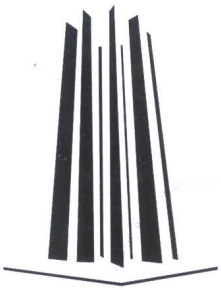
UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

Facultad de Estudios Profesionales campus Aragón

CONSTRUCCIÓN DE UN RECONOCEDOR DE VOZ PARA EL ESPAÑOL DE MÉXICO CON VARIACIÓN ALOFÓNICA MEDIA

T E S I S
QUE PARA OBTENER EL TÍTULO DE:
INGENIERO EN COMPUTACIÓN
P R E S E N T A:

JANET ARACELI JUÁREZ VÁZQUEZ



Asesor: Dr. Luis Alberto Pineda Cortés

México, DF, 2009

Universidad Nacional Autónoma de México
UNAM
Biblioteca Central
Dirección General de Bibliotecas de la UNAM

UNAM – Dirección General de Bibliotecas
Tesis Digitales
Restricciones de uso

DERECHOS RESERVADOS ©
PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL

Todo el material contenido en esta tesis esta protegido por la Ley Federal del Derecho de Autor (LFDA) de los Estados Unidos Mexicanos (México).

El uso de imágenes, fragmentos de videos, y demás material que sea objeto de protección de los derechos de autor, será exclusivamente para fines educativos e informativos y deberá citar la fuente donde la obtuvo mencionando el autor o autores. Cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por el respectivo titular de los Derechos de Autor.

Agradecimientos

En primer lugar quiero agradecer a Dios por haberme guiado durante este tiempo, por ser mi fe y mi fortaleza y por haber puesto en mi camino a todas las personas que me han apoyado incondicionalmente.

A mis padres Sara y Leopoldo, a quienes dedico este trabajo, por confiar en mí siempre y darme la educación y valores que con orgullo siempre práctico. Sin su apoyo no hubiera culminado, gracias por sus bendiciones.

A mis hermanos que siempre me apoyaron, en especial a mi hermana Ana que sin su ayuda no hubiera podido concluir tan rápido con los trámites de la tesis.

A mi sobrino rayito que es mi inspiración día con día para ser mejor.

Sin ser menos importante a mi asesor de tesis el Dr. Luis A. Pineda Cortés por su apoyo y confianza, sobre todo por compartir sus conocimientos conmigo y por haberme exigido lo mejor y hacer que diera el 200% siempre.

A mis compañeros y amigos del instituto, Ana, Alejandra, Hayde, Wendy, Irving, Varinia, Laura y en especial a Paty por haberme enseñado, apoyado y ayudado en toda mi estancia en el instituto. Pero sobre todo gracias por haberme brindado su amistad.

A todos mis amigos que siempre me incitaron e inspiraron a terminar. Gracias por sus consejos y apoyo, son los mejores.

A las personas que ya no se encuentran conmigo en este momento pero que en el camino me apoyaron y brindaron su amistad.

Quiero agradecer al Dr. Luis Villaseñor del INAOE, Puebla, por prestarnos sus instalaciones, herramientas y su tiempo para obtener parte de los recursos necesarios para el término de este trabajo.

A todas la personas que contribuyeron en recopilar y etiquetar el corpus DIMEx100 (ISBN:970-32-3395-3), el cual fue creado en el departamento de Ciencias de la Computación del Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas (IIMAS), UNAM en el contexto del proyecto DIME-II: Diálogos Inteligentes Multimodales en Español, NSF/CONACYT 39380-U.

Gracias a todos por ser parte de mi vida, los quiero y esté es nuestro logro.

Índice

Índice de Tablas

Índice de Figuras

Introducción

Cada vez son más frecuentes los escenarios en los que el hombre tiene que interactuar con máquinas, ésta se lleva a cabo mediante una pantalla en la cual el usuario recibe una determinada información y por medio de un teclado o mouse -por ejemplo - le dice que hacer; pero sin duda alguna, resulta más deseable poder recibir información de la misma forma como lo hace naturalmente el ser humano: mediante la comunicación oral en su propio idioma.

Desde el inicio de la humanidad la forma de entendimiento entre los hombres ha sido de manera oral y es por ello que los investigadores y desarrolladores de tecnología buscan reproducir este tipo de comunicación con las máquinas, ya que es percibida como un factor determinante en la mejora de la interacción hombre-máquina.

Las tecnologías del habla son las disciplinas que se encargan de hacer esto posible y están divididas principalmente en dos campos: el reconocimiento automático del habla y la síntesis. El primer campo está enfocado en desarrollar técnicas y algoritmos para crear sistemas que produzcan voz, es decir, que puedan hablar. El segundo campo tiene por objetivo desarrollar técnicas y algoritmos para crear sistemas capaces de escuchar y transcribir lo que escucho a texto.

Este trabajo se enmarca dentro de las tecnologías del habla, particularmente del reconocimiento automático de voz y es por ello que en lo subsiguiente solo se abordaran temas relacionados con éste.

En los últimos años se han producido avances importantes en lo que al reconocimiento automático del habla se refiere, aunque todavía con limitaciones de vocabulario, por dominio de aplicación, por disfluencia en la locución, etc.; pero pese a estas limitaciones, la tecnología está preparada para ofrecer una amplia gama de servicios; así muchos

investigadores predicen que es un buen momento para perfeccionar el procesamiento del lenguaje[1].

El primer objetivo para el procesamiento del lenguaje es entender el lenguaje hablado; esto se vuelve una tarea difícil debido a los diferentes factores que influyen en su reconocimiento, tales como la ambigüedad, la pronunciación, la velocidad con la que se habla, el acento del hablante, el medio ambiente, sexo, edad y hasta el estado de ánimo; existen además factores de carácter lingüístico, como los relacionados con las distintas formas dialécticas, la utilización de palabras no contempladas en el vocabulario, la construcción de frases no gramaticales, la utilización de abreviaturas, los escenarios semánticos de las palabras, etc. Por ello la forma de atacar computacionalmente algunos de éstos, es construir sistemas que mapeen una señal acústica a una cadena de palabras contenidas en un diccionario de pronunciación - ya que la función de un sistema de reconocimiento de voz es transformar automáticamente una señal acústica a texto -, convirtiéndose éste en el principal objetivo del reconocimiento automático de voz (ASR – Automatic Speech Recognition).

Aunque los sistemas ASR aún se encuentren lejos de resolver el problema de la transcripción automática de voz de cualquier hablante en cualquier ambiente, en recientes años se ha visto una maduración en esta tecnología, hasta el punto en donde esto es viable si el dominio es limitado[2].

Ahora bien el proceso para llevar a cabo esta conversión se puede ver en dos fases; primero el sistema modela cada uno de los sonidos básicos que componen el habla y después utiliza este conocimiento para encontrar la secuencia de palabras que más se parecen a lo que el hablante dijo. En este último paso el sistema incorpora conocimiento acerca de la estructura sintáctica, semántica y pragmática del lenguaje.

[1] Jurasfky & Martin, *Speech and Language Processing*, Prentice Hall 2000.
[2] Elia P. Peréz Pavón, *Construcción de un reconocedor de voz utilizando Sphinx y el corpus DIMEx100*, UNAM, 2006.

Pero para que el sistema obtenga el conocimiento necesario y realice de una manera óptima el reconocimiento se requiere de una base empírica, la cual consiste en la recopilación, análisis y transcripción de vastos recursos lingüísticos, ya que éstos constituyen una de las principales formas de representar el conocimiento de la lengua. Estos recursos llamados corpora[3] presentan una colección de textos o grabaciones que reflejan el contexto en el que se utiliza la lengua. En ellos podemos encontrar ejemplos concretos acerca del uso de palabras, expresiones, regímenes verbales, locuciones, entre otros. Por ejemplo, un corpus puede contener grabaciones de conversaciones entre recepcionistas y clientes que luego se utilizarán para desarrollar un sistema telefónico automático de reservas que reconoce las elocuciones del cliente.

Podemos distinguir entonces dos características que deben presentar los corpora, primero, que estén bien etiquetados para que el reconocedor aprenda la variabilidad acústica de la lengua, es decir, tenga bien definidas cada una de las unidades fonéticas o alofónicas y segundo, tener un recurso que refleje el uso de palabras y expresiones en un determinado contexto.

Actualmente, la comunicación hombre-máquina a través de la voz se ha convertido en una meta común de varias disciplinas, la cual las ha llevado a conjuntarse por la necesidad de desarrollar sistemas en los cuales se pueda apoyar el ser humano, y sobre todo con el cual se pueda establecer la comunicación; sistemas que permitan a una máquina interpretar el significado de nuestras palabras y, a partir de ellas, tomar decisiones, ejecutar ordenes, o simplemente dar la información que le sea solicitada; sistemas en los cuales sea más práctico incorporar un altavoz que una pantalla, en los que se requiera del acceso a información a través de un teléfono, o bien, en los que se pueda prescindir de un teclado. Adicionalmente, estos sistemas resultan de gran utilidad para personas invidentes o con alguna otra discapacidad.

En la actualidad está tecnología ha sido muy solicitada sobretodo en la telefonía, en la cual el reconocimiento de voz se utiliza para reconocer dígitos o para interpretar palabras que

[3] Corpora es el plural de la palabra corpus.

hacen alusión a un servicio; dentro de otros usos actuales y potenciales se pueden mencionar los restaurantes en donde se accede al menú a través del habla, sitios de información turística y lugares en donde el usuario tiene que manipular objetos o tiene que controlar equipos. Otra aplicación es en el dictado, en el cuál se hace la transcripción de lo que dice un solo usuario a texto.

Adicionalmente, los adelantos en esta tecnología – principalmente en países de habla inglesa - permiten la comunicación multimodal con una máquina sin la necesidad del teclado, mouse o pantalla; así, algunas personas discapacitadas pueden utilizar una computadora o dispositivo para trabajar y comunicarse; pero no sólo ellos precisan de las nuevas tecnologías para poder llevar a cabo su vida de la manera más cómoda posible, sino todas las personas que necesitan esta tecnología para facilitar y optimar su trabajo diario; por ello, en el mercado existen productos basados en el reconocimiento de voz como medio para cerrar una puerta, una ventana o encender la televisión, así como herramientas para el desarrollo de las mismas. A pesar de este avance y apertura, en México, así como en otros países de habla hispana, no se ha hecho mucho trabajo en torno a esta tecnología, lo cual da como resultado muy pocos recursos y herramientas de fácil acceso para crear interfaces en el idioma español.

Tomando en cuenta que el español es la segunda lengua más hablada en el mundo y que la lingüística del español – en todos sus niveles – es tan compleja como la de los demás idiomas y con base en el poco trabajo que se ha realizado en el país, el presente trabajo tiene por objetivo construir un reconocedor de voz en español de México para validar cualitativamente la etiquetación del corpus DIMEx100[4] - un nuevo corpus en español de México etiquetado manualmente en tres niveles de representación de la lengua (fina, media y básica) creado dentro del proyecto DIME II (Diálogos Inteligentes Multimodales en Español), desarrollado en el departamento de Ciencias de la Computación del IIMAS (Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas) - en el nivel medio; para este efecto se creará un diccionario fonético en este nivel para ser utilizado en la construcción de los modelos acústicos. Asimismo este reconocedor servirá para comparar

[4] Corpus DIMEx100. http://leibniz.iimas.unam.mx/~luis/DIME/index.html

los resultados arrojados por el reconocedor creado con el nivel de granularidad básica, y así poder deducir que nivel de granularidad de transcripción es la que da mejores resultados en la construcción de sistemas ASR.

Dentro del departamento también se desarrolla el proyecto GOLEM, que tiene por objetivo estudiar la integración de información multimodal, principalmente lingüística y de lenguaje natural en español hablado, para la navegación de un robot en un espacio limitado; dentro de este contexto se programó al robot móvil (GOLEM), para que dé una visita guiada a los visitantes del Departamento de Ciencias de la Computación del IIMAS-UNAM.

En la segunda fase de este trabajo se construyó un modelo del lenguaje para el robot *GOLEM*; no hay que perder de vista que los reconocedores de voz creados sólo están validando los modelos acústicos, es decir, solo se está trabajando con la primera fase en el proceso de reconocimiento; por ello en esta segunda parte se creó un modelo del lenguaje que permite identificar las secuencias de palabras más probables en los diálogos que se llevan a cabo con GOLEM; es decir, se creó un recurso que refleja el uso de palabras y expresiones utilizadas dentro del contexto de la visita guiada del Departamento de Ciencias de la Computación del IIMAS-UNAM.

En los siguientes capítulos de la tesis se explicará el avance e impacto que ha tenido esta tecnología a lo largo del tiempo, la importancia de contar con un recurso lingüístico en español de fácil acceso - que ha sido una parte importante para la realización de este trabajo, y una parte del trabajo de investigación que se realiza en México.

Esta tesis está dividida en seis capítulos, una introducción y una sección de apéndices en la parte final, en donde se encuentran los códigos, algoritmos y procedimientos utilizados a lo largo del desarrollo del trabajo.

