaciones=Matrices.redimensionar(observaciones, y);
    }

    public void borrarObservacion()
    {
        this.observaciones=null;
    }
     /*Inicializa el estado es decir genera el GMM para este estado*/
    public void inicializar()
    {
        gmm=new GMM(observaciones, nroGaussianas);
    }
    //b es la funcion de propabilidad para el la observacion y_t vector y en tiempo t
    //retorna el logaritmo de la probabilidad de y_t asociado al GMM de un estado
    public double b(double []y)
    {
        return gmm.getLogProb(y);
    }

    public GMM getGmm() {
        return gmm;
    }

    public double[] getGammaLog() {
        return GammaLog;
    }

    public double[][] getGamma_jm_Log() {
        return gamma_jm_Log;
    }

    public void setGammaLog(double[] GammaLog) {
        this.GammaLog = GammaLog;
    }

    public void setGamma_jm_Log(double[][] gamma_jm_Log) {
        this.gamma_jm_Log = gamma_jm_Log;
    }

    //notar que este T en T+2 en HMM
    public void setGamma_ij_Log(int M, int T) {
        gamma_jm_Log = new double[M][T];
        //inicializa todo el vector con 0's
        for(int m=0;m<M;m++)
            for(int t=0;t<T;t++)
                gamma_jm_Log[m][t]=0;
    }

    public double[] getBetaLog() {
        return betaLog;
    }

    public double[] getAlfaLog() {
        return alfaLog;
    }

    public void setGmm(GMM gmm) {
        this.gmm = gmm;
    }

//  public void setGammaLog(double[][] gamma_jm_Log) {
//      this.gamma_jm_Log = gamma_jm_Log;
//  }


    public void setBetaLog(double[] betaLog) {
        this.betaLog = betaLog;
    }

    public void setAlfaLog(double[] alfaLog) {
        this.alfaLog = alfaLog;
    }

    public void setAlfaLog(int tam) {
        alfaLog = new double[tam];
        //inicializa todo el vector con 0's
        for(int i=0;i<tam;i++)
            alfaLog[i]=0;
    }

    public void setGammaLog(int tam) {
        GammaLog = new double[tam];
        //inicializa todo el vector con 0's
        for(int i=0;i<tam;i++)
            GammaLog[i]=0;
    }

    public void setBetaLog(int tam){
        betaLog=new double[tam];
        for(int i=0;i<tam;i++)
            betaLog[i]=0;
    }

    public void setAlfaViterbiLog(int tam)
    {
        alfaViterbiLog=new double[tam];
        for(int i=0;i<tam;i++)
```

```
            alfaViterbiLog[i]=0;

    }


    public void setAlfaLog(int pos,double alfaLogProb)
    {
        this.alfaLog[pos]=alfaLogProb;
    }

    public void setNroGaussianas(int nroGaussianas) {
        this.nroGaussianas = nroGaussianas;
    }

    public void setObservaciones(double[][] observaciones) {
        this.observaciones = observaciones;
    }

    public void setLogProb(double logProb) {
        this.logProb = logProb;
    }

    public void setId(String id) {
        this.id = id;
    }

    public void setPi(double pi) {
        this.pi = pi;
    }

    public void setAlfaViterbiLog(double[] alfaViterbiLog) {
        this.alfaViterbiLog = alfaViterbiLog;
    }

    public double getPi() {
        return pi;
    }

    public double[][] getObservaciones() {
        return observaciones;
    }

    public int getNroGaussianas() {
        return nroGaussianas;
    }

    public double getLogProb() {
        return logProb;
    }

    public String getId() {
        return id;
    }

    public double[] getAlfaViterbiLog() {
        return alfaViterbiLog;
    }


}

package modeloocultodemarkov;

/**
 *Clase que implementa un HMM
 *notacion a usar del libro Speech Recognition and Synthesis capitulo 9
 * Y= vectores columna de observaciones
 * S=estados siemrep T+2 estados : 1 estado inicial T estados para los simbolos de observacion y 1 estado final
 * a=transiciones
 * b=observaciones
 */
public class HMM
{
    Estado [] S;  //estados s[0] , s[1]... en ese orden
    double [][]A; //matriz de transiciones
    int [][] grafoHMM;//matriz que guarda la configuracion del HMM

    public HMM(int nroEstados)//nroEstados=T+2 estados
    {
        S=new Estado[nroEstados];
        A=new double[nroEstados][nroEstados];
        grafoHMM=new int[nroEstados][nroEstados];
        inicializarEstados();
    }

    public void agregarTransiciones(int i, int j)
    {
        //inicializaremos todas las transiciones que tienen participación en el modelo a 1
        //luego calcularemos las probabilidades iniciales de estas transiciones
        //el enfoque será dividirlas según el numero de conexiones de salida
        //así por ejemplo A[i][j]=1 se podría convertir en 1/3 si para el estado i
        // existen 3 conexiones de salida eso se ha´ra en inicializarAij
        A[i][j]=1;
        grafoHMM[i][j]=1;
    }

    public void inicializarAij()
    {
        double suma;
        int i=0;
        int j=0;
```

```
        for(i=0;i<A.length;i++)
        {
            suma=0;

            for(j=0;j<A[0].length;j++)
                suma=suma+A[i][j];

            suma=1.0/suma;

            for(j=0;j<A[0].length;j++)
                if(A[i][j]!=0)
                    A[i][j]=suma;

        }
    }

    /*inicializa los estados
     */

    private void inicializarEstados()
    {
        S[0]=new Estado("Estado = Inicial");
        S[S.length-1]=new Estado("Estado = Final");
        for(int i=1;i<S.length-1;i++)
            S[i]=new Estado("Estado = "+i);

    }
    //Entra la matriz de observaciones para una señal de habla
    //debemos llamar tantas veces esta función como señales de habla existan en
    //la base de entrenamiento (testeado funciona ok)
    //lo que hace es distribuir uniformemente las observaciones/ estado
    public void addObservaciones(double y[][])
    {
        int nroEstados=S.length;
        //nroEstados-2 pues solo T estados emiten simbolos de observacion
        double nroObsPorEstado=(double)y[0].length/(double)(nroEstados-2);

        //asignando una cantidad de observaciones proporcional a cada estado

        //trabajar con las traspuestas
        double [][]yTranspuest = Matrices.traspuesta(y);
        int j=1;                        //para cada estado
        for(int i=0;i<y[0].length;i++)  //para cada observacion
        {
            if(i<j*nroObsPorEstado)
                S[j].addObservacion(yTranspuest[i]);
            else
            {   j++;
                S[j].addObservacion(yTranspuest[i]);
            }
        }
    }

    /*utilizar este metodo si se desea inicializar todos los estados con el mismo numero de gaussianas
     *si se desea ingresar un numero de gaussianas por estado diferente usar el metodo setNroGaussianas de Estado
     * de manera individual (testeado funciona ok) y luego llamar a inicializar Aij
     * ejemplo:
     *
     */
    public void inicializarGaussianas(int nroGaussianas)
    {
        //para los T estados (en total son T+2 estados)
        for(int i=1;i<S.length-1;i++)
        {
            S[i].setNroGaussianas(nroGaussianas);
        }
    }

    /*inicializa la matriz de transiciones A y los GMM para cada estado
     */
    public void inicializarKmeansGMM()
    {
        inicializarAij();
        for(int i=1;i<S.length-1;i++)
        {
            S[i].inicializar();
        }
    }

    public void entrenarHMM(Patron patronesE[])
    {
        int i=0;
        int E=patronesE.length;
        double sum=0;

        while(i<10)
        {
            algoritmoBaumWelch(patronesE);
            this.mostrar();
            this.verificar();

            for(int e=0;e<E;e++)
                sum=sum+algoritmoViterbiII(patronesE[e].datos);

            System.out.println("log P(O/modelo)= "+sum+"     P(O/modelo) = "+Math.exp(sum));

            sum=0;
            i++;

        }
```

```
}

/*algoritmo de viterbi
 * este procedimiento es el mismo que algoritmode viterbi pero solo sirve
 * para testear P(O/modelo) y no actualiza las variables del modelo oculto
 */
public double algoritmoViterbiII(double [][]y)
{
    int N=S.length-2;       //numero de estados sin contar estado inicial ni final
    int T=y[0].length;      //numero de frames T
    int I=0;                //estado inicial
    int F=S.length-1;       //estado final
    double [][] estados=Matrices.ceros(S.length, T+2);

    double[][]yTranspuesta=Matrices.traspuesta(y);

    for(int i=1;i<=N;i++)
    {
        if(grafoHMM[I][i]==1)//si existe una conexion del estado I al estado i
            estados[i][1]=Util.log(A[I][i])+ S[i].b(yTranspuesta[0]);
    }

    double []temp=new double[N];//acumulador
    int cont; //cont para temp

    for(int t=2;t<=T;t++)
    {
        for(int j=1;j<=N;j++)
        {
            cont=0;
            for(int i=1;i<=N;i++)
            {   //     =log( alfa _i(t-1) ) + log (Aij) donde  S[i].alfaLog[t-1]=log( alfa _i(t-1) )
                if(grafoHMM[i][j]==1)//si existe una conexion del estado i al estado j
                {
                    if(S[i].alfaViterbiLog[t-1]!=0)
                    {
                        temp[cont]=S[i].alfaViterbiLog[t-1]+Util.log(A[i][j]);
                        cont++;
                    }
                }
            }
          //log( alfa_j(t) ) = log( sum_{i=1...N}(alfa _i(t-1)Aij) ) + log( b_j(y_t) )
          //  S[j].alfaViterbiLog[t]  = Util.max(temp)    +   S[j].b(yTranspuesta[t-1]);

            if(Util.sumLog(temp)!=0)
            {
                temp=Util.eliminarCeros(temp);
                estados[j][t]=Util.max(temp)    +   S[j].b(yTranspuesta[t-1]);
                if(Math.exp(estados[j][t])>1)
                    System.out.println("error en los alfas viterbi");
                //System.out.println(" probabilidad"+Math.exp(S[j].alfaLog[t]));
                temp=new double[N];//acumulador
            }
            temp=Matrices.resetear(temp);

        }
    }
    //TERMINACION
    //alfa_F(T)  ecu 9.9
    cont=0;
    for(int i=1;i<=N;i++)
    {
        if(grafoHMM[i][F]==1)//si existe una conexion del estado i al estado F
        {
            temp[cont]=S[i].alfaViterbiLog[T]+Util.log(A[i][F]);
            cont++;
        }
    }
    temp=Util.eliminarCeros(temp);
    return Util.max(temp);
}

public void resetear()
{
    for(int i=0;i<S.length;i++)
    {
        S[i].GammaLog=null;
        S[i].alfaLog=null;
        S[i].alfaViterbiLog=null;
        S[i].betaLog=null;
        S[i].gamma_jm_Log=null;
    }

}

public void algoritmoBaumWelch(Patron patronesE[])
{
    int N=S.length-2;       //numero de estados sin contar estado inicial ni final
    int I=0;                //estado inicial
    int F=S.length-1;       //estado final
    int E=patronesE.length; //numero de patrones
    int D=patronesE[0].getD();//dimension del patron

    //para cjm ec 9.44
    double [][][] cjmNumerador=new double[E][S.length][]; //E*(N+2)*M
    double [][] cjmDenominador=new double[E][S.length];   //E*(N+2)

    //para ujm ec 9.45
    double [][][][] ujmNumerador=new double[E][S.length][][]; //E*(N+2)*M*D
    double [][][] ujmDenominador=new double[E][S.length][];    //E*(N+2)*M
```