El capítulo uno introduce la historia del procesamiento del lenguaje, el trabajo y los avances que se han realizado, así como algunos de los modelos y teorías matemáticas que resuelven algunos problemas del procesamiento del lenguaje entre los que destacan: los

modelos ocultos de Markov, el algoritmo de viterbi, la teoría de los autómatas, etc.; se hablará, además, de los modelos probabilísticos del procesamiento del lenguaje, particularmente del teorema de inferencia de Bayes y de los componentes de un reconocedor de voz y su funcionamiento.

En el capítulo dos se describe el corpus Dimex100: un nuevo corpus de habla para el español de México; su recopilación, etiquetación, características y estadísticas; se mostrará cómo este corpus representa un valioso recurso para la construcción de reconocedores de voz en español, para la investigación en reconocimiento de locutores y para la creación de voces para sistemas de síntesis de voz. De igual forma, se explica la importancia de los conocimientos lingüísticos involucrados en la construcción de este tipo de sistemas.

En el capítulo tres, se explica la construcción del reconocedor de voz utilizando Sphinx[5], asimismo se detallan los datos que se requieren para su construcción tales como los modelos acústicos, el diccionario de pronunciación y el modelo de lenguaje, así como las diferentes técnicas empleadas para crearlos.

En el capítulo cuatro, se presenta el modelo del lenguaje; éste es un indispensable componente de cualquier sistema de reconocimiento de voz, cuyo propósito es reducir el espacio de búsqueda y acelerar el proceso de reconocimiento; se presentan también las técnicas de recopilación del corpus para la creación del mismo. Además, se hablará del modelo del lenguaje creado para *GOLEM*.

El capítulo cinco se divide en dos secciones. En la primera se presentan la técnica y los resultados arrojados por los experimentos realizados con el reconocedor de voz que se construyó con la etiquetación del nivel medio; así como la comparación de resultados con el reconocedor de voz del nivel básico. En la segunda sección se presentan los resultados arrojados por los experimentos realizados con el modelo del lenguaje creado para *GOLEM*.

[5] Software de la universidad de Carnegie Mellon. http://cmusphinx.sourceforge.net/html/cmusphinx.php

Finalmente en el capítulo seis se dan las conclusiones y consideraciones surgidas con el desarrollo del presente trabajo, así como las expectativas a largo plazo.

Capítulo 1

Estado del arte en el procesamiento del lenguaje

A lo largo de la historia el ser humano ha utilizado el lenguaje para transmitir sus conocimientos, sentimientos, emociones, sensaciones, es decir, para comunicarse con el mundo y esta necesidad lo ha llevado a desarrollar el lenguaje de manera oral, gráfica, escrita e incluso por señas. Es por ello que el procesamiento del lenguaje natural pretende que una computadora pueda comunicarse con un ser humano en su propia lengua (inglés, francés, español, etc.). Esto también ha llevado a que el lenguaje sea tratado por diferentes áreas de investigación: la lingüística computacional en lingüística, el procesamiento del lenguaje natural en ciencias de la computación, el reconocimiento del habla en ingeniería eléctrica y la psicolingüística computacional en psicología.

Para empezar a hablar de un sistema de reconocimiento de voz es necesario conocer que es lo que hay detrás, la historia, las teorías, que son la base para entender el funcionamiento de esta tecnología y por supuesto parte del marco teórico de este trabajo.

En este capítulo se presentan los inicios y trabajos que se han hecho con respecto al procesamiento del lenguaje, ya que es el primer paso y la base para construir sistemas que reconozcan la voz automáticamente; también se presenta la formulación matemática y el funcionamiento general de un sistema de reconocimiento de voz.

1.1 Historia

Después de la Segunda Guerra Mundial, la investigación del procesamiento del lenguaje tuvo un importante crecimiento, particularmente durante los años cuarenta y finales de los cincuenta, en donde se ve reflejado el trabajo con la creación de los autómatas y las teorías probabilísticas. Los autómatas surgieron como consecuencia de la máquina de Turing (1936), que para muchos es la raíz de la computación moderna; principio que más tarde

dio lugar a la creación de la computadora digital. El trabajo de Turing además inspiró a McColloch y Pitts (1943) quienes desarrollaron el modelo de la neurona descrito en términos de la lógica proposicional; Kleene (1951 y 1956) introdujo los autómatas finitos y expresiones regulares; Claude Shannon (1948) aplicó modelos probabilísticas de procesos discretos de Markov. Con base en esto Noam Chomsky (1956) definió un lenguaje de estado finito como un lenguaje generado por una gramática de estados finitos. De estas ideas surgió la teoría de los lenguajes formales.

Un aspecto importante de este período fue el desarrollo de algoritmos probabilísticos para el procesamiento del lenguaje. Además Shannon propuso la teoría de la información en *Teoría Matemática de la Comunicación* donde se muestra que todas las fuentes de información se pueden medir y que los canales de comunicación tienen una unidad de medida similar. En esta teoría sentó las bases para la corrección de errores, supresión de ruidos y redundancia. El concepto de la entropía es una característica importante de su teoría que explica que existe un cierto grado de incertidumbre de que el mensaje llegué completo.

A principios de los sesenta el procesamiento del lenguaje se dividió en dos paradigmas principales: los simbólicos y los estocásticos.

El paradigma simbólico tomó dos vertientes; por un lado el trabajo de Chomsky y su teoría de los lenguajes formales, así como el trabajo de muchos lingüistas y científicos de la computación con sus algoritmos de parseo y programación dinámica y, por el otro, el nacimiento de un nuevo campo, la Inteligencia Artificial.

El paradigma estocástico surgió en los departamentos de estadística y de ingeniería electrónica, donde el método de Bayes se empezó a usar para resolver el problema del reconocimiento óptico de caracteres (Optical Character Recognition – OCR). En 1959, Bledsoe y Browning construyeron un sistema bayesiano para reconocimiento de texto basado en una red neuronal.

Durante los siguientes años y hasta comienzos de los ochenta el paradigma estocástico jugó el papel principal en el desarrollo de algoritmos para reconocimiento de voz. Trabajaron bajo este paradigma Shannon, Jelinek, Bahl, Mercer y otros investigadores del Centro de Investigación Thomas J. Watson de IBM y Baker en la Universidad de Carnegie Mellon, quien fue influenciado por el trabajo de Baum y sus colegas en Princeton.

El entendimiento del lenguaje natural, despegó con Terry Winograd y su sistema SHRDLU en 1972, al simular un robot dentro de un mundo de bloques de juguete, que era capaz de aceptar comandos de texto en lenguaje natural. Otro ejemplo es el sistema de pregunta-respuesta LUNAR (Woods, 1967, 1973), creado bajo el concepto de los dos paradigmas – estocástico y simbólico – los cuales utilizaron la lógica de predicados como una representación semántica.

Entre los años 1983 y 1993 se retomaron los modelos de estado finito gracias al trabajo de Kaplan y Kay (1981), y Church(1980). A este período se le llamo *el regreso del empirismo* porque se les dio un mayor uso a los modelos probabilísticos en el procesamiento del lenguaje. Durante este periodo se vio un gran auge en la generación del lenguaje natural.

En los últimos años del siglo XX, se estandarizó el uso de la probabilidad en las herramientas para el procesamiento del lenguaje; además muchos algoritmos de parseo, de etiquetación de partes del habla, de procesamiento del discurso y otros métodos empezaron a incorporar probabilidades. El incremento en la velocidad de la memoria de las computadoras ha permitido la explotación comercial de algunas subáreas del procesamiento del lenguaje, principalmente en el reconocimiento del habla y la revisión de la ortografía y gramática.

Durante este periodo de investigaciones se han desarrollado muchas aplicaciones, y con el crecimiento de la web se enfatiza aún más la necesidad de desarrollar nuevas aplicaciones de esta tecnología.

A continuación se mencionan algunos escenarios en donde se ha aplicado, en alguna de sus subáreas, el procesamiento del lenguaje:

- En Cambridge, Massachussets, un visitante puede solicitar a una computadora lugares en donde comer usando el lenguaje oral, a lo cual el sistema responde con información relevante acerca de posibles restaurantes en la zona (Zue *et al.*, 1991).
- El sistema de traducción *Babel Fish* de Systran – utilizado en el sitio web *Altavista* - maneja más de un millón de peticiones de traducción al día.

Estos escenarios representan algunas de las posibles aplicaciones de esta tecnología; a continuación se presentan investigaciones y trabajos que han sido realizados en diferentes laboratorios alrededor del mundo.

- En el laboratorio Bell en 1952, se creó la primera máquina reconocedora de dígitos, este sistema estadístico podía reconocer cualquiera de los diez dígitos (0-9) dichos por un hablante (Davis *et al.*, 1952).
- A principios de 1970, Lenny Baum, de la Universidad de Princeton, desarrolla el enfoque del Modelo Oculto de Markov hacia el reconocimiento de voz.
- En 1971, DARPA (*Defense Advanced Research Projects Agency*) establece el programa SUR (*Speech Understanding Research*) para desarrollar un sistema computacional que pudiera entender el habla continua. Los principales grupos del proyecto SUR se establecieron en CMU, SRI, el Laboratorio Lincoln de MIT, SDC (Systems Development Corporation) y BBN (Bolt, Beranek and Newman).
- Durante los años setenta se desarrollaron los sistemas de conversión de texto en habla, con los que es posible "escuchar" un texto almacenado en una computadora. Al mismo tiempo surgen sistemas que reconocen palabras aisladas, sistemas que verifican la identidad de la persona que habla y nuevas técnicas de codificación de la voz para mejorar su procesamiento.
- En la década de los ochenta, los adelantos en el campo del reconocimiento de voz continua eliminaron las pausas entre cada palabra para que la computadora reconociera enunciados. En 1982, Jim y Janet Barker, pioneros de la industria de habla, fundaron Dragon Systems. En 1989, Steve Young desarrolla la primera

versión de HTK (Hidden Markov Model Toolkit) dentro del grupo *Speech Vision and Robotics* de la Universidad de Cambridge.

En la actualidad hay un sin fin de aplicaciones que fueron creadas con la tecnología del procesamiento del lenguaje en cualquiera de sus niveles; además en el mercado, es posible encontrar desde un sistema de dictado, hasta un sistema tan complejo que pueda responder por un lugar o darnos sugerencias de dónde comer. Además ya es posible encontrar tanto navegadores como sistemas operativos con los que se interactúa por medio del habla.

Existe un gran número de empresas que desarrollan productos con aplicaciones comerciales, tal es el caso de Scansoft que en conjunto con Nuance, dominan el campo de aplicaciones para telefonía; IBM con su sistema IBM Via Voice y el sistema de dictado de la compañía Philips.

1.2 Modelos y Algoritmos

Los sistemas de reconocimiento de voz en la actualidad tienen un sin fin de aplicaciones; por ello se requiere que estos sistemas sean robustos y que tengan un buen desempeño e interacción más natural. Para esto se han desarrollado técnicas y modelos que permiten a los sistemas establecer la comunicación de una manera sencilla con los humanos a través de la voz; pero existen diferentes niveles de representación en el lenguaje humano como la fonética, la fonología, la morfología, la sintaxis, la semántica, la pragmática y el discurso, que aunque todos están involucrados en el proceso del lenguaje humano, los modelos computacionales de reconocimiento tienen que centrarse en uno o varios dependiendo de la tarea, ya que considerarlos todos es sumamente complejo.

En los últimos cincuenta años, se han desarrollado modelos formales y teorías que resuelven algunos de estos problemas. Estos modelos y teorías han sido incluidos en algunas herramientas utilizadas en el reconocimiento de voz, las cuales resultan de gran ayuda en el desarrollo de los sistemas y permiten el desarrollo de nuevas técnicas.

Los modelos más utilizados en el desarrollo de las herramientas son: las máquinas de estados finitos, los sistemas de reglas formales, la lógica y las teorías probabilísticas. Estos modelos a su vez, utilizan diferentes algoritmos entre los que se encuentran los de búsqueda en el espacio de estados y los de programación dinámica.

Las máquinas de estados finitos, en su formulación más simple, son consideradas modelos formales que consisten de estados, transiciones entre los estados y una entrada que es un factor determinante en su comportamiento. Algunas de las variaciones de este modelo básico son: los autómatas determinísticos de estado finito y los no determinísticos, transductores de estado finito, autómatas de peso, modelos de Markov y Modelos Ocultos de Markov (HMM por sus siglas en inglés Hidden Markov Model)[1], los cuales tienen un componente probabilístico.

Estos últimos son los más utilizados en el reconocimiento de voz, ya que sus características permiten obtener un mejor reconocimiento. Para entender mejor un HMM es necesario hablar del modelo de Markov, el cual, en su formulación más simple, nos dice que los eventos futuros de una secuencia son condicionalmente independientes del pasado dado el presente estado [Manning, 99]; además, está compuesto por un conjunto de estados finales y uno inicial, y evoluciona de un estado a otro con transiciones probabilísticas, las cuales son totalmente observables. El HMM toma características adicionales de un modelo de Markov y cambia algunas otras; en el HMM no se conoce la secuencia de estados por los cuales ha pasado (el proceso), pero si las funciones probabilísticas de las observaciones. Además la función de probabilidad de observación no está limitada a solo dos valores 0 ó 1, sino que puede tomar cualquier valor en el rango de 0 a 1.