```
    //para Ejm ec 9.46
    double [][][][] EjmNumerador=new double[E][S.length][][]; //E*(N+2)*M*D
    double [][][] EjmDenominador=new double[E][S.length][];    //E*(N+2)*M mismo que ujmDenominador
    //para aij ec 9.24
    double [][][] aijNumerador=new double[E][A.length][A[0].length];
    double [][][] aijDenominador=new double[E][S.length][A[0].length];

    //para la sumatoria t=1:T
    for(int e=0;e<E;e++)
    {
        resetear();
         algoritmoForward(patronesE[e].datos);
         algoritmoBackward(patronesE[e].datos);
         algoritmoViterbi(patronesE[e].datos);
         getGamma_jm(patronesE[e].datos);
         getGammaLog(patronesE[e].datos);


         //actualizar cjm ec9.44
         //actualizar ujm ec9.45
         //actualizar Ejm ec9.46
         for(int j=1;j<=N;j++)
         {
             int M=S[j].getNroGaussianas();
             cjmNumerador[e][j] =new double[M];
             ujmNumerador[e][j]=new double[M][D];
             ujmDenominador[e][j]=new double[M];
             EjmNumerador[e][j]=new double[M][D];
             EjmDenominador[e][j]=new double[M];
             for(int m=0;m<M;m++)
             {
                 cjmNumerador[e][j][m]=numeradorLogCjm(patronesE[e].getDatos(), j, m);
                 ujmNumerador[e][j][m]=numeradorLogUjm(patronesE[e].getDatos(), j, m);
                 ujmDenominador[e][j][m]=denominadorLogUjm(patronesE[e].getDatos(), j, m);
                 EjmNumerador[e][j][m]=numeradorLogEjm(patronesE[e].getDatos(), j, m);
                 EjmDenominador[e][j][m]=denominadorLogEjm(patronesE[e].getDatos(), j, m);
             }
             cjmDenominador[e][j]=denominadorLogCjm(patronesE[e].getDatos(), j);
         }
         //fin de actualizar

         //actualizar aij
         //ec. 9.24 9.25 9.26 numerador y denominador
         int T=patronesE[e].datos[0].length;
         for(int i=0;i<S.length;i++)
         {
             for(int j=0;j<S.length;j++)
             {
                 if(this.grafoHMM[i][j]==1)
                 {
                     aijNumerador[e][i][j]=actualizarAijNumerador(patronesE[e].getDatos(), i, j);
//                     System.out.println("numerador A[i][j] = "+aijNumerador[e][i][j]);
                     aijDenominador[e][i][j]=actualizarAijDenominador(patronesE[e].getDatos(), i, j);
  //                   System.out.println("denominador A[i][j] = "+aijDenominador[e][i][j]);
                 }
             }
         }

    }
    /*punto de verificacion martes 08 de diciembre*/

    //para la sumatoria e=1:E
    for(int j=1;j<=N;j++)
    {
        int M=S[j].getNroGaussianas();
        double []tempDenominadorCjm=new double[E];
        for(int e=0;e<E;e++)
        {
            tempDenominadorCjm[e]=cjmDenominador[e][j];

        }
        for(int m=0;m<M;m++)
        {
            //para cjm
            double []tempNumeradorCjm = new double[E];


            double [][]tempNumeradorUjm= new double[E][D];
            double []tempDenominadorUjm=new double[E];

            double [][]tempNumeradorEjm= new double[E][D];
            double []tempDenominadorEjm=new double[E];

            for(int e=0;e<E;e++)
            {
                //para cjm
                tempNumeradorCjm[e]=cjmNumerador[e][j][m];

                //para ujm
                tempNumeradorUjm[e]=ujmNumerador[e][j][m];

                tempDenominadorUjm[e]=ujmDenominador[e][j][m];
                //para Ejm
                tempNumeradorEjm[e]=EjmNumerador[e][j][m];
                tempDenominadorEjm[e]=EjmDenominador[e][j][m];

            }

            //para cjm

            S[j].gmm.c[m]=Math.exp( Util.sumLog(tempNumeradorCjm)-Util.sumLog(tempDenominadorCjm));
```

```
            if(S[j].gmm.c[m]>1.0005)
                System.out.println("error en c "+S[j].gmm.c[m]);
            //para ujm
            double [] numeradorUjm = new double[D];
            double denominadorUjm=Util.sumLog(tempDenominadorUjm);
            tempNumeradorUjm=Matrices.traspuesta(tempNumeradorUjm);
            for(int d=0;d<D;d++)
                numeradorUjm[d]=Util.sumLog(tempNumeradorUjm[d]);

            double []mediaLog=Util.restar(numeradorUjm, denominadorUjm);
            S[j].gmm.N[m].setU(Util.exp(mediaLog));

            //para Ejm
            double [] numeradorEjm = new double[D];
            double denominadorEjm=Util.sumLog(tempDenominadorEjm);
            tempNumeradorEjm=Matrices.traspuesta(tempNumeradorEjm);
            for(int d=0;d<D;d++)
                numeradorEjm[d]=Util.sumLog(tempNumeradorEjm[d]);

            double []covariazaLog=Util.restar(numeradorEjm, denominadorEjm);
            S[j].gmm.N[m].setE(covariazaLog);

            /*resetear valore temporales*/
            Matrices.resetear(tempNumeradorCjm);
            Matrices.resetear(tempNumeradorUjm);
            Matrices.resetear(tempDenominadorUjm);
            Matrices.resetear(tempNumeradorEjm);
            Matrices.resetear(tempDenominadorEjm);
        }
        Matrices.resetear(tempDenominadorCjm);
    }

    double []tempNumeradorAij=new double[E];
    double []tempDenominadorAij=new double[E];
    //para aij

    /*punto de verificacion sabado 26 de diciembre*/
    //para el denominador

    for(int i=0;i<S.length;i++)
    {
        for(int j=0;j<S.length;j++)
        {
            if(this.grafoHMM[i][j]==1)
            {
                for(int e=0;e<E;e++)
                {
                    tempNumeradorAij[e]=aijNumerador[e][i][j];
                    tempDenominadorAij[e]=aijDenominador[e][i][j];
                }
                if(i==I)
                {
                    A[i][j]=Math.exp(Util.sumLog(tempNumeradorAij)-Util.log(E));
                    if(A[i][j]>1.005)
                    {
                        System.out.println("error 1");
                        System.out.println("valores "+Util.sumLog(tempNumeradorAij)+" "+Util.log(E)+" prob "+A[i][j]);
                    }
                    //para evitar mucho error numérico
                    if(A[i][j]>=0.99999999&&A[i][j]<=1.0000000009)
                        A[i][j]=1;

                    Matrices.resetear(tempNumeradorAij);
                }
                else
                {
                    A[i][j]=Math.exp(Util.sumLog(tempNumeradorAij)-Util.sumLog(tempDenominadorAij));
                    if(A[i][j]>1.005)
                    {
                        System.out.println("error 2");
                        System.out.println("valores "+Util.sumLog(tempNumeradorAij)+" "+Util.sumLog(tempDenominadorAij)+" prob "+A[i][j]);
                    }
                }
                //resetear vectores temporales
                Matrices.resetear(tempNumeradorAij);
                Matrices.resetear(tempDenominadorAij);
            }
        }
    }
}

/* computa todas los logaritmos de las probabilidades alfa_t(i), para cada uno de los estados
 */
public void algoritmoForward(double [][]y)
{
    int N=S.length-2;       //numero de estados sin contar estado inicial ni final
    int T=y[0].length;      //numero de frames T
    int I=0;                //estado inicial
    int F=S.length-1;       //estado final

    //trabajar con las traspuestas
    double[][]yTranspuesta=Matrices.traspuesta(y);

    /*INICIALIZACION*/
    //se inicializa T=1
    //tener en cuenta yTranspuesta[0] es Y en el tiempo t=1
    for(int j=1;j<=N;j++){
        //inicializa el vector alfa, tener en cuenta que alfa[0] no existe y alfa[T]es alfaF(T)
        S[j].setAlfaLog(T+2);
        //log (alfa_i(1)) =log(A[I=0][j])             +        log (b_i(Y_1))  ec 9.8
        if(this.grafoHMM[I][j]==1)//si existe una conexion del estado I al estado j
```

```
            S[j].alfaLog[1]   =Util.log(A[I][j])+ S[j].b(yTranspuesta[0]);
        }

        /*INDUCCION*/
        //para t=2 hasta t=T
        //ec 9.7
        //alfa_j(t)=sum_{i=1...N}(alfa _i(t-1)Aij)b_j(y_t)
        //log( alfa_j(t) ) = log( sum_{i=1...N}(alfa _i(t-1)Aij) ) + log( b_j(y_t) )

        double []temp=new double[N];//acumulador
        int cont; //cont para temp

        for(int t=2;t<=T;t++){
            for(int j=1;j<=N;j++){
                cont=0;
                for(int i=1;i<=N;i++){
                   //     =log( alfa _i(t-1) ) + log (Aij) donde  S[i].alfaLog[t-1]=log( alfa _i(t-1) )
                    if(this.grafoHMM[i][j]==1)//si existe una conexion del estado i al estado j
                    {
                        if(S[i].alfaLog[t-1]!=0)
                        {
                            temp[cont]=S[i].alfaLog[t-1]+Util.log(A[i][j]);
                            cont++;
                        }
                    }
                }
              //log( alfa_j(t) ) = log( sum_{i=1...N}(alfa _i(t-1)Aij) ) + log( b_j(y_t) )
                double suma=Util.sumLog(temp);
                if(suma!=0)
                {
                    S[j].alfaLog[t]  =suma    +   S[j].b(yTranspuesta[t-1]);
                    if(Math.exp(S[j].alfaLog[t])>1)
                        System.out.println("error en los alfas");
                    //System.out.println(" probabilidad"+Math.exp(S[j].alfaLog[t]));
                }
                temp=Matrices.resetear(temp);
            }
        }
        //TERMINACION
        //alfa_F(T)  ecu 9.9
        cont=0;
        for(int i=1;i<=N;i++)
        {
            if(this.grafoHMM[i][F]==1)//si existe una conexion del estado i al estado F
            {
                temp[cont]=S[i].alfaLog[T]+Util.log(A[i][F]);
                cont++;
            }
        }
        S[F].setAlfaLog(T+2);
        S[F].alfaLog[T+1]=Util.sumLog(temp);
    }

    public void algoritmoBackward(double [][]y)
    {
        int N=S.length-2;       //numero de estados sin contar estado inicial ni final
        int T=y[0].length;      //numero de frames T
        int I=0;                //estado inicial
        int F=S.length-1;       //estado final
         //trabajar con las traspuestas
        double[][]yTranspuesta=Matrices.traspuesta(y);

        for(int i=1;i<=N;i++)
        {
            S[i].setBetaLog(T+2);
            if(grafoHMM[i][F]==1)
                S[i].betaLog[T]=Util.log(A[i][F]);  //ec 9.16
        }

        double []temp=new double[N];//acumulador
        int cont; //cont para temp

        for (int t=T-1; t>=1;t--)
        {
            for(int i=1;i<=N;i++)
            {
                cont=0;
                for(int j=1;j<=N;j++){
                    if(grafoHMM[i][j]==1)
                    {
                        if(S[j].betaLog[t+1]!=0)
                        {
                            temp[cont]=Util.log(A[i][j]) + S[j].b(yTranspuesta[t]) + S[j].betaLog[t+1];
                            cont++;
                        }
                    }
                }

                double suma=Util.sumLog(temp);
                if(suma!=0)
                {
                    S[i].betaLog[t]  = suma;
                    if(Math.exp(S[i].betaLog[t])>1)
                        System.out.println("error en los betas"+Math.exp(S[i].betaLog[t]));
                    //System.out.println(" probabilidad"+Math.exp(S[j].alfaLog[t]));
                }
                temp=Matrices.resetear(temp);
            }
        }
    }
```

```
public void algoritmoViterbi(double [][]y)
{
    int N=S.length-2;       //numero de estados sin contar estado inicial ni final
    int T=y[0].length;      //numero de frames T
    int I=0;                //estado inicial
    int F=S.length-1;       //estado final

    //trabajar con las traspuestas
    double[][]yTranspuesta=Matrices.traspuesta(y);

    /*INICIALIZACION*/
    //se inicializa T=1
    //tener en cuenta yTranspuesta[0] es Y en el tiempo t=1
    for(int i=1;i<=N;i++)
    {
        //inicializa el vector alfa, tener en cuenta que alfa[0] no existe y alfa[T]es alfaF(T)
        S[i].setAlfaViterbiLog(T+2);
        //log (alfa_i(1)) =log(A[I=0][j])             +        log (b_i(Y_1))  ec 9.8
        if(grafoHMM[I][i]==1)//si existe una conexion del estado I al estado i
            S[i].alfaViterbiLog[1]   =Util.log(A[I][i])+ S[i].b(yTranspuesta[0]);

    }

    /*INDUCCION*/
    //para t=2 hasta t=T
    //ec 9.7
    //alfa_j(t)=sum_{i=1...N}(alfa _i(t-1)Aij)b_j(y_t)
    //log( alfa_j(t) ) = log( sum_{i=1...N}(alfa _i(t-1)Aij) ) + log( b_j(y_t) )

    double []temp=new double[N];//acumulador
    int cont; //cont para temp


    for(int t=2;t<=T;t++)
    {
        for(int j=1;j<=N;j++)
        {
            cont=0;
            for(int i=1;i<=N;i++)
            {   //     =log( alfa _i(t-1) ) + log (Aij) donde  S[i].alfaLog[t-1]=log( alfa _i(t-1) )
                if(grafoHMM[i][j]==1) //si existe una conexion del estado i al estado j
                {
                    if(S[i].alfaViterbiLog[t-1]!=0)
                    {
                        temp[cont]=S[i].alfaViterbiLog[t-1]+Util.log(A[i][j]);
                        cont++;
                    }
                }
            }
          //log( alfa_j(t) ) = log( sum_{i=1...N}(alfa _i(t-1)Aij) ) + log( b_j(y_t) )

            if(Util.sumLog(temp)!=0)
            {
                temp=Util.eliminarCeros(temp);
                S[j].alfaViterbiLog[t]  = Util.max(temp)    +   S[j].b(yTranspuesta[t-1]);
                if(Math.exp(S[j].alfaViterbiLog[t])>1)
                    System.out.println("error en los alfas viterbi");
                //System.out.println(" probabilidad"+Math.exp(S[j].alfaLog[t]));
                temp=new double[N];//acumulador
            }
            temp=Matrices.resetear(temp);

        }
    }
    //TERMINACION
    //alfa_F(T)  ecu 9.9
    cont=0;
    for(int i=1;i<=N;i++)
    {
        if(grafoHMM[i][F]==1)// si existe una conexion del estado i al estado F
        {
            temp[cont]=S[i].alfaViterbiLog[T]+Util.log(A[i][F]);
            cont++;
        }
    }
    S[S.length-1].setAlfaViterbiLog(T+2);
    temp=Util.eliminarCeros(temp);
    S[S.length-1].alfaViterbiLog[T+1]=Util.max(temp);
}

 //implementacion de ec 9.19
 public void getGamma_jm(double y[][]){
    int N=S.length-2;       //numero de estados sin contar estado inicial ni final
    int T=y[0].length;      //numero de frames T
    int I=0;                //estado inicial
    int F=S.length-1;       //estado final

    //trabajar con las traspuestas
    double[][]yTranspuesta=Matrices.traspuesta(y);

    //inicializar
    for(int i=1;i<=N;i++){
        S[i].setGamma_ij_Log(S[i].getNroGaussianas(),T+2);
    }

    //implementacion ec 9.43
    double []temp=new double[N];//acumulador
    int cont; //cont para temp
    double alfaF=S[F].alfaLog[T+1];//probabilidad forward
    int M; //numero de gaussianas por estado
```

```
   for(int t=1;t<=T;t++){     //desde t=2, pues no hay manera ce calcular en t=1 pues en t=0 no existen alfas

       //para t=1, usar formula deducida y que esta en la tesis para gamma_{jm}(t=1)
       if(t==1)
       {
           for(int j=1;j<=N;j++)
           {
               M=S[j].getNroGaussianas();// cada estado puede tener un numero diferente de gaussianas
               if(grafoHMM[I][j]==1)
               {
                   for(int m=0;m<M;m++)
                   {
                       if(S[j].betaLog[t]!=0)
                       {
                           S[j].gamma_jm_Log[m][t]= Util.log(A[I][j])+Util.log(S[j].getGmm().c[m])+S[j].getGmm().N[m].logP(yTranspuesta[t-1])
                           if(Math.exp(S[j].gamma_jm_Log[m][t])>1.0005)// se verifica si la probabilidad no rebasa a 1 con un error de 0.0005
                               System.out.println("ERROR probabilidad en gamma jm"+Math.exp(S[j].gamma_jm_Log[m][t]));
                       }
                   }
               }
           }
       }
       else
       {
           for(int j=1;j<=N;j++)
           {
               M=S[j].getNroGaussianas();// cada estado puede tener un numero diferente de gaussianas
               for(int m=0;m<M;m++)
               {
                   cont=0;
                   for(int i=1;i<=N;i++){//para el numeradorCjm de la ec 9.43
                       if(grafoHMM[i][j]==1)
                       {
                           //validaciones para que no haya ceros
                           if(S[i].alfaLog[t-1]!=0&&S[j].betaLog[t]!=0)
                           {
                         //      temp[cont]=S[i].alfaLog[t-1]+Util.log(A[i][j])+Util.log(S[j].getGmm().c[m])+S[j].getGmm().N[m].logP(yTranspu
                               temp[cont]=S[i].alfaLog[t-1]+Util.log(A[i][j]);
                               cont++;
                           }
                       }
                   }
                   double suma=Util.sumLog(temp);
                   if(suma!=0)//validacion para evitar sumar los ceros en el dominio de los logaritmos
                   {
                       S[j].gamma_jm_Log[m][t]=suma+Util.log(S[j].getGmm().c[m])+S[j].getGmm().N[m].logP(yTranspuesta[t-1])+S[j].betaLog[t]-a
                       if(Math.exp(S[j].gamma_jm_Log[m][t])>1.0005)// se verifica si la probabilidad no rebasa a 1 con un error de 0.0005
                           System.out.println("ERROR probabilidad en gamma jm "+Math.exp(S[j].gamma_jm_Log[m][t]));
                       Matrices.resetear(temp);
                   }

               }
           }
       }
   }
}

public void getGammaLog(double y [][])
{
   int N=S.length-2;       //numero de estados sin contar estado inicial ni final
   int T=y[0].length;      //numero de frames T
   int I=0;                //estado inicial
   int F=S.length-1;       //estado final

           //inicializar
   for(int i=1;i<=N;i++){
       S[i].setGammaLog(T+2);
   }

   //implementacion ec 9.43
   double []temp=new double[N];//acumulador
   int cont; //cont para temp
   double alfaF=S[F].alfaLog[T+1];//probabilidad forward
   int M; //numero de gaussianas por estado

   for(int t=1;t<=T;t++){
       for(int j=1;j<=N;j++){
           if(S[j].alfaLog[t]!=0 && S[j].betaLog[t]!=0)   //validacion
           {
               S[j].GammaLog[t]=S[j].alfaLog[t]+S[j].betaLog[t]-alfaF;
               if(Math.exp(S[j].GammaLog[t])>1.0005)
                   System.out.println("error en gamma "+Math.exp(S[j].GammaLog[t])+" S[j].alfaLog[t]  "+S[j].alfaLog[t]+"S[j].betaLog[t] "+S[
           }
       }
   }

   for(int t=1;t<=T;t++)
    {
        for(int i=1;i<=N;i++)
        {
            double[][] transpuesta=Matrices.traspuesta(S[i].gamma_jm_Log);
            S[i].GammaLog[t]=Util.sumLog(transpuesta[t]);
        }
    }
}


//implementacion de ec 9.44 (no usar)
//para actualizar cjm, entra como argumento, los patrones, el estado Sj, y la mixtura m
public double actualizarCij(Patron patronesE[], int j, int m)
{
```

```
        int E=patronesE.length; //numero de patrones de entrenamiento
        int N=S.length-2;       //numero de estados sin contar estado inicial ni final
        int I=0;                //estado inicial
        int F=S.length-1;       //estado final

        int T;      //numero de frames T
        double []temENumerador =new double[E];// los vectores temporales temT* y temE* sirven para acumular los logProb y luego sumarlos
        double []temEDenominador =new double[E];
        for(int e=0;e<E;e++)    //para todos los patrones de entrenamiento
        {
            /*para el numeradorCjm*/
            T=patronesE[e].getT();  //cada patron tiene tamaño diferente
            double []temT=new double[T];


            for(int t=2;t<=T;t++)
                temT[t]=S[j].gamma_jm_Log[m][t];

            temENumerador[e]=Util.sumLog(temT);

            //resetear el vector
            temT=Matrices.resetear(temT);
            /*para el denominadorCjm*/
            for(int t=2;t<=T;t++)
                temT[t]=S[j].GammaLog[t];

            temEDenominador[e]=Util.sumLog(temT);
        }
        return Util.sumLog(temENumerador)-Util.sumLog(temEDenominador);
    }

/*************************************************************/
    //implementacion de ec 9.44
    //para actualizar cjm, entra como argumento, el vector de observaciones Y, el estado Sj, y la mixtura m
    //solo actualiza para la sumatoria t=1:T, no tiene en cuenta e=1:E
    //solo actualiza el numeradorCjm de la formula
    public double numeradorLogCjm(double [][] y, int j, int m)
    {
        int I=0;                //estado inicial
        int F=S.length-1;       //estado final

        int T=y[0].length;      //numero de frames T
        /*para el numeradorCjm*/
        double []temT=new double[T];

        for(int t=1;t<=T;t++)
            temT[t-1]=S[j].gamma_jm_Log[m][t];

        return Util.sumLog(temT);

    }

    /*para el denominadorCjm*/

    public double denominadorLogCjm(double [][] y, int j)
    {
        int N=S.length-2;       //numero de estados sin contar estado inicial ni final
        int I=0;                //estado inicial
        int F=S.length-1;       //estado final

        int T=y[0].length;      //numero de frames T
        double []temT=new double[T];

        for(int t=1;t<=T;t++)
            temT[t-1]=S[j].GammaLog[t];

        return Util.sumLog(temT);