Los sistemas de reglas formales - también llamados axiomáticos -, son artificios matemáticos compuestos de símbolos que se unen entre sí para formar cadenas, que a su vez, pueden ser manipuladas según reglas para producir otras cadenas. De esta manera, el sistema formal es capaz de representar cierto aspecto de la realidad. Dentro de estas reglas

[1] De ahora en adelante me referiré a los Modelos Ocultos de Markov como HMM.

formales se encuentran: las gramáticas regulares, las gramáticas libres de contexto y las gramáticas de características aumentadas, incluyendo todas sus variantes probabilísticas. Las máquinas de estado y los sistemas de reglas formales son las principales herramientas usadas en el tratamiento de la fonología, morfología y sintaxis [Jurasfky&Martin, 00].

Los algoritmos asociados a las máquinas de estados finitos, con los sistemas de reglas formales, comúnmente confinan un espacio de búsqueda a través de los estados que representan las hipótesis arrojadas como consecuencia de la entrada. Algunas tareas representativas son: buscar las secuencias fonológicas para la palabra de entrada probable en el reconocimiento del habla en un determinado espacio, o buscar el árbol sintáctico correcto de un enunciado de entrada. Los algoritmos más utilizados para estas tareas son los llamados algoritmos de grafos, entre los que se encuentran: búsqueda en profundidad, y sus variantes heurísticas, búsqueda Best-first, y búsqueda A*.

El algoritmo de *búsqueda en profundidad* es un algoritmo que permite recorrer todos los nodos de un grafo o árbol de manera ordenada, pero no uniforme. Su funcionamiento consiste en ir expandiendo los nodos que va localizando, de forma recurrente, en un camino concreto. Cuando ya no quedan más nodos que visitar en dicho camino, regresa, de modo que repite el mismo proceso para cada uno de los nodos restantes.

El algoritmo de *búsqueda A** es un algoritmo que encuentra, el camino de menor coste entre un nodo origen y el nodo objetivo. El algoritmo de *búsqueda A** junto con el algoritmo de *Viterbi* calculan simultáneamente la probabilidad de una secuencia de observación dada, y arrojan la elocución más probable de acuerdo con el canal ruidoso. El algoritmo de *Viterbi* se utiliza para encontrar, dada una secuencia de símbolos, la serie de transiciones más probable entre los estados de una cadena de Markov necesaria para producir dicha secuencia.

El tercer modelo que juega un papel importante en el procesamiento del lenguaje es la lógica; con este modelo se pretende capturar el conocimiento del lenguaje, estudiando la forma del razonamiento; es una disciplina que estudia si los argumentos expresados a

través del lenguaje son válidos o no, por medio de reglas y técnicas. Dentro de este ramo se encuentra: la lógica de primer orden o cálculo de predicados, redes semánticas y dependencia conceptual. Estas representaciones lógicas comúnmente son utilizadas para resolver problemas, hasta cierto punto, en aspectos de semántica, pragmática y discurso.

La teoría probabilística, el elemento final en este conjunto de modelos, ayuda a capturar el conocimiento del lenguaje; así cada uno de los modelos, antes descritos, puede ser enriquecido con las teorías probabilísticas. Un uso de la teoría de la probabilidad es resolver diferentes clases de problemas de ambigüedad; otra aplicación de esta teoría es su uso en las máquinas de aprendizaje.

1.3 Fundamentos matemáticos

La variación en la pronunciación, la ambigüedad, etc. son algunos problemas a los que se enfrentan los sistemas de reconocimiento de voz; por ello tanto para ASR como para otras aplicaciones[2], se propone mapear de una cadena de símbolos a otra, es decir, dada una cadena de símbolos – en este caso la representación fonética de la señal hablada - se busca la correspondiente cadena de símbolos en el diccionario de pronunciación. Generalizando de forma matemática, el problema de ASR se puede formular desde un punto de vista estadístico. Particularmente el modelo del canal ruidoso y las reglas de inferencia de Bayes proveen la estructura probabilística para atacar algunos de estos problemas.

La metáfora del canal ruidoso o modelo de inferencia de Bayes (Figura 1) fue introducido por Jelinek en 1976, en un modelo del reconocimiento de voz creado en los laboratorios de IBM. Está intuición propone tratar a la entrada (una mala pronunciación) como una instancia de la forma léxica (pronunciación léxica), la cual ha pasado a través de un canal de comunicación ruidoso. Este canal introduce ruido, lo cual dificulta el reconocimiento de la palabra o elocución original; este ruido puede ser causado por variación en la

[2] Reconocimiento Óptico de Caracteres.

pronunciación, variación en la realización de los fonemas o variación acústica debido al canal (micrófono, teléfono, la red, etc.).

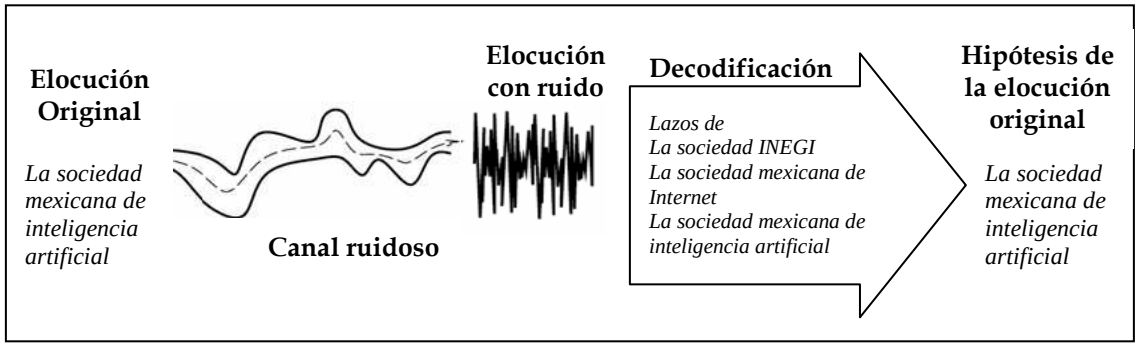


Figura 1. Modelo del canal ruidoso.

El objetivo es construir un modelo que permita entender como fue modificada la palabra o elocución original para poder recuperarla; lo que se busca es la elocución de salida más probable de todas las elocuciones dentro de un vocabulario V dado que hay una entrada acústica O, en donde O es un conjunto de símbolos tomados de algún alfabeto *O*; así la entrada acústica O es tratada como una secuencia de símbolos u observaciones individuales (por ejemplo, muestreando la señal cada 10 ms y representando cada segmento mediante valores de la frecuencia de ese segmento). Cada índice representa un intervalo de tiempo, y sucesivos o_i representan segmentos consecutivos temporales de la entrada:

$$\mathbf{O} = o_1, o_2, o_3, \ldots, o_t \qquad o_i \in O$$

De la misma forma se trata a una elocución, esto es, como si estuviera formada por una cadena de palabras que pertenecen a un cierto vocabulario o diccionario V:

$$W = w_1, w_2, w_3, \ldots, w_n \qquad w_i \in V$$

Lo que se busca es la cadena de palabras más probable dado que existe la observación O, esto se puede expresar de la siguiente manera:

$$\hat{w} = \underset{W \in V}{\operatorname{argmax}}\, P(W \mid O) \qquad \textbf{(1)}$$

En donde $\hat{w}$ hace referencia a la secuencia de palabras más probable dentro de *V* dado que se observó la secuencia *O*; así la ecuación (1) expresa cuál es la secuencia optima W, pero dada una elocución W y una secuencia acústica O se necesita calcular P(W|O), lo que significa obtener la probabilidad de una elocución W dadas todas las posibles observaciones en el mundo de acuerdo a un alfabeto cualquiera O. En la práctica, P(W|O) resulta imposible de obtener; sin embargo, es posible aproximar este valor a través del teorema de Bayes.

Una prueba diagramática de este teorema se ilustra en la Figura 2, en donde se observa que la probabilidad de que ocurra el evento *x* dado que el evento *y* ha sucedido, es decir P(x|y), es,

$$P(x|y) = P(x \cap y) \,/\, P(y) \qquad \textbf{(2)}$$

De manera similar la probabilidad de P(y|x) es,

$$P(y|x) = P(x \cap y) \,/\, P(x) \qquad \textbf{(3)}$$

por lo que despejando el término común P(x∩y) tenemos que,

$$P(x|y)\,P(y) = P(y|x)\,P(x) \qquad \textbf{(4)}$$

Así despejando P(x|y) de esta igualdad tenemos,

$$P(x|y) = \frac{P(y|x)P(x)}{P(y)} \qquad \textbf{(5)}$$

Esto es porque el conjunto de intersecciones es simétrico.

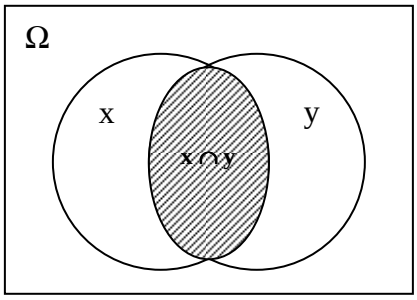


Figura 2. Un diagrama ilustrando el cálculo de la probabilidad condicional P(x|y).

Para nuestro caso $x = W$, es decir, la cadena a reconocer y $y = O$, la observación, como lo que se busca es la W con mayor probabilidad dada una O, entonces se sustituye (5) en (1) y obtenemos la ecuación (6):

$$\hat{w} = \underset{W \in V}{argmax} \frac{P(O|W)\,P(W)}{P(O)} \tag{6}$$

En donde:

P(W): Es la probabilidad de la secuencia de palabras W,

P(O|W): Es la probabilidad de observar la secuencia de datos acústicos O dado que se expresó la secuencia de palabras W, y

P(O): Es la probabilidad de la secuencia de observación.

La probabilidad de la secuencia de observación acústica es muy difícil de estimar ya que para este efecto se requeriría contar todas las observaciones posibles; sin embargo se puede observar que $P(O)$ es un valor constante, ya que O es la misma secuencia de observación de referencia para cada elocución; por lo tanto $P(O)$ es la misma para todas las W en un evento dado. Como el denominador es el mismo para cada elocución candidata W se puede ignorar, obteniendo la expresión en (7):

$$\hat{w} = \underset{W \in V}{argmax} \frac{P(O|W)\,P(W)}{P(O)} = \underset{W \in V}{argmax}\, P(O|W)\,P(W) \tag{7}$$

El término $P(O|W)$ se puede pensar como la probabilidad de que el hablante haya dicho O dado que se tenía la intención de comunicar W, lo cual es mucho más fácil de modelar que $P(W|O)$. Por su parte $P(W)$ representa la probabilidad absoluta de que W ocurra del todo, es decir, esta probabilidad es una función del lenguaje como un todo y es independiente de cada evento de comunicación.

Así, la elocución W más probable dada una secuencia de observación O se calcula mediante el producto de dos probabilidades $P(O|W)$ y $P(W)$. En el reconocimiento de voz, la probabilidad de observación $P(O|W)$, que relaciona los datos acústicos con la secuencia

de palabras se calcula a través del modelo acústico, y la probabilidad *P(W)*, llamada probabilidad *a priori*, se calcula a través del modelo del lenguaje como se ve en (8).

$$\hat{w} = \underset{W \in V}{argmax} \underbrace{P(O \mid W)}_{\text{Modelo acústico}} \underbrace{P(W)}_{\text{Modelo del lenguaje}} \quad \textbf{(8)}$$

1.4 Funcionamiento de un sistema reconocedor de voz

La formula (8) determina los procesos y componentes pertenecientes al diseño de un reconocedor de voz. Primero, el reconocedor necesita determinar el valor de *P(O|W)*, es decir, la probabilidad de que el hablante produjo *O*, dado que quiso decir W [Jelinek, 98] y segundo el valor de *P(W);* para determinar estos valores, se toman en cuenta dos etapas; en primer lugar se le proporciona al sistema una serie de pronunciaciones (elementos del habla: fonemas, alófonos, palabras, etc.), que se desea que éste almacene (aprenda o memorice); cabe destacar que no se almacenan las pronunciaciones en sí, sino propiedades de ese conjunto, éste proceso es llamado *entrenamiento*. En segundo lugar, se usa ese conocimiento adquirido para deducir la secuencia de unidades más probable dada una señal, es decir, el sistema identifica una pronunciación dada como alguna de las que ya conoce o parecida a las que tiene almacenada[3], a este proceso se le llama *decodificación o reconocimiento*.

En la Figura 3 se muestra el esquema general de un sistema de reconocimiento de voz, en el cuál se pueden observar los elementos generales que lo componen; como se puede ver, la señal de voz entra por el micrófono y se convierte en una señal analógica, la cual es digitalizada por el módulo de adquisición de datos, que además se encarga de almacenar los datos obtenidos de la conversión. La señal ya digitalizada pasa al módulo de extracción de características en donde se obtienen sus propiedades como son: energía espectral, tono, formantes, etc., correspondientes a una pronunciación; éste módulo se encarga de dividir

[3] La pronunciación a reconocer no es, necesariamente, una de las que se usan en la etapa de entrenamiento.

la secuencia de valores obtenida por el módulo de adquisición de datos en segmentos correspondientes a una duración de entre 10 y 35 milisegundos; es decir, se realiza una compresión de los datos para obtener un vector de propiedades de cada segmento y de cada sonido de la pronunciación, así la salida de este módulo comprende una secuencia de vectores de propiedades de los segmentos.