    }

/*************************************************************/

    //implementacion de ec 9.45 (no usar)
    //para actualizar Ujm, entra como argumento, los patrones, el estado Sj, y la mixtura m
    public double[] actualizarUjm(Patron patronesE[], int j, int m)
    {
        int E=patronesE.length; //numero de patrones de entrenamiento
        int N=S.length-2;       //numero de estados sin contar estado inicial ni final
        int I=0;                //estado inicial
        int F=S.length-1;       //estado final

        int T;      //numero de frames T
        int D;      //dimension del patron
        D=patronesE[0].getD();  //dimension del patron
        double []numerador =new double[D];// los vectores temporales temT* y temE* sirven para acumular los logProb y luego sumarlos
        double []temEDenominador =new double[E];

        double [][] temE = new double[D][E];

        for(int e=0;e<E;e++)
        {
            /*NUMERADOR*/
            T=patronesE[e].getT();  //cada patron tiene tamaño diferente

            //trabajar con las traspuestas
            //obtener los datos del patron y convertirlo en transpuesta
            double[][]yTranspuesta=Matrices.traspuesta(patronesE[e].getDatos());
            //temporal para trabajar con yTranspuesta
            //acumulara los valores parciales de la primera parte de la sumatoria
            double [][]temT=new double[T][D];
            //termino de la sumatoria t=1:T
```

```
            for(int t=2;t<=T;t++)
            {
                temT[t]=Util.sumar(S[j].gamma_jm_Log[m][t], yTranspuesta[t-1]);
            }

            //sumar logProb
            //tomaremos la traspuesta de tempT, para hacer mas facil la suma
            //y acumularemos los valores parciales en tempE
            temT=Matrices.traspuesta(temT);
            for(int d=0;d<D;d++)
            {
                temE[d][e]=Util.sumLog(temT[d]);
            }
            Matrices.resetear(temT);
            /*FIN NUMERADOR*/

            /*DENOMINADOR*/
            double []temp=new double[T];
            for(int t=2;t<=T;t++)
            {
                temp[t]=S[j].gamma_jm_Log[m][t];
            }

            temEDenominador[e]=Util.sumLog(temp);
            Matrices.resetear(temp);

        }

        //cacular el vector correspondiente al denominadorCjm esto se hace
        // resolviendo el termino de la sumatoria e=1:E
        //para el numeradorCjm
        for(int d=0;d<D;d++)
        {
            numerador[d]=Util.sumLog(temE[d]);
        }
        Matrices.resetear(temE);
        //para el denominadorCjm
        double denominador=Util.sumLog(temEDenominador);


        double [] Ujm=new double[D];
        Ujm=Util.restar(numerador, denominador);

        return Util.exp(Ujm);

    }

    /*****************************************************************************************/
    //implementacion de ec 9.45
    //para actualizar Ujm, entra como argumento,  el vector de observaciones Y, el estado Sj, y la mixtura m
    //solo actualiza para la sumatoria t=1:T, no tiene en cuenta e=1:E
    //solo actualiza el numeradorCjm de la formula
    public double[] numeradorLogUjm(double y[][], int j, int m)
    {
        int N=S.length-2;       //numero de estados sin contar estado inicial ni final
        int I=0;                //estado inicial
        int F=S.length-1;       //estado final
        int T=y[0].length;      //numero de frames T
        int D=y.length;      //dimension del patron

        //trabajar con las traspuestas
        double[][]yTranspuesta=Matrices.traspuesta(y);
        //temporal para trabajar con yTranspuesta
        //acumulara los valores parciales de la primera parte de la sumatoria
        double [][]temT=new double[T][D];
        //termino de la sumatoria t=1:T
        for(int t=1;t<=T;t++)
            temT[t-1]=Util.sumar(S[j].gamma_jm_Log[m][t], Util.log(yTranspuesta[t-1]));

        //sumar logProb
        //tomaremos la traspuesta de tempT, para hacer mas facil la suma
        double []mediaLogNumerador = new double[D];
        temT=Matrices.traspuesta(temT);

        for(int d=0;d<D;d++)
            mediaLogNumerador[d]=Util.sumLog(temT[d]);

        return mediaLogNumerador;

    }
    //para el denominadorEjm
    public double denominadorLogUjm(double [][]y,int j, int m)
    {
        int N=S.length-2;       //numero de estados sin contar estado inicial ni final
        int I=0;                //estado inicial
        int F=S.length-1;       //estado final
        int T=y[0].length;      //numero de frames T

        double []temp=new double[T];

        for(int t=1;t<=T;t++)
            temp[t-1]=S[j].gamma_jm_Log[m][t];

        return Util.sumLog(temp);

    }

    /*****************************************************************************************/


     /*****************************************************************************************/
```

```
    //implementacion de ec 9.46
    //para actualizar Ujm, entra como argumento,  el vector de observaciones Y, el estado Sj, y la mixtura m
    //solo actualiza para la sumatoria t=1:T, no tiene en cuenta e=1:E
    //solo actualiza el numeradorCjm de la formula
    public double[] numeradorLogEjm(double y[][], int j, int m)
    {
        int N=S.length-2;       //numero de estados sin contar estado inicial ni final
        int I=0;                //estado inicial
        int F=S.length-1;       //estado final
        int T=y[0].length;      //numero de frames T
        int D=y.length;      //dimension del patron

        //trabajar con las traspuestas
        double[][]yTranspuesta=Matrices.traspuesta(y);
        //temporal para trabajar con yTranspuesta
        //acumulara los valores parciales de la primera parte de la sumatoria
        double [][]temT=new double[T][D];
        //termino de la sumatoria t=1:T
        for(int t=1;t<=T;t++)
        {   //(y-u)
            yTranspuesta[t-1] =Util.restarAbs(yTranspuesta[t-1], S[j].gmm.N[m].getU());
            //2*log(y-u)
            yTranspuesta[t-1]=Util.multiplicar(2, Util.log(yTranspuesta[t-1]));
             //log(gama)+2*log(y-u)
            temT[t-1]=Util.sumar(S[j].gamma_jm_Log[m][t], yTranspuesta[t-1]);
        }

        //sumar logProb
        //tomaremos la traspuesta de tempT, para hacer mas facil la suma
        double []covariazaLogNumerador = new double[D];
        temT=Matrices.traspuesta(temT);

        for(int d=0;d<D;d++)
            covariazaLogNumerador[d]=Util.sumLog(temT[d]);

        return covariazaLogNumerador;

    }

    //para el denominadorEjm
    public double denominadorLogEjm(double [][]y,int j, int m)
    {
        int N=S.length-2;       //numero de estados sin contar estado inicial ni final
        int I=0;                //estado inicial
        int F=S.length-1;       //estado final
        int T=y[0].length;      //numero de frames T

        double []temp=new double[T];

        for(int t=1;t<=T;t++)
            temp[t-1]=S[j].gamma_jm_Log[m][t];

        return Util.sumLog(temp);


    }


    /*****************************************************************************************/

    //implementacion de ec 9.46 (no usar)
    //para actualizar Ejm, entra como argumento, los patrones, el estado Sj, y la mixtura m

/*     public double[] actualizarEjm(Patron patronesE[], int j, int m)
    {
        int E=patronesE.length; //numero de patrones de entrenamiento
        int N=S.length-2;       //numero de estados sin contar estado inicial ni final
        int I=0;                //estado inicial
        int F=S.length-1;       //estado final

        int T;      //numero de frames T
        int D;      //dimension del patron
        D=patronesE[0].getD();  //dimension del patron
        double []numerador =new double[D];// los vectores temporales temT* y temE* sirven para acumular los logProb y luego sumarlos
        double []temEDenominador =new double[E];

        double [][] temE = new double[D][E];

        for(int e=0;e<E;e++)
        {
            /*NUMERADOR*/
 /*           T=patronesE[e].getT();  //cada patron tiene tamaño diferente

            //trabajar con las traspuestas
            //obtener los datos del patron y convertirlo en transpuesta
            double[][]yTranspuesta=Matrices.traspuesta(patronesE[e].getDatos());
            //temporal para trabajar con yTranspuesta
            //acumulara los valores parciales de la primera parte de la sumatoria
            double [][]temT=new double[T][D];
            //termino de la sumatoria t=1:T
            for(int t=2;t<=T;t++)
            {
                //2*log(y_te-u_jm)
                yTranspuesta[t-1]=Util.multiplicar(2, Util.restar(yTranspuesta[t-1], S[j].getGmm().N[m].u));

                temT[t]=Util.sumar(S[j].gamma_jm_Log[m][t], yTranspuesta[t-1]);
            }

            //sumar logProb
            //tomaremos la traspuesta de tempT, para hacer mas facil la suma
            //y acumularemos los valores parciales en tempE
```

```
            temT=Matrices.traspuesta(temT);
            for(int d=0;d<D;d++)
            {
                temE[d][e]=Util.sumLog(temT[d]);
            }
            /*FIN NUMERADOR*/

            /*DENOMINADOR*/
  /*          double []temp=new double[T];
            for(int t=2;t<=T;t++)
            {
                temp[t]=S[j].gamma_jm_Log[m][t];
            }

            temEDenominador[e]=Util.sumLog(temp);

        }

        //cacular el vector correspondiente al denominadorCjm esto se hace
        // resolviendo el termino de la sumatoria e=1:E
        //para el numeradorCjm
        for(int d=0;d<D;d++)
        {
            numerador[d]=Util.sumLog(temE[d]);
        }
        //para el denominadorCjm
        double denominador=Util.sumLog(temEDenominador);


        double [] Ejm=new double[D];
        Ejm=Util.restar(numerador, denominador);

        return Util.exp(Ejm);

    }
*/
    //ec 9.21
    public double epsilonLog(int i, int j,int t,double [][]y)
    {
        int F=S.length-1; //estado final
        int I=0;          //estado inicial
        double [][]yTranspuesta=Matrices.traspuesta(y);
        int T=y[0].length;
        double alfaF=S[F].alfaLog[T+1];//probabilidad forward
        //ec 9.22
        if (j==F)
            if(S[i].alfaLog[T]!=0)//validaciones
            {
                double result=S[i].alfaLog[T]+Util.log(A[i][F])-alfaF;
                if (result==0)
                    result=0.000000000000001;
                return result;

            }
            else
                return 0;

        //ec 9.23

        if (i==I)
            if(S[j].betaLog[1]!=0)//validaciones
            {
                double result=Util.log(A[I][j])+S[j].b(yTranspuesta[0])+S[j].betaLog[1]-alfaF;
                if (result==0)
                    result=0.000000000000001;
                return result;
            }

            else
                return 0;

        else
        {
       //ec 9.21 y 9.24 para el numerador devuelve no solo el epsilon si no toda la sumatoria de epsilon
      //      double []temp= new double[T-1];
      //      for(int t=1;t<=(T-1);t++)
      //      {
      //          temp[t]=S[i].alfaLog[t]+Util.log(A[i][j])+S[j].b(yTranspuesta[t])+S[j].betaLog[t+1]-alfaF;
      //      }
      //      return Util.sumLog(temp);
            if(S[i].alfaLog[t]!=0&&S[j].betaLog[t+1]!=0)
            {