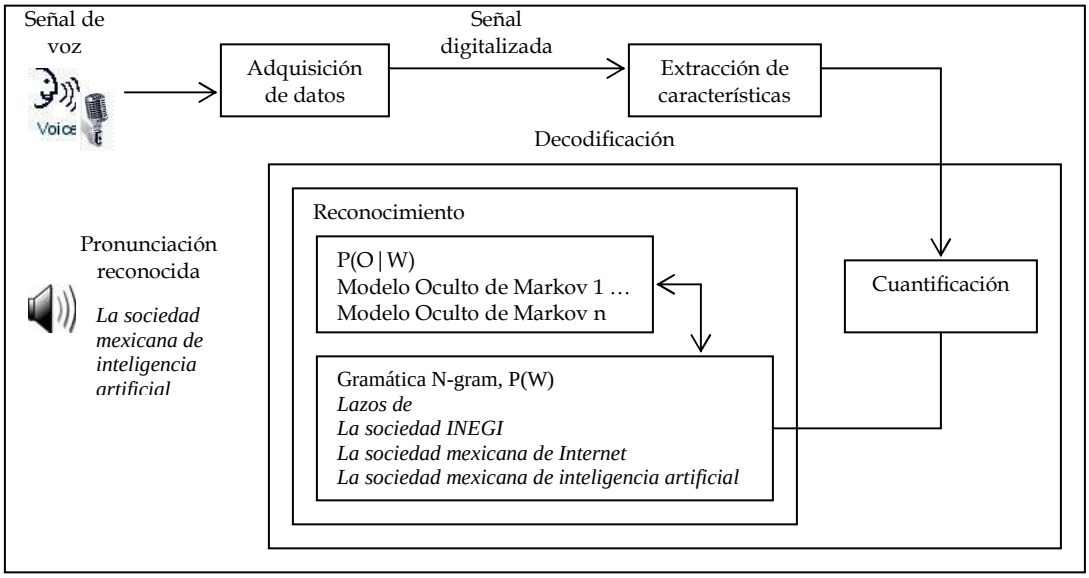


Figura 3. Diagrama a bloques de un sistema reconocedor de voz.

El módulo de decodificación encierra dos módulos: cuantificación y reconocimiento. El módulo de cuantificación identifica los distintos sonidos que están presentes en la pronunciación y asocia a cada sonido con un vector; obteniendo así como salida una secuencia de valores, donde cada valor representa el sonido con el que está asociado un vector de propiedades[4]. Por su parte, el módulo de reconocimiento identifica una pronunciación dada y la clasifica dentro de sus modelos como conocida o parecida a una conocida, o bien como desconocida. Para ello recibe desde el módulo de cuantificación, la secuencia de valores que corresponde a un conjunto de sonidos que puede tratar el sistema; estos sonidos, individualmente corresponden a un segmento de la señal de la voz pero, en conjunto y en la secuencia correcta constituyen la señal completa de la pronunciación que se desea reconocer. La complejidad de este módulo depende del tipo de identificación que se requiera. Por ejemplo, un reconocedor de gramáticas será más

4 Un mismo valor o un mismo sonido, puede aparecer varias veces en esta secuencia de salida.

complejo que un reconocedor de palabras y uno de palabras será más complejo que uno de letras o de fonemas [Maldonado, 98].

En el capitulo tres se aborda más a fondo este proceso, en el cuál intervienen tres elementos importantes en el reconocimiento de la voz: el modelo del lenguaje, los modelos acústicos y el diccionario de pronunciación, los cuales realizan el trabajo fuerte en el modulo de decodificación, ya que son los que generan la hipótesis de la frase a reconocer.

Capítulo 2

Corpus DIMEx100

En México como en otros países de habla hispana es poco el trabajo que se ha hecho en relación a las tecnologías del habla. Esta situación da como resultado que los programas y recursos se importen, y los trabajos lingüísticos sobre síntesis y reconocimiento de habla se copien de la fonética inglesa o en su mejor caso de la española; por ello es necesario desarrollar programas de tecnologías del habla para el español de México, así como la incorporación de conocimientos lingüísticos en el mismo.

Además tanto los sistemas de texto a voz (TTSs) como los de voz a texto (ASR) necesitan ser enfocados a comunidades lingüísticas específicas, esto es porque si se consideran los modelos acústicos para los alófonos más comunes de la lengua se puede incrementar el reconocimiento. Otro punto importante es que con el conocimiento empírico de estos alófonos se pueden crear diccionarios de pronunciación mucho más ricos que representen a la lengua de manera más precisa. Por ello es necesaria la creación de corpora fonéticos para el entrenamiento de modelos acústicos para el español de México.

Así, debido a que existen pocos recursos en español de México, se decidió utilizar el corpus DIMEx100 [Pineda *et al.*, 04], ya que es un corpus creado en español de México, además de que su etiquetación está basada en un exhaustivo análisis fonético (Cuétara, 2004). Asimismo este reconocedor de voz servirá para evaluar uno de los niveles de etiquetación del corpus DIMEx100, los cuáles se describirán en el siguiente capítulo.

2.1 Los sonidos del lenguaje hablado

Un fonema es una unidad abstracta que corresponde a una unidad básica de significado a nivel fonético de la lengua; estas unidades de contraste pueden ser pronunciadas de diferente forma dependiendo del dialecto o incluso por un hablante en particular, a estas realizaciones se les denominan *alófonos* y estas son las unidades observables de manera empírica en el análisis fonético. Debido a esta variación y con el fin de permitir la comunicación entre diferentes dialectos, los alfabetos textuales contienen un símbolo que hace alusión a cada fonema, aunque cada uno de estos pueda tener varias realizaciones fonéticas. Así, por ejemplo el fonema ***s*** comúnmente corresponde a la letra ***s*** del alfabeto, aunque pueda ser pronunciada de diferentes formas siempre que se pronuncie podrá ser identificada como tal. El español cuenta con 22 fonemas (17 consonantes y 5 vocales), los cuales en el español de México tienen una pronunciación básica, ver tablas 1 y 2.

Las unidades fonéticas se producen por la modificación del aire proveniente de los pulmones, en esta modificación intervienen varios órganos como son: la boca, garganta y nariz; principalmente el llamado *aparato fonador* que está compuesto por los dientes, la lengua, el paladar, etc.

Las unidades fonéticas pueden ser clasificados de acuerdo a como se da la obstrucción del aire en consonantes y vocales. Las consonantes son producidas por la restricción o bloqueo del aire y pueden ser sonoras o sordas, mientras que las vocales tienen menos obstrucción y generalmente son sonoras. La sonoridad se obtiene de las cuerdas vocales; si estas vibran los sonidos son sonoros, de lo contrario, son sordos.

Consonantes	Labiales	Labiodentales	Dentales	Alveolares	Palatales	Velares
Oclusivas sordas	p *pepe*		t *tío*			k *camino*
Oclusivas sonoras	b *bien*		d *diente*			g *gato*
Africada sorda					tS *hecho*	
Fricativas sordas		f *foco*		s *sol*		x *paja*
Fricativa sonora					Z *ayer*	
Nasales	m *más*			n *nana*	n~ *año*	
Vibrantes				r(/r *pero/perro*		
Lateral				l *lalo*		

Tabla 1. Fonemas consonánticos del español de México.

Vocales	Anteriores	Central	Posteriores
Cerradas	i *ahí*		u *su*
Medias	e *meta*		o *lo*
Abiertas		a *la*	

Tabla 2. Fonemas vocálicos del español de México.

Las consonantes se distinguen por dos características: su punto de articulación -en donde se hace la restricción del aire - y su modo de articulación - como es hecha la restricción del aire, si se bloqueo por completo, o solo una parte .

Por su punto de articulación se dividen en:

Labial: los dos labios obstaculizan el sonido.

Labiodental: contacto de los dientes superiores con el labio inferior.

Dental: contacto del ápice de la lengua con la parte inferior de los dientes incisivos superiores.

Alveolar: contacto de la lámina con los alvéolos mientras el ápice toca los incisivos inferiores.

Palatal: la lengua toca el paladar.

Velar: la parte dorsal de la lengua se eleva hacia el velo del paladar.

Por su modo de articulación se dividen en:

Oclusivo: Obstáculo total del aire.

Fricativo: Obstáculo parcial del aire.

Africado: Obstáculo total + obstáculo parcial del aire.

Nasal: Usa la cavidad nasal como resonador.

Vibrante: La lengua toca repetidas veces los alvéolos.

Lateral: El aire se escapa por los lados de la lengua.

Para clasificar a las vocales se toman en cuenta los mismos factores que con las consonantes; en este caso el punto de articulación se refiere a la parte de la boca donde se articulan, y pueden ser anteriores, medio o central y posteriores. El modo de articulación se refiere a la abertura de la boca al pronunciarlos, y pueden ser de abertura máxima o abierta, de abertura media o semiabiertos y de abertura mínima o cerrada.

El trabajo empírico realizado con el lenguaje del centro del país [Cuétara, 04], ha demostrado que hay 37 alófonos (26 sonidos consonánticos, 9 vocálicos y 2 semi-consonánticos), que aparecen frecuentemente y de manera sistemática en la lengua hablada; por ello sus características acústicas merecen la definición de un modelo acústico para cada unidad del conjunto; este conjunto se puede observar en las tablas 3 y 4.

Consonantes	Labiales	Labiodentales	Dentales	Alveolares	Palatales	Velares
Oclusivas sordas	p *pepe*		t *tío*		k_j *kilo*	k *camino*
Oclusivas sonoras	b *bien*		d *diente*			g *gato*
Africada sorda					tS *hecho*	
Africada sonora					dZ *lluvia*	
Fricativas sordas		f *foco*	s_[*asta*	s *sol*		x *paja*
Fricativa sonora				z *mismo*	Z *ayer*	
Aproximantes	V *haba*		D *hada*			G *el gato*
Nasales	m *más*		n_[*antes*	n *nana*	n~ *año*	N *ángel*
Vibrantes				r(/r *pero/perro*		
Lateral				l *lalo*		

Tabla 3. Alófonos consonánticos del Mexbet (Cuétara, 2004).

Vocales	Anteriores	Central	Posteriores
Paravocales	j *viene*		w *suave*
Cerradas	i *ahí*		u *su*
Medias	e *meta*		o *lo*
Medias abiertas	E *erre*		O *sol*
Abiertas	a_j *aire*	a *la*	a_2 *alma*

Tabla 4. Alófonos vocálicos del Mexbet (Cuétara, 2004).

Algunos corpus para el español de México, como el Tlatoa (Kirshning, 2001) y Gamboa (2002), solo han considerado los 22 fonemas del lenguaje, por ello tienen conflictos en la transcripción de algunos sonidos consonánticos, por ejemplo la *y* en *a**y**er* - y en sonidos semi-consonánticos como *j* y *w*. En nuestro caso, el alfabeto fonético Mexbet no tiene problemas con la representación de estos sonidos, para mayor referencia ver [Cuétara, 04].

2.2 Características del corpus DIMEx100

El corpus DIMEx100 [Pineda *et al.*, 04], fue diseñado y colectado con el objetivo de servir como un recurso fonético para el desarrollo de tecnologías del habla, especialmente para el reconocimiento de voz; además provee una base empírica para estudios fonéticos del español de México.

El corpus DIMEx100 está formado por un conjunto de 5010 frases seleccionadas del Corpus230 (Villaseñor *et al.*, 2004), el cual cuenta con 344,619 frases, 235,891 unidades léxicas y 15 millones de palabras recolectadas de Internet; estas frases estaban ordenadas de menor a mayor valor de perplejidad. La perplejidad es una medida del número de unidades lingüísticas que pueden seguir a una unidad de referencia en relación al corpus; intuitivamente el valor más bajo de perplejidad de una palabra es el menor número de palabras diferentes que es probable que la sigan en una oración. La perplejidad de una oración puede ser definida como una función de la perplejidad de los constituyentes de las palabras; de acuerdo con esto, las oraciones con un valor de perplejidad bajo están constituidas por palabras con un alto poder discriminatorio o con un alto contenido de información; estas oraciones son las que conforman al corpus DIMEx100.

El corpus DIMEx100 fue editado manualmente eliminándose palabras extranjeras y abreviaciones de modo que las frases fueran de fácil lectura; estas frases fueron grabadas por 100 personas, cada uno grabó 50 frases diferentes y 10 frases en común, para dar un total de 6000 frases repartidas en 100 carpetas. También se controlaron las características de los hablantes, se midió la cantidad y distribución de muestras por cada unidad fonética,

y se verificó que éstos abarcaran todo el conjunto alofónico y estuviera balanceado en lo referente a la lengua [Pineda *et al*., 04; Cuétara,04, Pineda *et al*., 07].

Las grabaciones se realizaron en el Centro de Ciencias Aplicadas y Desarrollo Tecnológico (CCADET-UNAM), en una cabina insonorizada, con el software *Wave Lab,* una tarjeta de sonido *Sound Blaster Audigy Platinum ex* (24 bit/96KHz/100db SNR) y un micrófono de condensación con diafragma sencillo. Cada frase fue grabada en un formato mono estéreo a 16 bits y con un periodo de muestreo de 44.1 Khz.

2.2.1 Características lingüísticas del corpus

Como se vio en el punto 2.2, el objetivo de hacer la trascripción alofónica es obtener la descripción del contenido fonético de las frases del corpus; adicionalmente se identificó el contexto en el cuál ocurren (tabla 5); este contexto se ha caracterizado a través de reglas fonológicas. Por ejemplo, la vocal /a/, puede ser realizada como velar ante otro sonido velar y ante /l/ en coda silábica, como *alto* => *[a_2lt_cto]*; la palatal /a/, es realizada ante sonidos palatales como *año* => [a_jn~o], y la central /a/ en cualquier otro lugar; en la tabla 6 se presentan los símbolos que representan estas reglas, por ejemplo, inicio y final de palabra, etc., utilizados en la columna de contexto de la tabla 5.

Se puede ver entonces, que la variación alofónica del español de México puede ser modelada con reglas fonológicas como lo apoyan Moreno y Mariño [Cuétara, 04].

Fonemas	Alófonos	Contexto
Bilabial oclusiva sorda p		
/p/	p_c p	En todos los casos
Dental oclusiva sorda t		
/t/	t_c t	En todos los casos
Velar oclusiva sorda k		
/k/	k_c k_j	_{e, i, j}
/k/	k_c k	En todos los demás casos
Bilabial oclusiva sonora b		
/b/	b_c b	///_
/b/	b_c b	{m, n}_
/b/	V	En todos los demás casos
Dental oclusiva sonora d		
/d/	d_c d	///_
/d/	d_c d	{m, n}_
/d/	D	En todos los demás casos
Velar oclusiva sonora g		
/g/	g_c g	///_
/g/	g_c g	{m, n}_
/g/	G	En todos los demás casos
Palatal africada sorda tS		
/tS/	tS_c tS	En todos los casos
Labiodental fricativa f		
/f/	f	En todos los casos
Alveolar fricativa sorda s		
/s/	z	V_V
/s/	z	_{b, d, g, Z, m, n, n~, l, r, r(}
/s/	s_[	_{t}
/s/	s	En todos los demás casos
Velar fricativa sorda x		
/x/	x	En todos los casos
Palatal fricativa sonora Z		
/Z/	dZ_c dZ	///_
/Z/	dZ_c dZ	{m, n}_
/Z/	Z	En todos los demás casos
Nasal bilabial m		
/m/	m	En todos los casos
Nasal alveolar n		
/n/	n_[	_{t, d}
/n/	N	_{k, g}
/n/	n	En todos los demás casos

Nasal palatal n~

/n~/	n~	En todos los casos

Lateral alveolar l

/l/	l	En todos los casos

Vibrante simple r(

/r(/	r(	En todos los casos

Vibrante múltiple r

/r/	r	En todos los casos

Vocal alta palatal i

/i/	j	_{a, e, o, u}
/i/	j	{a, e, o, u}_
/i/	i	En todos los demás casos

Vocal media palatal e

/e/	E	_{r}
/e/	E	{r}_
/e/	E	_{p, t, k, b, g, d, tS, f, x, Z, l, r(}$
/e/	e	En todos los demás casos

Vocal abierta a

/a/	a_2	_{u, x}
/a/	a_2	_{l}$
/a/	a_j	_{tS, n~, Z, j}
/a/	a	En todos los demás casos

Vocal media velar o

/o/	O	_{r}
/o/	O	{r}_
/o/	O	_{consonante}$
/o/	o	En todos los demás casos

Vocal alta velar u

/u/	w	_{a, e, o, i}
/u/	w	{a, e, o, i}_
/u/	u	En todos los demás casos

Tabla 5. Fonemas y Alófonos del español de México (Cuétara, 2004).

Símbolo	Significado
///_	Inicio absoluto de palabra
_{}	Posición anterior
{}_	Posición posterior
v_v	Intervocálico
{}$	Final de sílaba

Tabla 6. Significado de los contextos.

2.2.2 Características socio-lingüísticas de los hablantes

En la grabación de un corpus oral se deben de considerar un mínimo de características de los hablantes, para su posterior evaluación (Perissinotto, 1975); siguiendo estas recomendaciones, los hablantes fueron seleccionados de acuerdo a su edad (de 16 a 36 años de edad), nivel de educación (con estudios mínimos de preparatoria o equivalente) y lugar de origen (Ciudad de México).

Para el caso particular del corpus DIMEx100, la mayor parte de los hablantes fueron investigadores, estudiantes, profesores y trabajadores de la UNAM, y el rango de edad fue de 23 años. El 87% de los hablantes eran estudiantes de licenciatura y el resto ya estaban graduados; el 82% de los hablantes nacieron y vivían en la ciudad de México. Se aceptaron 18 personas de otros lugares, con residencia en la ciudad de México para participar en las grabaciones. El corpus resultó balanceado en género ya que el 49% fueron hombres y el 51% mujeres. Aún cuando el español de México tiene varios dialectos, el predominante en el corpus DIMEx100 es el que corresponde al de la ciudad de México.

2.3 Niveles de etiquetación

El corpus DIMEx100 se etiqueta en dos niveles de representación de la lengua: un nivel T54, que corresponde a la segmentación alofónica del inventario del español de México, más sus respectivos acentos, y un nivel Tp que corresponde a la representación ortográfica de las palabras. Además existen otros dos niveles de etiquetación derivados del nivel T54: el nivel T44 y el nivel T22. En el nivel de segmentación T44 se consideran principios acústicos básicos, incluyendo las codas silábicas, mientras que en el nivel de segmentación T22 solamente se consideran los 22 fonemas básicos del español de México.

La herramienta que se utiliza para el etiquetado del corpus DIMEx100 es SpeechView, que forma parte del toolkit CSLU (Center for Spoken Language and Understanding) de la

universidad de Oregon[1]. Esta herramienta permite ver el espectrograma, oscilograma y pitch de una señal acústica (Figura 4).

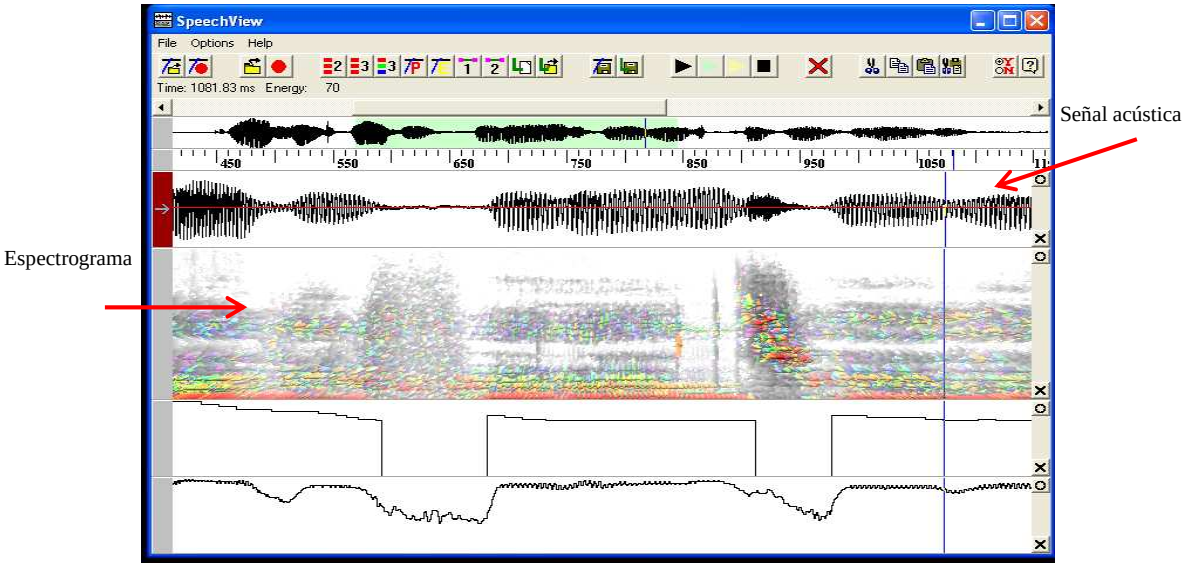


Figura 1. Speech View.

En el nivel T54, se busca tener la segmentación más fina con el fin de obtener más datos acústicos, en este nivel se tienen los 37 alófonos del Mexbet, más 8 cierres de oclusivas y africadas (p_c, t_c, k_c, b_c, d_c, g_c, tS_c,dZ_c), más 9 vocales que pueden recibir acentos (i_7, e_7, E_7, a_j_7, a_7, a_2_7, O_7, o_7 y u_7). Se consideran fenómenos como acentuación y asimilación de sonidos (en un sentido restringido, "asimilación" consiste en la conversión de un fonema en otro por la influencia del que le antecede o del que le precede). En las tablas 7, 8 y 9 se presenta el inventario de alófonos del español de México.

[1] http://www.cslu.ogi.edu/

Consonantes	Labiales	Labiodentales	Dentales	Alveolares	Palatales	Velares
Oclusivas sordas	p/p_c		t/t_c		k_j	k/k_c
Oclusivas sonoras	b/b_c		d/d_c			g/g_c
Africada sorda					tS_/tS_c	
Africada sonora					dZ/dZ_c	
Fricativas sordas		f	s_[	s		x
Fricativa sonora				z	Z	
Aproximantes	V		D			G
Nasales	m		n_[	n	n~	
Vibrantes				r(/r		N
Lateral				l		

Tabla 7. Consonantes del nivel T54.

Vocales	Anteriores	Central	Posteriores
Paravocales	j		w
Cerradas	i		u
Medias	e		o
Medias abiertas	E		O
Abiertas	a_j	a	a_2

Tabla 8. Vocales atonas del nivel T54.

Vocales	Anteriores	Central	Posteriores
Cerradas	i_7		u_7
Medias	e_7		o_7
Medias abiertas	E_7		O_7
Abiertas	a_j_7	a_7	a_2_7

Tabla 9. Vocales tónicas del nivel T54.

El nivel T44 se considera una transcripción media, e incluye, además de los 22 alófonos prototípicos del español de México, los cierres de las consonantes oclusivas y de la africada sorda ([p_c, t_c, k_c, b_c, d_c, g_c, tS_c]), los alófonos aproximantes de las oclusivas sonoras ([V, D, G]), las 9 vocales que pueden recibir acento ([i_7, e_7, E_7, a_j_7,

a_7, a_2_7, O_7, o_7, u_7]) y las paravocales ([j, w]). Asimismo, se marca un único símbolo para parejas de consonantes ([p/b, t/d, k/g, n/m, r(/r]) en posición final de sílaba o coda silábica ([-B, -D, -G, -N, -R]); el inventario completo se muestra en la tablas 10, 11, 12 y 13.

Consonantes	Labiales	Labiodentales	Dentales	Alveolares	Palatales	Velares
Oclusivas sordas	p/p_c		t/t_c			k/k_c
Oclusivas sonoras	b/b_c		d/d_c			g/g_c
Africada sorda					tS_/tS_c	
Africada sonora					dZ/dZ_c	
Fricativas sordas		f		s		x
Fricativa sonora					Z	
Aproximantes	V		D			G
Nasales	m			n	n~	
Vibrantes				r(/r		
Lateral				l		

Tabla 10. Consonantes del nivel T44.

Vocales	Anteriores	Central	Posteriores
Paravocales	j		w
Cerradas	i		u
Medias	e		o
Abiertas		a	

Tabla 11. Vocales atonas del nivel T44.

Vocales	Anteriores	Central	Posteriores
Cerradas	i_7		u_7
Medias	e_7		o_7
Abiertas		a_7	

Tabla 12. Vocales tónicas del nivel T44.

	Coda Silábica
Labiales p/b	-B
Dentales t/d	-D
Velares k/g	-G
Nasales n/m	-N
Vibrantes r(/r	-R

Tabla 13. Codas silábicas del nivel T44.

En el nivel T22 se representa única y exclusivamente las 22 formas alofónicas que se relacionan con los fonemas del español de México; el inventario completo se muestra en las tablas 1 y 2 de la sección 2.1.

2.4 Estadísticas del corpus

2.4.1 Estadísticas Generales del Corpus DIMEx100

El corpus DIMEx100 cuenta con 6,000 frases con un total de 59,812 palabras de las cuales 8,715 son diferentes. En la Figura 5 se muestran las palabras más representadas en el corpus DIMEx100.