              /*  System.out.println("S[i].alfaLog[t]"+S[i].alfaLog[t]);
                System.out.println("A[i][j] "+A[i][j]+" logA[i][j] "+Util.log(A[i][j]));
                System.out.println("b(yTranspuesta[0]) "+S[j].b(yTranspuesta[t]));
                System.out.println("S[j].betaLog[t+1] "+S[j].betaLog[t+1]+" alfaF "+alfaF);
                System.out.println("result "+(S[i].alfaLog[t]+Util.log(A[i][j])+S[j].b(yTranspuesta[t])+S[j].betaLog[t+1]-alfaF));
            */  double result =S[i].alfaLog[t]+Util.log(A[i][j])+S[j].b(yTranspuesta[t])+S[j].betaLog[t+1]-alfaF;
                if (result==0)
                    result=0.000000000000001;
                return result;
            }

            else
                return 0;
        }
    }

    public double actualizarAijNumerador(double [][]y,int i, int j)
    {
```

```
    int N=S.length-2;       //numero de estados sin contar estado inicial ni final
    int F=S.length-1; //estado final
    int I=0;          //estado inicial
    int T=y[0].length;
    double temp[] =new double[T-1];

    //ecuacion 9.25 parte del numerador
    if(j==F)
    {
        return epsilonLog(i, j, T,y);
    }
    //ecuacion 9.26 parte del numerador
    else if(i==I)
    {
        return epsilonLog(i, j, 0,y);
    }
    //ecuacion 9.24 parte del numerador
    else
    {
        for(int t=1;t<=(T-1);t++)
        {
            temp[t-1]=epsilonLog(i, j, t,y);
        }
        double result=Util.sumLog(temp);
        System.out.println("\n result "+result);
        System.out.println("");
        return result;
    }
}

public double actualizarAijDenominador(double [][]y,int i, int j)
{
    int N=S.length-2;       //numero de estados sin contar estado inicial ni final
    int F=S.length-1; //estado final
    int I=0;          //estado inicial
    int T=y[0].length;
    double temp[] =new double[T];

    //ecuacion 9.25 parte del denominador
    if(j==F)
    {
        for(int t=1;t<=T;t++)
        {
            temp[t-1]=S[i].GammaLog[t];
        }
        return Util.sumLog(temp);
    }
    //ecuacion 9.26 parte del denominador
    else if(i==I)
    {
        System.out.println("result 1");
        return 1;
    }
    //ecuacion 9.24 parte del denominador
    else
    {
        for(int t=1;t<=T;t++)
        {
            temp[t-1]=S[i].GammaLog[t];
        }
        return Util.sumLog(temp);
    }
}


public void verificar()
{
    int N=S.length-2;
    int T=S[1].alfaLog.length-2;
    double []temp = new double[N];
    double []suma=new double [T];
    int cont;

    System.out.println("verifica que la suma de S[i].alfaLog[t]+S[i].betaLog[t]  en cada instante de tiempo sea la misma");
    for(int t=1;t<=T;t++){
        cont=0;
        for(int i=1;i<=N;i++){
            if(S[i].alfaLog[t]!=0&&S[i].betaLog[t]!=0)
            {
                temp[cont]=S[i].alfaLog[t]+S[i].betaLog[t];
                cont++;
            }
        }
        suma[t-1]=Util.sumLog(temp);
        System.out.println(" suma "+suma[t-1]);
        Matrices.resetear(temp);
    }

    System.out.println("ver funciones getGammaLog y getGamma_jm");

    for(int t=1;t<=T;t++)
    {
        for(int i=1;i<=N;i++)
        {
            double[][] transpuesta=Matrices.traspuesta(S[i].gamma_jm_Log);
            System.out.println(" gamma_jm_Log"+Util.sumLog(transpuesta[t]));
            System.out.println("gammaLog"+S[i].GammaLog[t]);

        }
        System.out.println("");
    }
```

```
 }


public void mostrar()
{
    System.out.println("");
    System.out.println("PARAMETROS DEL HMM");
    System.out.println("S ");
    System.out.println("---------------");
    System.out.println(""+S[0].getId());
    System.out.println("---------------");
    for(int i=1;i<S.length-1;i++)
    {
        System.out.println("---------------");
        System.out.println(""+S[i].getId());
        System.out.println("---------------");
        System.out.println("observaciones");
     //   Matrices.mostrar(S[i].getObservaciones());
        S[i].getGmm().mostrar();
        System.out.println("---------------");
    }
    System.out.println("---------------");
    System.out.println(""+S[S.length-1].getId());
    System.out.println("---------------");
     System.out.println("A ");
     Matrices.mostrar(A);
     System.out.println("---------------");
     System.out.println("PI ");
     for(int i=0;i<S.length;i++)
         System.out.print(" "+S[i].getPi());
     System.out.println("");
     System.out.println("---------------");
     System.out.println("alfas");
     for(int i=1;i<S.length;i++)
     {
             System.out.print(" "+S[i].getId());
             if(S[i].getAlfaLog()==null)
                 System.out.println("null");
             else{
                 for(int j=0;j<S[i].getAlfaLog().length;j++)
                     System.out.print("   "+S[i].alfaLog[j]);
             }
             System.out.println(" ");
     }

     System.out.println("betas");
     for(int i=1;i<S.length-1;i++)
     {
             System.out.print(" "+S[i].getId());
             if(S[i].getBetaLog()==null)
                 System.out.println("null");
             else{
                 for(int j=0;j<S[i].getBetaLog().length;j++)
                     System.out.print("   "+S[i].betaLog[j]);
             }
             System.out.println(" ");
     }
     System.out.println("alfas viterbi");
     for(int i=1;i<S.length;i++)
     {
             System.out.print(" "+S[i].getId());
             if(S[i].getAlfaViterbiLog()==null)
                 System.out.println("null");
             else{
                 for(int j=0;j<S[i].getAlfaViterbiLog().length;j++)
                     System.out.print("   "+S[i].alfaViterbiLog[j]);
             }
             System.out.println(" ");
     }
     System.out.println("gamma jm");
     for(int i=1;i<S.length-1;i++)
     {
             System.out.println(" "+S[i].getId());
             if(S[i].getGamma_jm_Log()==null)
                 System.out.println("null");
             else
                 Matrices.mostrar(S[i].getGamma_jm_Log());
             System.out.println(" ");
     }
     System.out.println("GammaLog");
     for(int i=1;i<S.length-1;i++)
     {
             System.out.println(" "+S[i].getId());
             if(S[i].getGammaLog()==null)
                 System.out.println("null");
             else{
                 for(int j=0;j<S[i].getGammaLog().length;j++)
                     System.out.print("  "+S[i].GammaLog[j]);
             }
             System.out.println(" ");
     }

     System.out.println("");
     System.out.println("---------------");
}
```

```
    public void setS(Estado[] S) {
        this.S = S;
    }

    public void setA(double[][] A) {
        this.A = A;
    }

    public Estado[] getS() {
        return S;
    }

    public double[][] getA() {
        return A;
    }
}
```

A.7.3. Código de los Modelos de Mezclas de Gaussianas

```
package modeloocultodemarkov;

/**
 *
 * @author usuario
 */
public class GMM {

    public double []c;//coeficiente de las gaussianas
    public Gaussiana [] N;//varias gaussianas

    /*CONSTRUCTORES*/
    public  GMM(double [][] Y, int nroGaussianas)
    {
        c=new double[nroGaussianas];
        N=new Gaussiana[nroGaussianas];
        inicializacion(Y,nroGaussianas);
    }

    //este constructor solo sirve para inicializar manualmente sin usar kMeans
    public  GMM(double [][] Y, int nroGaussianas,int valor)
    {
        c=new double[nroGaussianas];
        N=new Gaussiana[nroGaussianas];
    }

    public GMM(double[] c, Gaussiana[] N) {
        this.c = c;
        this.N = N;
    }

    /*GETTER AND SETTER*/
    public Gaussiana[] getN() {
        return N;
    }

    public double[] getC() {
        return c;
    }

    public void setN(Gaussiana[] N) {
        this.N = N;
    }

    public void setC(double[] c) {
        this.c = c;
    }

    /*METODOS*/


    //inicializa los pararmetros u y E de las gaussianas de GMM
    private void inicializacion(double [][] Y,int nroGaussianas)
    {
        Cluster [] cluster = Algoritmos.kMeans(Y, nroGaussianas);
        System.out.println("Mixturas Gaussianas Inicializadas");
        for(int i=0;i<nroGaussianas;i++)
        {
            N[i]=new Gaussiana(cluster[i].getMedia(),cluster[i].getMatrizCoovarianzaDiagonal());
            c[i]=(double)cluster[i].getNumeroDatosPorCluster()/(double)Y[0].length;
            System.out.print("c["+i+"] "+c[i]);
        }
        System.out.println("");

    }

   public double getLogProb(double [] y)
    {
       double []temp=new double[N.length];
       for(int i=0;i<N.length;i++)
           temp[i]=N[i].logP(y)+Util.log(c[i]);

       return Util.sumLog(temp);
    }

    public double [] getCoeficientes()
    {
        return c;
    }
```

```
    public Gaussiana[] getGaussianas()
    {
        return N;
    }

    public void mostrar()
    {
        for(int i=0;i<N.length;i++)
        {
            System.out.println("Gaussianas "+i);
            System.out.println("Media ");
            Matrices.mostrar(N[i].getU());
            System.out.println("Coovarianza ");
            Matrices.mostrar(N[i].getE());
        }
           System.out.println("Coeficientes ");
        Matrices.mostrar(c);
    }


}

package modeloocultodemarkov;

/**
 *
 * @author usuarioGaussiana
 */
public class Gaussiana {

    double []u; //media
    double []E;//matriz de coovarianza diagonal

    //establece la media para la gaussiana
    //ingresan Y=vectores de observacion
    //speech recognition and synthesis ec 9.34
    public Gaussiana(double [][]Y)
    {
        u=Matrices.promedio(Y);
        E=Matrices.coovarianzaDiagonal(Y, u);
    }

    public Gaussiana(double []u, double []E)
    {
        this.u=u;
        this.E=E;
    }

    //ec 9.53
    //retorna el logaritmo de la probabilidad de la observacion

    public double logP(double[] y)
    {

        double k=u.length;
        double var1=0;
        double var2=0;

        for(int i=0;i<k;i++)
        {
            var1=var1+Util.log(E[i]);
            var2=var2+Math.pow((y[i]-u[i])/E[i], 2);

        }

        return -k/2*Util.log(2*Math.PI)-var1-1.0/2.0*var2;
    }


    //establece la media de la observacion en cierto estado
    //speech recognition and synthesis ec 9.34
    public void setU(double []u)
    {
        this.u=u;
    }

    public void setE(double []E)
    {
        this.E=E;
    }

    public double[] getU()
    {
        return u;
    }
    public double[] getE()
    {
        return E;
    }


}
```

A.7.4. Código de los Coeficientes Cepstrales en Escala mel MFCC

A continuación el código fuente de los Coeficientes Cepstrales en Escala mel MFCC en Java

```
package MFCC;

public class MFCC
{
    int frecuenciaDeSampleo ; //frecuencia de sampleo
    double[] arreglo;
    int nBins;

    double [] arregloPreenfasis; //arreglo despues del preenfasis
    double[][] matrizVentaneamiento;//matriz conteniendo los valores del ventaneamiento
    double [][]matrizFourier;//matriz con el módulo de la transformada de fourier
    double [][]matrizMFCC;//matriz con los valores MFCC
    double [][]MFCCDelta;//MFCC + delta
    double [][]MFCCDeltaDelta;//MFCC+delta+delta
    // lleva las frecuencias a la escala bin
    //nBins es el numero de bins
    //el arreglo que debe entrar es el modulo del FFT de tama?o N/2

    public MFCC(int frecuenciaDeSampleo, double [] arreglo, int nBins)
    {
        this.frecuenciaDeSampleo=frecuenciaDeSampleo;
this.arreglo=arreglo;
this.nBins=nBins;
    }

    public void procesar()
    {
        // filtro pre enfasis  alfa>0 filtro de enfasis en bajas frecuencias , alfa <0 se tiene un filtro de enfasis en altas frecuencias
        double alfa=-0.9;
        for(int i=1;i<arreglo.length;i++)
            arreglo[i]=arreglo[i]+alfa*arreglo[i-1];

        // ventaneamiento
        int tam=arreglo.length;
        int tamVentana= 512;   // 16ms
        int tamSlope = 160; //    10ms
        int nroFrames=(int)Math.ceil(((tam-tamVentana+1)/(double)tamSlope)+1);

        double [][] matriz = new double[nroFrames][];
        System.out.println("numero de Frames"+nroFrames);
        matriz=obtenerMatrizVentanamiento(arreglo,tamVentana, tamSlope,nroFrames);

        this.matrizVentaneamiento=matriz;//asignacion a atributo de la clase
        // fourier
         //computa iterativamente la matriz de fourier
        Fourier fft;
        double mayorEspectro=0;
        for(int i=0;i<matriz.length;i++)
        {
            fft=new Fourier(matriz[i]); //construye el objeto Fourier
            fft.FFTradix2DIF(); //computa el fft
            double [] arreglo1 ;
            matriz[i]=fft.moduloI(); //devuelve el modulo de la mitad de valores computados incluyendo el primer elemento

            if(mayorEspectro<fft.obtenerMayor())
                mayorEspectro=fft.obtenerMayor();
        }
        this.matrizFourier=matriz;//asignacion a atributo de la clase
        // bins  cepstrum  DTCII
        int filas=matriz.length;
        nBins=13;

        matrizMFCC=new double[filas][nBins];//crea matriz MFCC

        for(int i=0;i<filas;i++)
            matrizMFCC[i]=MFCC.coeficientesMFCC(frecuenciaDeSampleo,matriz[i],nBins);

        // delta
        // delta-delta

    }



    public double [][] obtenerMatrizVentanamiento( double [] arreglo, int tamVentana, int tamSlope, int nroFrames)
    {
        int a,b,j,k;
        int tam=arreglo.length;
        double [][] matriz = new double[nroFrames][];
j=0;k=0;
a=0;b=tamVentana;

        while(b<=tam)
        {
            matriz[j] = new double[tamVentana];
            for (int i=a;i<b;i++)
            {
                //ventana hamming
                matriz[j][k]=(0.54-0.46*Math.cos(2*Math.PI*k/(tamVentana-1)))*arreglo[i];
                k++;
            }
            a=a+tamSlope;
            b=b+tamSlope;
            j++;
            k=0;
        }
        if(b==tam)
            return matriz;
        if (b>tam)
        {
            matriz[j] = new double[tamVentana];
```

```
            for (int i=a;i<b;i++)
            {
                if(i<tam)
                    matriz[j][k]=(0.54-0.46*Math.cos(2*Math.PI*k/(tamVentana-1)))*arreglo[i];//ventana hamming

                else
                    matriz[j][k]=2*matriz[j][k-1]-matriz[j][k-2];    //extrapolacion

                k++;
            }
        }
        return matriz;
    }

    public static double []  coeficientesMFCC(int Fs, double [] arreglo, int nBins)
    {
        int N=arreglo.length;
double [] Bins= new double [nBins];
double fMenor=0;      //menor frecuencia   0Hz
double fMayor=(double)(N-1)*(double)Fs/((double)N*(double)2);    //mayor frecuencia  = 1/(2*T) o 1*fs/2 criterio nyquist

        double fMelMenor = 0;
double fMelMayor= 1125*(Math.log(1+(double)((fMayor))/700));

        double paso=fMelMayor/(nBins+1);
double k=0,H=0;

        int i=0;int j=0; int ban=0; int pos=0; int b=0;
double suma=0;
double fMenos,fMedio,fMas;

fMenos=0;fMedio=paso;fMas=2*paso;

for(i=0;i<N;i++)
        {
            k= Mel((double)i*(double)Fs/((double)N*(double)2));  //por que es i*Fs/N, pero este vector solo tiene los N/2 samples de fourier, no t

            if(k>fMas)
            {
                Bins[b]=Math.log(suma);   //cepstrum logaritmo natural
i=pos;
ban=0;
suma=0;
fMenos=fMenos+paso;
fMedio=fMedio+paso;
fMas=fMas+paso;
k= Mel((double)i*(double)Fs/((double)N*(double)2));
                b++;
            }
            if(k>=fMenos&&k<fMedio)
            {
                H=(k-fMenos)/paso;
            }
            if(k>=fMedio&&k<=fMas)
            {
                if(ban==0)
                {
                    pos=i;
                    ban=1;
                }
                H=(fMas-k)/paso;
            }
            suma =suma+ arreglo[i]*H;  //pues arreglo[i] contiene los modulos
        }
        Bins[b]=Math.log(suma);  //cepstrum
        Bins=MFCC.DTCII(Bins);
return Bins;
    }

    /****************/
    //trasnformada discreta del coseno para obtener el cepstrum
    //entrada ln[valor del bin]

    public static double [] DTCII(double [] arreglo)
    {
        double [] DCT = new double [arreglo.length];
double suma =0;
double N =(double)arreglo.length;
for(int i=0;i<N;i++)
        {
            for(int j=0;j<N;j++)
            {
            //    suma = suma +arreglo[j]*Math.cos((Math.PI*((double)i+1)*((double)j+0.5))/N);
                suma = suma +arreglo[j]*Math.cos((Math.PI*((double)i)*((double)j+0.5))/N); //ec 6.145 spoken languaje procesing
            }
            DCT[i] = Math.sqrt(2/N)*suma;
            suma=0;
        }
        return DCT;
    }
    private static double Mel(double frecuencia)
    {
        frecuencia=1125*Math.log(1+frecuencia/(double)700);
return frecuencia;
    }
}

package MFCC;

public class Fourier
{
```

```
    int estado;
    double[] arreglo;
    int tamano=0;
    Complejo [] W;
    Complejo [] Datos;
    double mayor=0; //para valor mayo en transformada de fourier
    double mayorValorEspectograma=0;//para mayor valor en espectograma;

    //Algoritmo radix 2 decimacion en frecuencia

    public Fourier(double []arreglo, int estado)
    {
        this.estado=estado;
        this.arreglo=arreglo;
        tamano=this.arreglo.length;
W= new Complejo[arreglo.length/2];  //arreglo de complejos para los factores mariposa
Datos=new Complejo[arreglo.length];
mayor=0;
    }

    public Fourier( double [] arreglo)
    {
        this.arreglo=arreglo;
tamano=this.arreglo.length;
W= new Complejo[arreglo.length/2];  //arreglo de complejos para los factores mariposa
Datos=new Complejo[arreglo.length];
estado=Potencia();
mayor=0;
    }

    //funcion que devuelve el "n" potencia de 2 mas cercana al tama?o del vector
    //si el arreglo de datos no coincide con una potencia de dos , entonces
    //rellena el arreglo con el valor promedio de este hasta
    //igualar una potencia de dos cercana
    public int Potencia()
    {
        double [] tem;
        int nPotencia=0;
        for(int i=0;i<tamano;i++)
        {
            nPotencia=(int)java.lang.Math.pow((double)2,(double)i);
            if(tamano<nPotencia)
            {
                //modifica el arreglo
                tem=this.arreglo;
                this.arreglo= new double[i];
                this.W= new Complejo[nPotencia/2];
                this.Datos=new Complejo[nPotencia];
                this.arreglo=modificarArreglo(tem,i);
                this.tamano=this.arreglo.length;
                this.estado=i-1;
                return estado;
            }
            if(tamano==nPotencia)
            {
                this.estado=i;
return estado;
            }
        }
        return 0;
    }
    //rellena un arreglo que no es potencia de dos hasta hacerlo
    //coincidir con una potencia de dos, con el ultimo
    //del arregloa modificar
    public double [] modificarArreglo(double [] arreglo, int n)
    {
        double [] arregloModificado= new double [(int)java.lang.Math.pow((double)2,(double)n)];
int i;
        for( i=arreglo.length;i< (int)java.lang.Math.pow((double)2,(double)n);i++)
        {
            arregloModificado[i]=arreglo[arreglo.length-1];
        }
        for(i =0;i<arreglo.length;i++)
        {
            arregloModificado[i]=arreglo[i];
        }
        return arregloModificado;
    }

    public void FFTradix2DIF()
    {
        int i=0,j=0;
        Complejo [] temporal=new Complejo [tamano];
        Complejo resultado=new Complejo();

//paso del arreglo de entrada a un arreglo de complejos
        for(i=0;i<tamano;i++)
            Datos[i]=new Complejo(this.arreglo[i]);

        //llenado de los factores mariposa
        int N=tamano;

        for(i=0;i<tamano/2;i++)
            W[i]=new Complejo(Math.cos(2*Math.PI*i/N),-Math.sin(2*Math.PI*i/N));

        //computa radix 2 decimacion en frecuencia FFT,The Gentleman-Sande butterfly.
        int mitad=tamano/2;
        int a=0;
        int division=N/2;
        int contador=1;
        int k=0;   //contador para los factores mariposa
```

```
        int factor=1;  // para accesar a los factores mariposa
        int paso=division;

for(i=0;i<estado;i++)
        {
            k=0;
            for(j=0;j<mitad;j++)
            {
                if(j==division)
                {
                    //contadores
                    a=N*contador;
                    division=division+paso;
                    contador++;
                    k=0;
                }
                temporal[a]       = resultado.sumar(Datos[a],Datos[a+N/2]);
temporal[a+N/2] = resultado.multiplicar(resultado.sumar(Datos[a],(Datos[a+N/2].menosC())),W[k*factor]);

   Datos[a].real=temporal[a].real;
         Datos[a].imaginario=temporal[a].imaginario;
         Datos[a+N/2].real=temporal[a+N/2].real;
         Datos[a+N/2].imaginario=temporal[a+N/2].imaginario;
a++;
                k++;
            }
            a=0;
            N=N/2;
            division=N/2;
            paso=division;
            contador=1;
            factor=factor*2;

        }
        this.Datos=temporal;

/******reordenamieto de bits: decodificando**************/////////

j = 0; i=0; k=0; double xt=0;
for (i = 0; i < tamano- 1; i++)
        {
            if (i < j)
            {
                xt = this.Datos[j].real;
                this.Datos[j].real = this.Datos[i].real;
                this.Datos[i].real = xt;
                xt = this.Datos[j].imaginario;
                this.Datos[j].imaginario = this.Datos[i].imaginario;
                this.Datos[i].imaginario = xt;
            }
            k = tamano / 2;
            while (k <= j)
            {
                j -= k;
k /= 2;
            }
            j += k;
        }

    }
    //devuelve el modulo de solo la mitad de los valores

    public double [] moduloI()
    {
        double [] temp =new double [tamano/2];
        int i=0;
        mayorValorEspectograma=0;

        for(i=0;i<tamano/2;i++)
        {
            temp[i]=Math.sqrt(Math.pow(Datos[i].real,2)+Math.pow((Datos[i].imaginario),2));
            if(mayor<temp[i])
            {
                mayor=temp[i];
            }
        }
        return temp;
    }