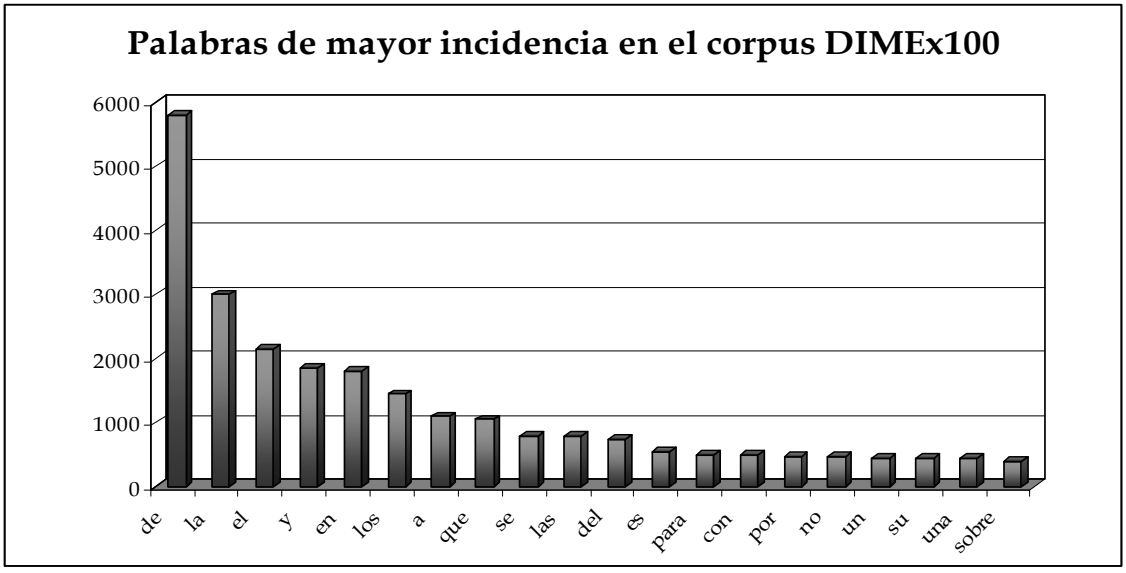


Figura 2. Palabras de mayor incidencia en el corpus DIMEx100.

2.4.2 Estadísticas del 50% del Corpus DIMEx100

Para realizar el reconocedor de voz se utilizó el 60% del corpus DIMEx100 etiquetado, del cual el 50% se utilizó para el proceso de entrenamiento y el 10% restante para realizar las pruebas.

El 50% del corpus DIMEx100 corresponde a 2,498 frases con un total de 25,990 palabras de las cuales 5,466 son diferentes. En las Figuras 6 y 7 se muestran las palabras que aparecen con más frecuencia en el 50% del corpus DIMEx100, es decir, en los datos que se utilizaron para el entrenamiento.

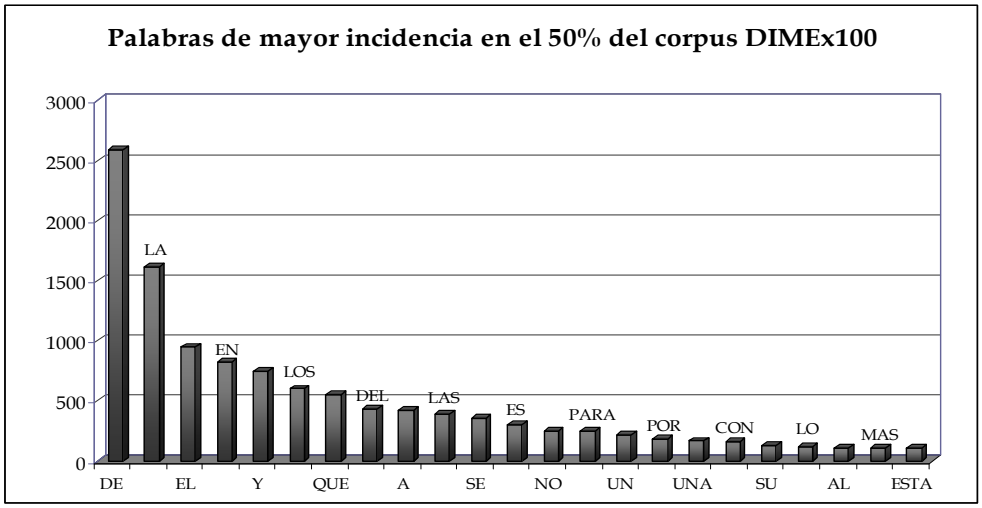


Figura 3. Palabras de mayor incidencia en el 50% del corpus DIMEx100 (parte1).

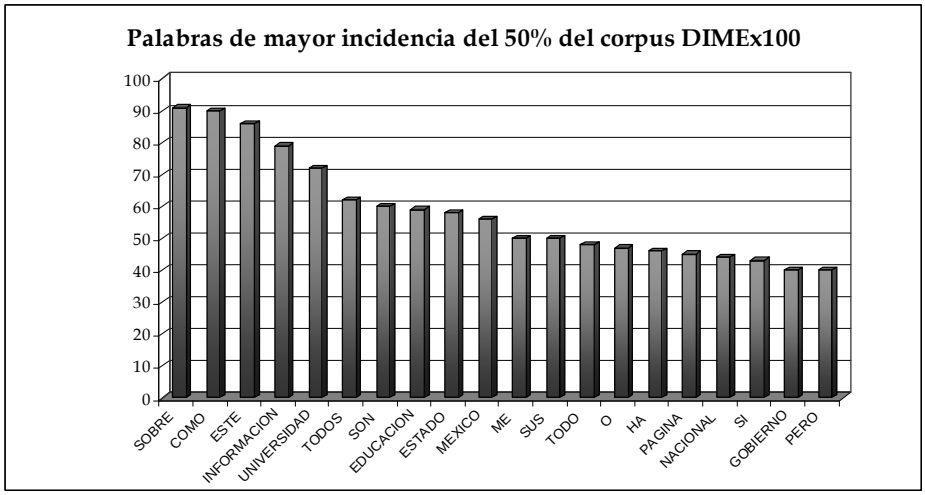


Figura 4. Palabras de mayor incidencia en el 50% del corpus DIMEx100 (parte2).

En las siguientes figuras se muestran las diferentes distribuciones fonéticas de acuerdo a cada nivel de etiquetación. Cabe destacar que el nivel de granularidad que se utiliza para el reconocimiento de voz influye de manera determinante en la construcción de los diccionarios de pronunciación; por ello para cada nivel de granularidad se mostrará además el número de pronunciaciones y de palabras que conforman el diccionario de pronunciación.

El diccionario de pronunciación creado con el nivel T22 cuenta con 6,668 palabras totales y con 5,466 palabras distintas. En la Figura 8 se muestra la distribución fonética y en la Figura 9 una gráfica con las palabras que tienen más pronunciaciones en este nivel.

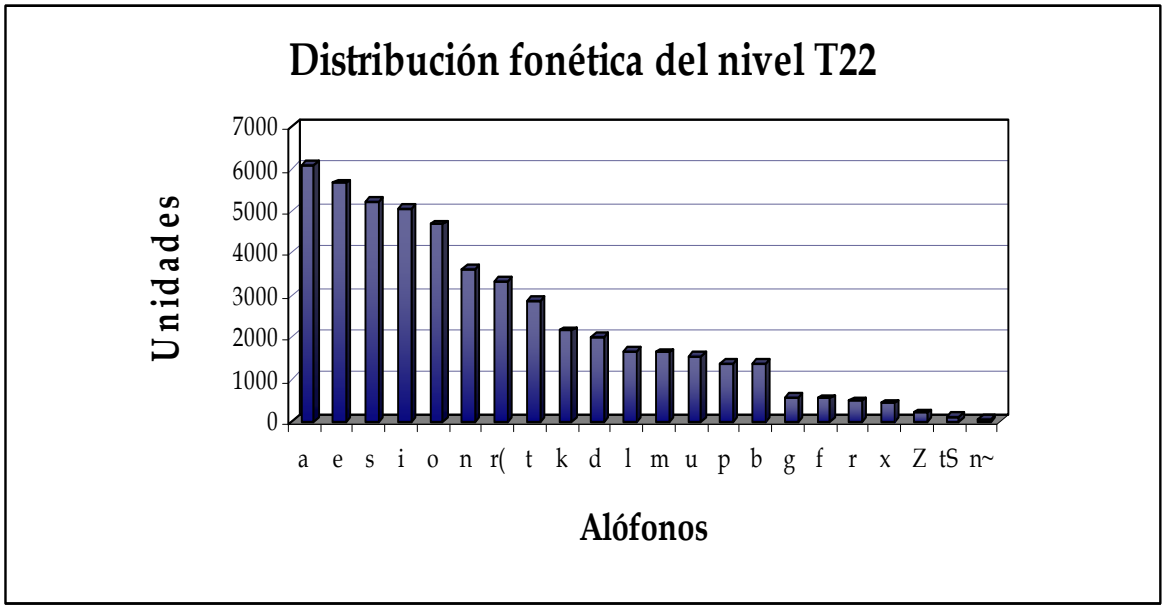


Figura 5. Distribución fonética del nivel T22.

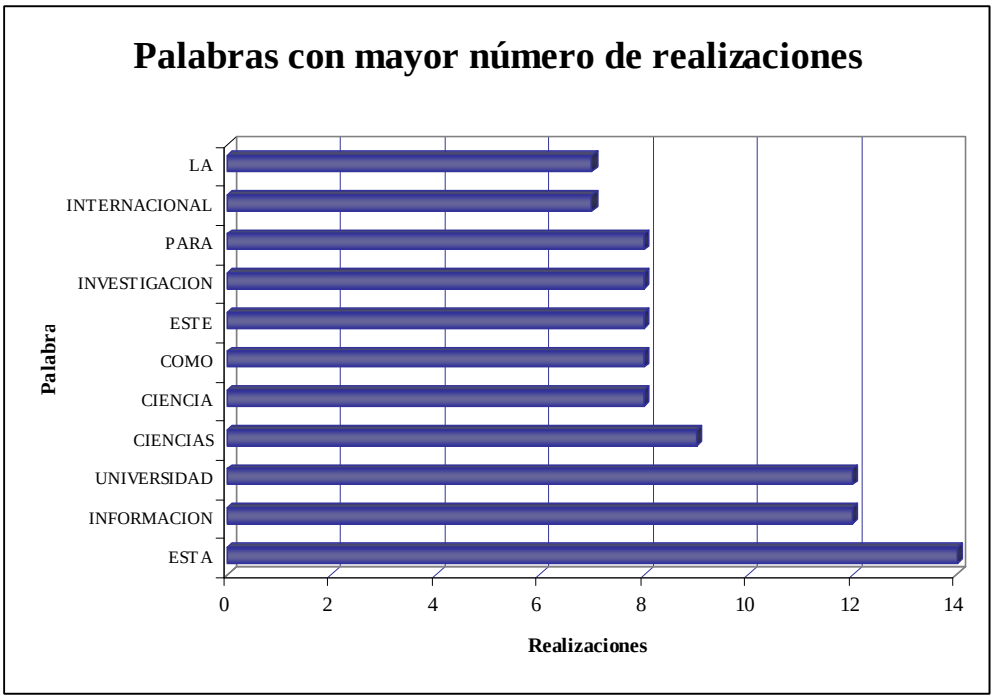


Figura 6. Palabras con mayor número de realizaciones del Nivel T22.

El diccionario de pronunciación creado con el nivel T44 cuenta con 8,449 pronunciaciones (palabras totales) y con 5,466 palabras distintas. En la Figura 11 se muestra una gráfica con las palabras que tuvieron un mayor número de pronunciaciones; como se puede ver la palabra universidad se pronuncio de 32 maneras diferentes según las estadísticas para este nivel en contraste con el número de realizaciones encontradas en el nivel T22, las cuales fueron 12.

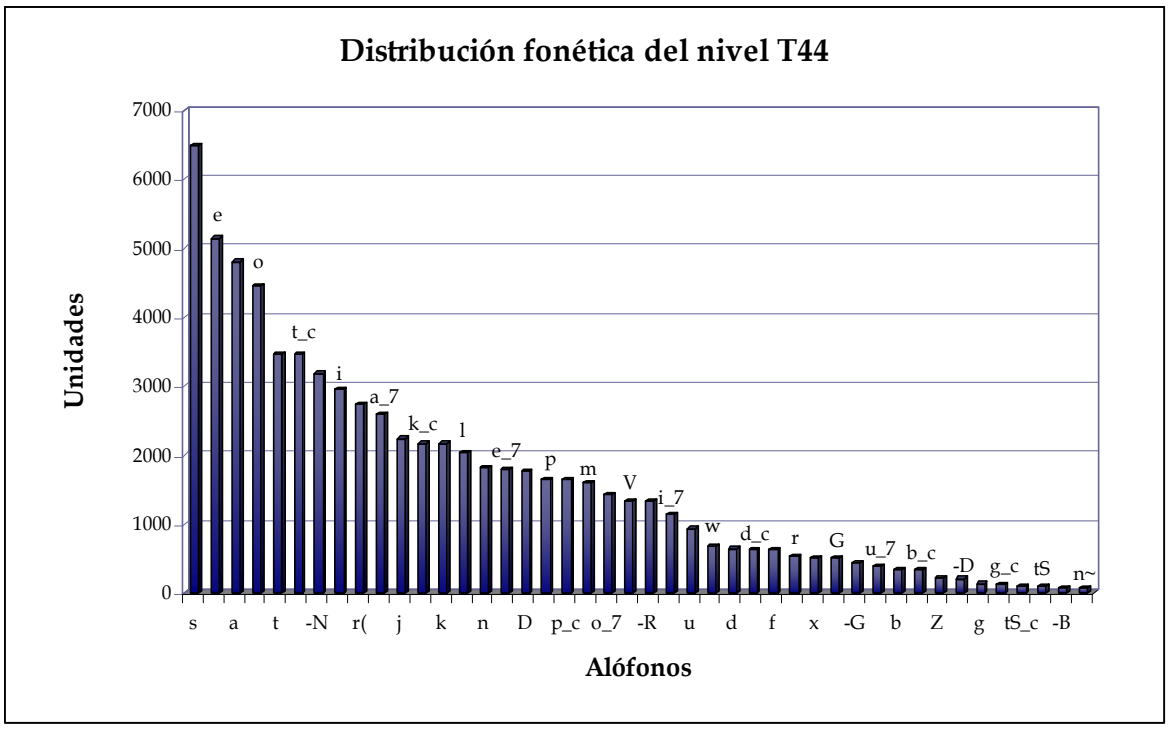


Figura 7. Distribución fonética del nivel T44.