    /*lo mismo que el anterior pero sin el primer elemento pues es solo el promedio*/
    public double [] moduloII()
    {
        double [] arreglo =new double [tamano/2-1];
int i=0;

        for(i=0;i<tamano/2-1;i++)
        {
            arreglo[i]=Math.sqrt(Math.pow(Datos[i+1].real,2)+Math.pow((Datos[i+1].imaginario),2));
            if(mayor<arreglo[i])
            {
                mayor=arreglo[i];
            }
        }
        return arreglo;
    }

    public double obtenerMayor()
    {
        return mayor;
    }
```

```
    public void limpiar (Fourier fourier)
    {
        fourier =null;
    }
}

package MFCC;

public class Complejo {

double real;
double imaginario;

/***cosntructores**///

public Complejo(){
real=0.0;
imaginario=0.0;

}

public Complejo(double real, double imaginario) {
this.real=real;
this.imaginario=imaginario;
}

public Complejo(double real){
this.real=real;
imaginario=0;
}

 /****metodos de instancia***///

// modifica la esctructura interna del numero complejo
public void conjugado(){


double tem;
tem= real;
real=imaginario;
imaginario=tem;

}

// no modifica la esctructura interna del numero complejo
public Complejo Cconjugado(){

Complejo conj=new Complejo(imaginario,real);
return conj;
}

public  void menosComplejo(){
this.real=-this.real;
this.imaginario=-this.imaginario;

}

public  Complejo menosC(){
Complejo complejo = new Complejo(-this.real,-this.imaginario);
return complejo;

}


public String cadena(){

if (this.imaginario>=0){
return ("( "+this.real+" + "+this.imaginario+"i"+" )");
}
else{
return ("( "+this.real+" - "+-this.imaginario+"i"+" )");}

}

/***metodos de clase***/

public static Complejo sumar(Complejo c1, Complejo c2) {
Complejo resultado = new Complejo();
resultado.real=c1.real+c2.real;
resultado.imaginario=c1.imaginario+c2.imaginario;
return resultado;

}

public static Complejo multiplicar(Complejo c1, Complejo c2) {

Complejo resultado = new Complejo();
resultado.real=c1.real*c2.real-c1.imaginario*c2.imaginario;
resultado.imaginario=c1.real*c2.imaginario+c1.imaginario*c2.real;
return resultado;

       }


}
```

A.7.5. Diversos utilitarios

```
package modeloocultodemarkov;

/**
 *
 * @author usuario
 */
public class Util {

   //devuelve la suma de dos logaritmos ec 9.54 log(A+B) , pero cuando se ingresa log(A) y log(B)
   public static double sumLog(double logA, double logB)
    {
        double C;
        double temp;
        //puede haber una situacion como sumLog(0,0)=log(exp(0)+exp(0))
        if(logA==0&&logB==0)
        {
            return log(2);
        }
        if (logB>logA)
        {
            temp=logA;
            logA=logB;
            logB=temp;
        }

        C=logB-logA;
        if(C<-700.0005) //pues existe mucha diferencia de magnitud ver alg pag 154 Holmes y aproximadamente el log del numero mas bajo representab
            C=0;
        else
            C=Util.log(1+Math.exp(C));

       return logA+C;
    }
   //devuelve la suma de log(A+B+....)
   public static double sumLog(double[] logs)
   {
        /*ordenar*/
       // sirve para tener ordenadas logaritmos de las probabilidades antes de sumarlas
       // pues se requiere paraevitar errores numericos pues se requiere que cuando
       //sum (logA + logB) logA siempre sea mayot a logB, entonces se requiere realizar una
       //eliminar ceros
       logs=Util.eliminarCeros(logs);

       if(logs==null)
           return 0;
    /*   else
       {
           logs=Util.eliminarCeros(logs);
       }
      */ //ordenacion descendente
       Util.ordenar(logs);
       return sumLogProb(logs);
   }
    //metodo recursivo a usarse en sumLog para devolver la suma de log(A+B+....)
   private static double sumLogProb(double [] logs)
   {
       if(logs.length==1)
               return logs[0];

       if (logs.length==2)
            return Util.sumLog(logs[0],logs[1]);
       else
       {
            double [] temp=new double[logs.length-1];
            for(int i=0;i<temp.length;i++)
                temp[i]=logs[i];

            return sumLogProb(temp)+Util.log(1+Math.exp(logs[logs.length-1]-sumLogProb(temp)));
        }
   }

   public static double[] ordenar(double[] vector)
   {
       quicksort(vector,0,vector.length-1);
       return vector;
   }

   /*quicksort introduction to algorithms capitulo 7*/
   public static void quicksort(double []A,int p,int r)
   {
       if(p<r)
       {
           int q=partition(A,p,r);
           quicksort(A,p,q-1);
           quicksort(A,q+1,r);
       }
   }
   public static int partition(double []A,int p,int r)
   {
       double x=A[r];
       int i=p-1;
       for(int j=p;j<=r-1;j++)
       {
           if (A[j]>=x)
           {
               i=i+1;
               double temp;
               temp=A[i];
               A[i]=A[j];
```

```
            A[j]=temp;
        }
    }
    double temp;
    temp=A[i+1];
    A[i+1]=A[r];
    A[r]=temp;

    return i+1;
}


//funcion de trasnformacion c, d rango de entrada, m,n rango de salida, Y es el numero a encontrar
public static double T(double y, double c, double d, double m, double n)
{
    return ((n-m)*y+m*d-n*c)/(d-c);
}

//calcula la distancia euclidiana entre dos vectores
public static double distancia(double []A,double []B)
{
    double dist=0;
    for(int i=0;i<A.length;i++)
    {
        dist=dist+Math.pow((A[i]-B[i]),2);
    }
    return Math.sqrt(dist);
}
public static double max(double[] y){
    double max=-700;
    for(int i=0;i<y.length;i++){
        if(max<y[i])
            max=y[i];
    }
        return max;
}

public static double[] sumar(double c, double[] A)
{
    for(int i=0;i<A.length;i++)
        A[i]=A[i]+c;
    return A;
}

public static double[] restar(double[] A, double c)
{
    for(int i=0;i<A.length;i++)
        A[i]=A[i]-c;
    return A;
}

public static double[] multiplicar(double c, double[] A)
{
    for(int i=0;i<A.length;i++)
        A[i]=c*A[i];
    return A;
}

public static double [] exp(double []A)
{
    for(int i=0;i<A.length;i++)
        A[i]=Math.exp(A[i]);
    return A;
}
public static double [] log(double [] A)
{
    for(int i=0;i<A.length;i++)
        A[i]=Util.log(A[i]);
    return A;
}
public static double[] restarAbs(double[] A, double[] B)
{
    for(int i=0;i<A.length;i++)
        A[i]=Math.abs(A[i]-B[i]);
    return A;
}
public static double [] eliminarCeros(double[] A)
{
    int cont=0;
    for(int i=0;i<A.length;i++)
    {
            if(A[i]==0)
                cont++;
    }

    double []B=new double[A.length-cont];
    cont=0;
    for(int i=0;i<A.length;i++)
    {
        if(A[i]!=0)
        {
            B[cont]=A[i];
            cont++;
        }
    }
    if(B.length==0)
        return null;
    else
        return B;
}
public static double log(double n)
```

```
   {
       if(n==0)
           return 0;
       if (n==1)
           return 0.000000000000001;
       else if(n>0)
       {
           return Math.log(n);
       }
       else
       {
               return Math.log(-n);
       }
   }
}

package modeloocultodemarkov;

/**
 *
 * @author usuario
 */
public class Matrices {

    public static double [][] determinante(double [][]M)
    {

        return M;

    }



    //calculo de la inversa de una matriz por medio de eliminacion Gauss-Jordan
    public static double [][] inversa(double [][]M)
    {
        System.out.println("matriz inversa");
        double [][]Tem=identidad(M.length);
        double [][]I=identidad(M.length);

        for(int k=0;k<M.length;k++)
        {
            for(int i=0;i<M.length;i++)
            {
                if(k!=i){ Tem[i][k]=-M[i][k]/M[k][k];}
            }

            M=multiplicar(Tem,M);
            I=multiplicar(Tem,I);

            for(int i=0;i<M.length;i++)
            {
                if(k!=i){Tem[i][k]=0;}
            }
        }

        for(int k=0;k<M.length;k++)
        {
            if(M[k][k]>1)
            {
                for(int i=0;i<M.length;i++){ I[k][i]=I[k][i]/M[k][k];}
            }
        }
        return I;
    }

    //devuelde la matriz identidad de tamaño n x n
    public static double[][] identidad(int n)
    {
        double [][]I=new double[n][n];
        for(int i=0;i<n;i++)
        {
            for(int j=0;j<n;j++)
            {
                if(i==j)
                {
                    I[i][j]=1;
                }
                else
                {
                    I[i][j]=0;
                }

            }
        }
        return I;
    }

    //obtiene el promedio de los elementos de cada fila de la matriz
    //M = matriz d x n , n datos d dimensionales
    public static double [] promedio(double [][]M)
    {
        double []u=new double[M.length];
        double suma=0;

        for(int i=0;i<M.length;i++)
        {
            suma=0;
            for(int j=0;j<M[0].length;j++)
            {
            suma=suma+M[i][j];
```

```
        }
        u[i]=suma/M[i].length;
    }

    return u;
}

//matriz M de d x n , es decir n vectores columna d dimensionales

public static double [][] coovarianza(double [][]M)
{
    double []u=Matrices.promedio(M);
    double [][]suma=new double[M.length][M.length];
    double[][]A=new double[M.length][1];
    double[][]B=new double[1][M.length];

    for (int i=0;i<M[0].length;i++)//para todos los n vectores
    {
        for(int j=0;j<M.length;j++) //para cada dimension
        {
            A[j][0]=M[j][i]-u[j];
            B[0][j]=A[j][0];
        }

        suma=sumar(suma,multiplicar(A,B));
    }

    return escalar(suma,(M[0].length-1));
}

public static double [] coovarianzaDiagonal(double [][]M,double [] u)
{
    //double []u=Matrices.promedio(M);
    double []A=new double[M.length];
    double []suma=new double[M.length];

    for(int i=0;i<M.length;i++)
    {
        suma[i]=0;
    }

    for (int i=0;i<M[0].length;i++)//para todos los n vectores
    {
        for(int j=0;j<M.length;j++) //para cada dimension
        {
            A[j]=(M[j][i]-u[j])*(M[j][i]-u[j]);
        }

        //suma=sumar(suma,multiplicar(A,B));
        for(int j=0;j<M.length;j++)
        {
        suma[j]=suma[j]+A[j];
        }

    }

    for(int j=0;j<M.length;j++)
    {
        if(M[0].length!=1)
        {
            suma[j]=suma[j]/(M[0].length-1);
        }
        if(suma[j]==0.0)
        {
            suma[j]=0.00000000000000000000000000000000000000000000000000000000000000000000000000000009;
        }

    }

    return suma;
}


public static double[][] multiplicar(double [][]A,double[][]B)
{
    double[][] C=new double[A.length][B[0].length];
    double suma=0;

    for(int i=0;i<A.length;i++)
    {
        for(int j=0;j<B[0].length;j++)
        {
            suma=0;
            for(int k=0;k<A[0].length;k++)
            {
                suma=suma+A[i][k]*B[k][j];
            }
            C[i][j]=suma;
        }
    }

    return C;
}

//
public static double[][] escalar(double [][]A,double f)
{
    double[][] C=new double[A.length][A[0].length];

    for(int i=0;i<A.length;i++)
    {
```

```
        for(int j=0;j<A[0].length;j++)
        {
            C[i][j]=A[i][j]/f;
        }
    }

    return C;
}

//suma dos matrices C=A+B
public static double [][] sumar(double [][]A,double[][]B)
{
    double[][] C=new double[A.length][A[0].length];


    for(int i=0;i<A.length;i++)
    {
        for(int j=0;j<A[0].length;j++)
        {
            C[i][j]=A[i][j]+B[i][j];
        }
    }
    return C;
}

public static double[][] traspuesta(double [][]M)
{
    double [][]T=new double[M[0].length][M.length];

    for(int i=0;i<T.length;i++)
    {
        for(int j=0;j<T[0].length;j++)
        {
            T[i][j]=M[j][i];
        }
    }

    return T;

}

//redimensiona la matriz de d x n a una matriz de d x (n+1) datos, agregando el vector Y[] en la ultima columna de la matriz
public static double[][] redimensionar(double[][]M,double[]Y)
{
    double[][]temp=M;

    M=new double [M.length][M[0].length+1];

    for(int i=0;i<M.length;i++)
    {
        for(int j=0;j<M[0].length-1;j++)
        {
            M[i][j]=temp[i][j];
        }
    }

    for(int i=0;i<M.length;i++)
    {
        M[i][M[0].length-1]=Y[i];
    }

    return M;
}

public static void mostrar(double [][]M)
{
    for(int i=0;i<M.length;i++)
    {
        for(int j=0;j<M[0].length;j++)
        {
            System.out.print(""+M[i][j]+" ");
        }
        System.out.println(" ");
    }
}

public static void mostrar(double []M)
{
    if(M==null)
        System.out.println("no hay nada");
    else
    {
        for(int i=0;i<M.length;i++)
        {
            System.out.println(""+M[i]+" ");
        }
        System.out.println(" ");
    }

}

public double[] copiar(double []A,double B[])
{
    for(int i=0;i<A.length;i++)
    {
        A[i]=B[i];
    }
    return A;
}

//retorna una matriz de ceros de mxn
```

```
    public static double[][] ceros(int m, int n)
    {
        double[][] M=new double[m][n];
        for(int i=0;i<m;i++)
            for(int j=0;j<n;j++)
                M[i][j]=0;
        return M;
    }

    public static double[] resetear(double A[])
    {
        for(int i=0;i<A.length;i++)
            A[i]=0;
        return A;
    }

    public static double[][] resetear(double A[][])
    {
        for(int i=0;i<A.length;i++)
            for(int j=0;j<A[0].length;j++)
                A[i][j]=0;
        return A;
    }
}
```

Bibliografía

[AC99] International P. Association e C. A. I. Corporate. *Handbook of the International Phonetic Association : A Guide to the Use of the International Phonetic Alphabet*. Cambridge University Press, June 1999. 49, 66, 105

[AU72] Alfred V. Aho e Jeffrey D. Ullman. *The Theory of Parsing, Translation, and Compiling*, volume I: Parsing of *Series in Automatic Computation*. Prentice Hall, Englewood Cliffs, New Jersey, 1972. 49

[Bak] 33

[BE01] Rodrigo Barrantes E. *Investigación: Un camino al conocimiento, un enfoque cualitativo y cuantitativo*. San Jose, CR, 2001. 59, 60

[CT65] James W. Cooley e John W. Tukey. An algorithm for the machine calculation of complex fourier series. *Mathematics of Computation*, 19(90):297–301, 1965. 12

[DÍA05] Salazar O. DÍAZ, J. L. G. Un modelo para el reconocimiento automàtico del habla utilizando wavelts b-spline. *1st Euro-Latin American Workshop on Engineering Systems*, Trujillo. SISTING, 2005. 2

[DÍA06] J. L. G. DÍAZ. Coeficientes cepstrales en escala mel y dynamic time warping para el reconocimiento automático del habla. I Semana de Ciencia de la Computación, 2006. 3

[DÍA07] Salazar O. DÍAZ, J. L. G. Extracción de caracteristicas en el procesamiento digital de una señal para el reconocimiento automático del habla usando wavelets. *VI Jornadas Peruanas de Computación*, Trujillo,2007. 2

[DÍA08a] J. L. G. DÍAZ. Estudio comparativo de extractores de características en un sistema de reconocimiento automático del habla. Sociedad Peruana de Inteligencia Artificia, 2008. 3

[DÍA08b] J. L. G. DÍAZ. Minicurso en reconocimiento automàtico del habla, algoritmos y aplicaciones. III Semana de Ciencia de la Computación, 2008. 3

[DÍA09a] J. L. G. DÍAZ. Clustering para la inicialización de un modelo oculto de markov para el reconocimiento automá;tico del habla. I Evento en Speech Recognition: Algoritmos y Aplicaciones. 2009., 2009. 3

[DÍA09b] J. L. G. DÍAZ. Criterio de implementación de un modelo oculto de markov usando logaritmos de probabilidades. IV Semana de Ciencia de la Computación, 2009. 3

[DHS01] Richard O. Duda, Peter E. Hart, e David G. Stork. *Pattern classification*. Wiley, 2 edição, November 2001. 60, 85, 90, 98

[DM90] Steven B. Davis e Paul Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. páginas 65–74, 1990. 26

[EA00] Chu E. e George A. *Inside the FFT Black Box: Serial and Parallel Fast Fourier Transform Algorithms*. CRC Press,Boca Raton, FL., first edição, 2000. xv, 12, 15

[Foo99] Jonathan Foote. An overview of audio information retrieval. *Multimedia Syst.*, 7(1):2–10, 1999. 1

[GD07] Salazar C.O Guevara D.J. Extracción de características en el procesamiento digital de una señal para el mejoramiento del reconocimiento automático del habla usando wavelets. Tesis pregrado, Universidad Nacional de Trujillo, Escuela de Informática, 2007. 2, 60

[GJJ96] Earl Gose, Richard Johnsonbaugh, e Steve Jost. *Pattern recognition and image analysis*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1996. 25

[HAH01] Xuedong Huang, Alex Acero, e Hsiao-Wuen Hon. *Spoken Language Processing: A Guide to Theory, Algorithm, and System Development*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2001. Foreword By-Reddy, Raj. xv, xv, xv, xv, xv, 16, 17, 18, 19, 21, 30, 49, 63

[HW97] Alexander G. Hauptmann e Michael J. Witbrock. Informedia: news-on-demand multimedia information acquisition and retrieval. páginas 215–239, 1997. 2

[JEC+95] P. Jeanrenaud, E. Eide, U. Chaudhari, J. McDonough, K. Ng, M. Siu, e H. Gish. Reducing word error rate on conversational speech from the switchboard corpus. Em *Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995 International Conference on*, volume 1, páginas 53 –56 vol.1, 9-12 1995. 1, 2

[JFSJY96] G. J. F. Jones, J. T. Foote, K. Spark Jones, e S. J. Young. Robust talker-independent audio document retrieval. Em *ICASSP '96: Proceedings of the Acoustics, Speech, and Signal Processing, 1996. on Conference Proceedings., 1996 IEEE International Conference*, páginas 311–314, Washington, DC, USA, 1996. IEEE Computer Society. 2

[JM09] Daniel Jurafsky e James H. Martin. *Speech and Language Processing (2nd Edition)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 2009. xvi, xvi, xvi, 27, 41, 42, 43, 101

[KJM+97] F. Kubala, H. Jin, S. Matsoukas, L. Nguyen, e R. Schwartz. Broadcast news transcription. Em *Acoustics, Speech, and Signal Processing, 1997. ICASSP-97., 1997 IEEE International Conference on*, volume 1, páginas 203 –206 vol.1, 21-24 1997. 1

[KJN+97] F. Kubala, H. Jin, L. Nguyen, R. Schwartz, e S. Matsoukas. Broadcast news transcription. Em *ICASSP '97: Proceedings of the 1997 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '97) -Volume 1*, página 203, Washington, DC, USA, 1997. IEEE Computer Society. 2

[KKB94] Julian Kupiec, Don Kimber, e Vijay Balasubramanian. Speech-based retrieval using semantic co-occurrence filtering. Em *In Proc. ARPA Human Language Technology Workshop, Plainsboro, NJ*, páginas 373–377. ARPA, 1994. 1

[Mar67] A. A. Markov. An inequality with applications to statistical estimation for probabilistic functions of markov processes and to a model for ecology. *Math. Stat.*, 37(v37.):360–363., 1967. 1, 2, 33, 39

[MNJ+94] J. McDonough, K. Ng, P. Jeanrenaud, H. Gish, e J.R. Rohlicek. Approaches to topic identification on the switchboard corpus. Em *Acoustics, Speech, and Signal Processing, 1994. ICASSP-94., 1994 IEEE International Conference on*, volume i, páginas I/385 –I/388 vol.1, 19-22 1994. 1

[Opp65] Alan V Oppenheim. *Superposition in a Class of Nonlinear Systems*. Tese de Doutorado, Instituto de Computação, Universidade de Campinas, Brasil, marzo 1965. 20

[OS09] Alan V. Oppenheim e Ronald W. Schafer. *Discrete-Time Signal Processing*. Prentice Hall Press, Upper Saddle River, NJ, USA, 2009. 18, 20

[PFF+94] David S. Pallett, Jonathan G. Fiscus, William M. Fisher, John S. Garofolo, Bruce A. Lund, e Mark A. Przybocki. 1993 benchmark tests for the arpa spoken language program. Em *HLT '94: Proceedings of the workshop on Human Language Technology*, páginas 49–74, Morristown, NJ, USA, 1994. Association for Computational Linguistics. 2

[Pur72] Paul Purdom. A sentence generator for testing parsers. *BIT Numerical Mathematics*, 12:366–375, 1972. 10.1007/BF01932308. 68

[Rab90] Lawrence R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. páginas 267–296, 1990. 34, 40

[Rob10] Tony Robinson. *The british english example pronunciation(beep) dictionary*. Cambridge University Engineering Department, Cambridge, UK, 2010. 67

[Ros90] *A hidden Markov model based keyword recognition system*, 1990. 1

[Ros91] R. C. Rose. Techniques for information retrieval from speech messages. *Lincoln Lab. J.*, 4(1):45–60, 1991. 1

[SC78] H. Sakoe e S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 26(1):43 – 49, feb 1978. 1

[SV40] S.S. Stevens e J. Volkman. The relation of pitch to frequency. *Journal of Psychology*, 53:329, 1940. 20

[WB92] L.D. Wilcox e M.A. Bush. Training and search algorithms for an interactive wordspotting system. Em *Acoustics, Speech, and Signal Processing, 1992. ICASSP-92., 1992 IEEE International Conference on*, volume 2, páginas 97 –100 vol.2, 23-26 1992. 1

[Wil92] *Training and search algorithms for an interactive wordspotting system*, volume 2, 1992. 1

[YEG+06] S. J. Young, G. Evermann, M. J. F. Gales, T. Hain, D. Kershaw, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, e P. C. Woodland. *The HTK Book, version 3.4*. Cambridge University Engineering Department, Cambridge, UK, 2006. 50, 51, 55, 64, 67, 70, 72, 73, 85, 103

[YRT89] S.J. Young, N.H. Russell, e J.H.S Thornton. Token passing: a simple conceptual model for connected speech recognition systems. Relatório técnico, 1989. xvi, 53, 54, 76

[ZW09] Lixiao Zheng e Duanyi Wu. A sentence generation algorithm for testing grammars. *Computer Software and Applications Conference, Annual International*, 1:130–135, 2009. 68