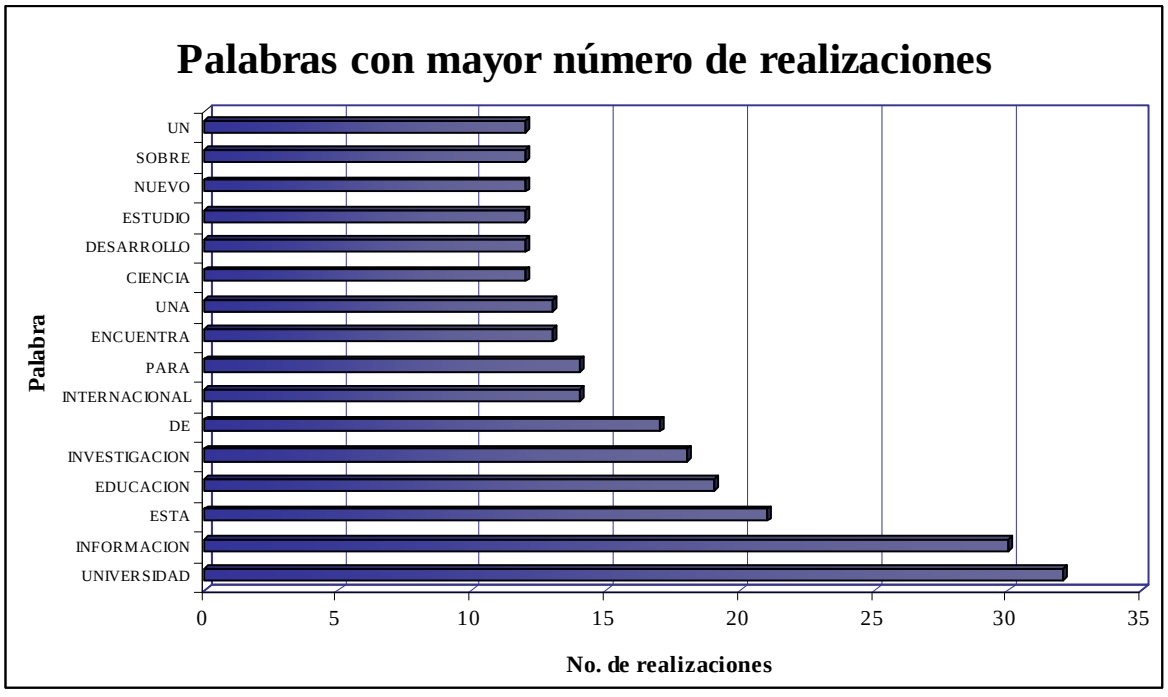


Figura 8. Palabras con mayor número de realizaciones del Nivel T44.

Este nivel cuenta con 54 unidades fonéticas, las cuales dieron lugar a un diccionario de pronunciación con 9,859 pronunciaciones y con 5,466 palabras distintas. En la Figura 14 se muestran las palabras que tuvieron un mayor número de pronunciaciones; por ejemplo se puede ver que a diferencia de los otros dos niveles la palabra información en este caso fue la que tuvo un mayor número de pronunciaciones.

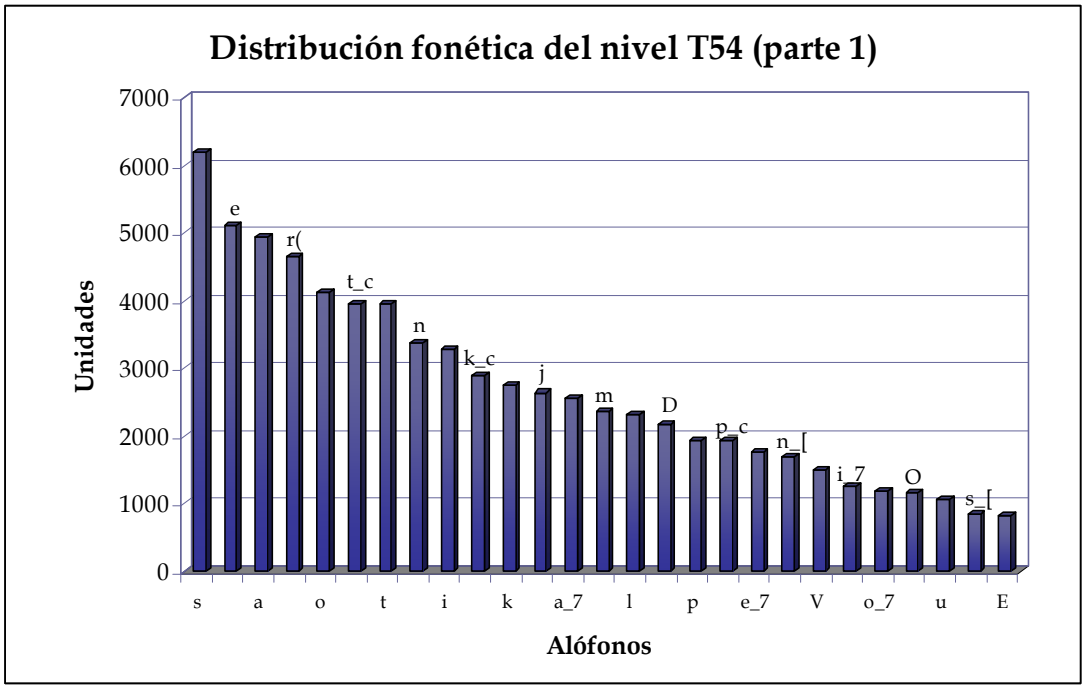


Figura 9. Distribución fonética del nivel T54 (parte1).

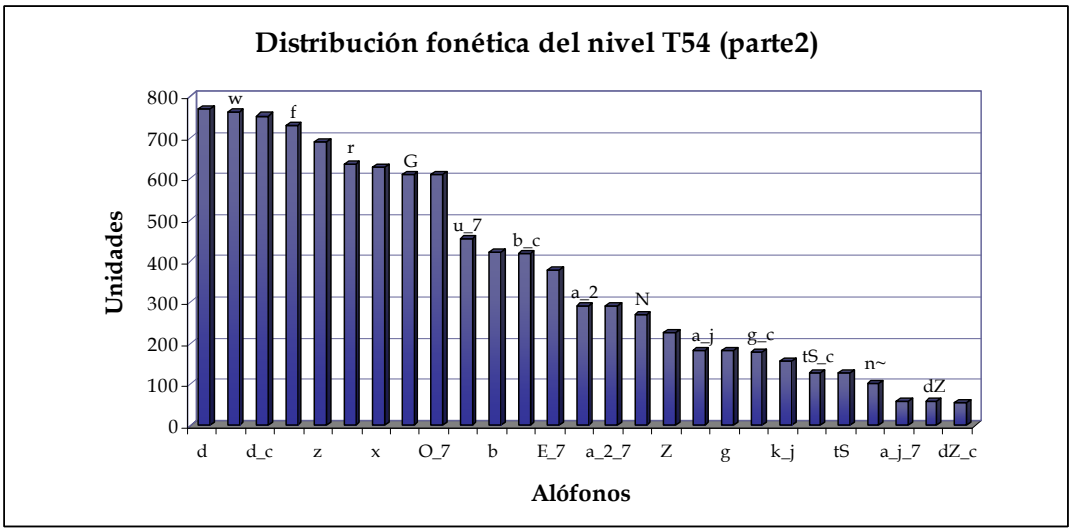


Figura 10. Distribución fonética del nivel T54 (parte2).

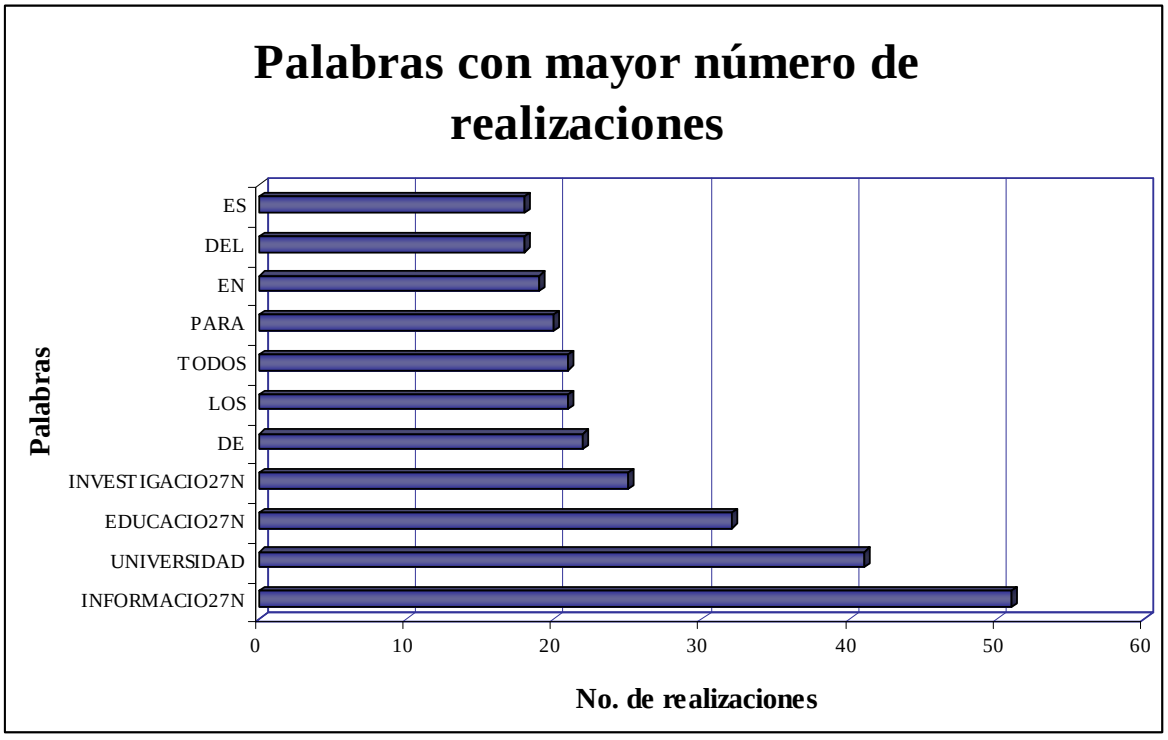


Figura 11. Palabras con mayor número de realizaciones del Nivel T54.

Para crear los modelos acústicos se requiere de 50 muestras de cada alófono como mínimo; en nuestro caso se puede observar en las gráficas, que en todos los niveles hay más de 50 muestras de cada alófono, por lo que es posible crear los modelos acústicos en cada nivel. Aunque en el nivel T54 sería recomendable realizar los modelos acústicos con un porcentaje mayor del corpus para obtener más muestras en este nivel.

Capítulo 3

Sistema reconocedor de voz

El objetivo de un sistema de reconocimiento de voz es traducir el habla a su representación textual, esto es, toma una señal acústica capturada por un micrófono, teléfono, etc., y la segmenta para obtener la secuencia de unidades fonéticas; para reconocer cada unidad fonética los modelos generados en la fase de entrenamiento tratan de generar la secuencia de observación; así, el modelo con la mayor probabilidad de haberlo generado gana, una vez que se obtiene la unidad fonética se busca dentro del diccionario de pronunciación, el cuál contiene la secuencia de unidades fonéticas correspondientes a una palabra. Una vez que se obtiene la palabra más parecida del diccionario de pronunciación, con ayuda del modelo del lenguaje se obtiene la secuencia de palabras más probable, es decir, la hipótesis de lo que se dijo. Cabe mencionar que todo este procedimiento se hace en paralelo, esto se ilustra en la Figura 15.

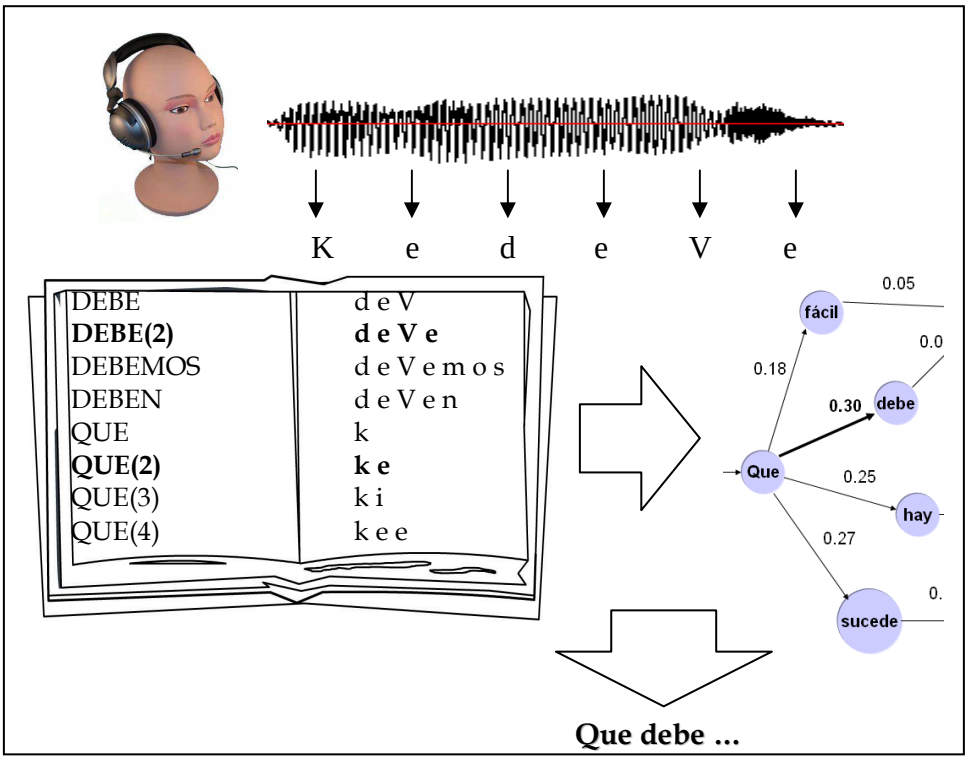


Figura 1. Funcionamiento de un sistema de reconocimiento de voz.

Como se puede apreciar en la figura anterior, existen tres elementos fundamentales para que un sistema de reconocimiento de voz lleve a cabo su función: los modelos acústicos, el diccionario de pronunciación y el modelo del lenguaje. En este capítulo se hablará a detalle de cada uno de estos elementos, así como del procedimiento de obtención de cada uno de ellos y de su función dentro del sistema.

3.1 Elementos de un reconocedor de voz

Como se vio en el primer capítulo de esta tesis, el proceso de reconocimiento de voz esta dado por la siguiente intuición matemática,

$$\hat{w} = \underset{W \in V}{argmax} \underbrace{P(O \mid W)}_{\text{Modelo acústico}} \underbrace{P(W)}_{\text{Modelo del lenguaje}}$$

Esta ecuación nos permite identificar cada uno de los procesos y componentes requeridos en el diseño de un reconocedor de voz; como se vio en la sección 1.3 de este trabajo - en donde se plantea que la elocución de entrada a pasado por un canal de comunicación ruidoso - el objetivo es entender como fue modificada la elocución original ***O*** para poder recuperarla. En la Figura 16 se puede ver la interacción de los distintos componentes de un sistema reconocedor de voz de acuerdo al esquema del canal ruidoso.

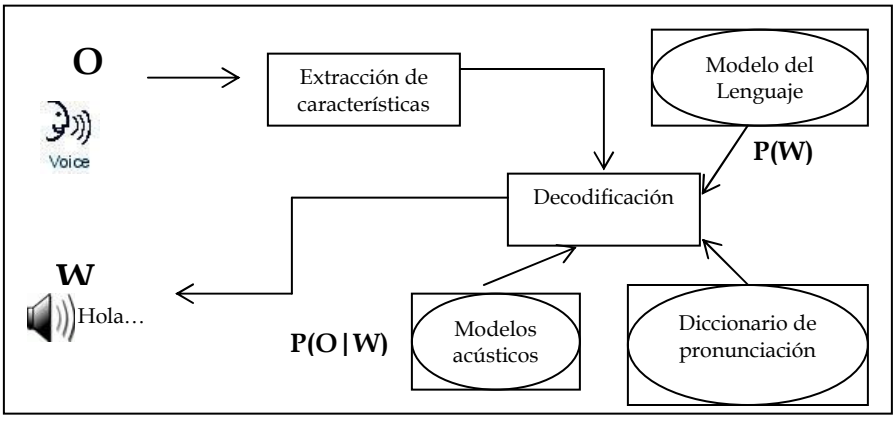


Figura 2. Sistema de reconocimiento de voz de acuerdo al canal ruidoso.

Este esquema describe de manera general como funciona un sistema reconocedor de voz, independientemente de la tecnología que se utilice. Además, se puede observar la interacción de los tres elementos básicos en la formulación básica de un sistema en ASR, cuyo objetivo es buscar la elocución más probable ***W*** dentro de todas las elocuciones posibles que se encuentran en un vocabulario dado. Esta interacción se puede describir de la siguiente manera:

1. ***Modelación acústica, P(O|W)***. Aquí se asignan probabilidades a las realizaciones acústicas de una secuencia de palabras. Para ello se crean modelos estadísticos de la señal acústica, ya sea de una palabra o de una subunidad (e.g., alófonos). Para crear estos modelos se necesita además contar con un diccionario de pronunciación, el cuál contiene todas las posibles pronunciaciones de cada palabra.

2. ***Modelo del lenguaje, P(W)***. Aquí se asignan probabilidades a las secuencias de palabras que forman oraciones validas en el lenguaje y además son consistentes con las tareas de reconocimiento. El entrenamiento del modelo del lenguaje puede ser hecho con secuencias genéricas de un texto o con la transcripción de diálogos específicos.

Una vez que están asignadas estas probabilidades se busca a través de todas las posibles secuencias de palabras en el lenguaje la secuencia de palabras con mayor probabilidad, así se obtiene la hipótesis de la elocución original.

Con la estructura del esquema de la Figura 16 como guía, a continuación se describe cada uno de los componentes y procesos pertenecientes a un sistema reconocedor de voz.

3.1.1 Modelos Acústicos

Como ya vimos para el entrenamiento de los modelos acústicos se necesita contar con dos elementos básicos:

1. Un conjunto de datos acústicos, los cuales están dados por un corpus del habla previamente etiquetado; éste puede ser grabado por uno o más hablantes.

2. Un conjunto de datos textuales, que corresponden a las transcripciones del corpus del habla. Así, utilizando las transcripciones del corpus y la etiquetación del mismo se crea el diccionario de pronunciación.

Para el caso particular de este trabajo se utilizó el corpus Dimex100, un nuevo corpus en español de México etiquetado manualmente. Antes de realizar el entrenamiento de los modelos acústicos se lleva a cabo el procesamiento de la señal, ya que con las características de esta se elaboran los modelos estadísticos de cada palabra o subunidad según sea el caso.

Para el análisis de la señal acústica, también llamado extracción de características, comúnmente se utilizan las técnicas de filtrado de LPC (Linear Predictive Coding), PLP (Perceptual Linear Predictive) y MFCC (Mel Frecuency Cepstral Coeficients). El objetivo en esta fase es extraer un conjunto de características destacadas de las propiedades espectrales[1] de cada sonido de la palabra o subunidad y que puede ser medido eficientemente. En la Figura 17 se muestra un diagrama a bloques, que resume este proceso.

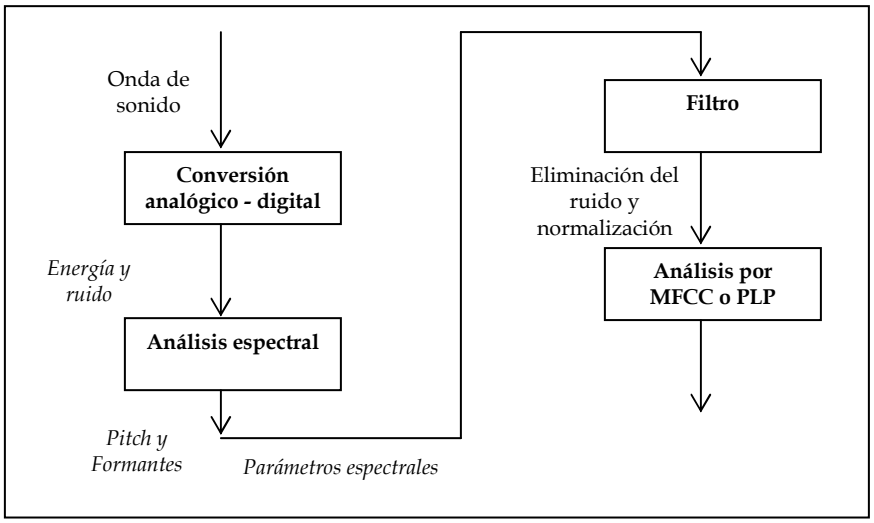


Figura 3. Diagrama a bloques del análisis de extracción de características.

[1] Una característica o propiedad espectral representa a la onda de sonido en términos de la distribución de las diferentes frecuencias con las que se caracteriza; así una distribución de frecuencias es llamada "espectro".

Cuando una señal acústica entra, primero se digitaliza. Este proceso de conversión analógico-digital pasa por dos fases: muestreo y cuantización; el muestreo de la señal se hace comúnmente entre los 8,000 Hz y 16,000 Hz.

Una vez que la señal acústica ha sido digitalizada, se obtienen sus propiedades espectrales, usando una Transformada de Fourier o el método LPC. Un espectro LPC se representa por un vector de características; este vector a su vez, contiene dos elementos: la frecuencia y la amplitud de cada formante[2] en un cierto instante de la señal del habla. Se pueden utilizar estas características directamente como símbolos de observación de un HMM, por ejemplo. Sin embargo se utilizan otras técnicas de procesamiento como PLP y MFCC.

En el análisis cepstral [Rabiner&Juang, 04], los coeficientes en escala lineal provenientes del análisis espectral son convertidos a coeficientes en escala MEL; estos son filtrados y sometidos a un análisis cepstral, el proceso finaliza con la extracción de características a través del cálculo de las derivadas de primer y segundo orden de los coeficientes MFCC. El análisis PLP es en esencia una combinación de las técnicas de la transformada discreta de Fourier y de predicción lineal.

Para la construcción de los modelos acústicos se utilizan diferentes métodos, dentro de los que se encuentran los métodos determinísticos y los estocásticos. En el modelado determinístico se utilizan técnicas de coincidencia de patrones acústicos donde se realiza la comparación del lenguaje hablado y el de referencia.

Un ejemplo de técnica de coincidencia de patrones acústicos es el método de alineamiento DTW (Dynamic Time-Warping) que busca encontrar el camino óptimo que minimice la distancia entre los patrones de prueba y de referencia. El problema que se presenta con el método de alineamiento DTW, es que cuando se pronuncia una palabra esta no siempre se realiza a la misma velocidad, lo que produce importantes distorsiones. La forma de resolver este problema es mediante programación dinámica.

[2] Se le llama formante a la resonancia proveniente del tracto vocal; estos se pueden ver en un espectrograma como barras negras horizontales, las cuales representan los picos espectrales de estas resonancias.

En contraste con el modelado determinístico, los modelos estocásticos caracterizan la variabilidad de la voz de forma inherente. Los métodos más utilizados son los modelos ocultos de Markov y las redes neuronales. Sin embargo los resultados que se obtienen en la actualidad con los HMM son mejores que los obtenidos con las redes neuronales [Faundez, 00].

Modelos Ocultos de Markov

Un HMM es un autómata de estados finitos utilizado para modelar la variabilidad espectral de cada uno de los sonidos del lenguaje. Un HMM tiene asociados dos procesos: un proceso de Markov no observado (oculto) y un proceso observado, cuyos estados son dependientes estocásticamente de los estados ocultos.

Si se trabaja con vocabularios grandes, se suelen modelar subunidades (por ejemplo, alófonos), mientras que si el vocabulario es pequeño, se suelen modelar palabras. La palabra o subunidad que será modelada es considerada como una secuencia de vectores, llamados observaciones.

Los HMM se utilizan durante la etapa de entrenamiento, en donde, el HMM aprende del conjunto de vectores de la observación de la palabra o subunidad; de esta manera durante el proceso de reconocimiento, a partir de una secuencia de observaciones de entrada, se evalúa cuál de todos los HMM es el que presenta mayor probabilidad de haber generado dichas observaciones, y éste corresponde al modelo de la palabra o subunidad a reconocer.

Se puede considerar que el HMM genera observaciones cada vez que salta de un estado a otro; esto es, cada vez que se produce una observación ocurre una transición entre los estados del modelo. La probabilidad de estas transiciones se representa mediante una matriz de transición entre estados $\boldsymbol{A}$:

$$A = \begin{pmatrix} a(1,1) & a(1,2) & \dots & a(1,s-1) & a(1,s) \\ \dots & & & & \\ & & a(i,j) & & \\ & & & \dots & \\ a(s,1) & a(s,2) & \dots & a(s,s-1) & a(s,s) \end{pmatrix}$$

Se asume que las probabilidades de transición entre estados no varían con el tiempo, de tal forma que el valor de ***a(i,j)*** no depende del instante ***t*** en el que ocurre la transición; además la suma de todos los elementos de cualquiera de las columnas de ***A*** vale 1; esto es porque la probabilidad de transición de un estado a otro no depende de la historia pasada de las transiciones entre estados, sino únicamente del estado actual y futuro.

Asimismo, la probabilidad de que se emita algún símbolo ***o*** del alfabeto en un cierto instante de tiempo ***t*** esta dado por la matriz de emisiones $\boldsymbol{B = bi(o_t)}$. Además, la probabilidad de distribución inicial sobre los estados $\boldsymbol{\pi}$, tal que $\boldsymbol{\pi_i}$ es la probabilidad de que el HMM empiece en el estado i_t, junto con las probabilidades de transición especifican la probabilidad de estar en cualquier estado en cualquier instante de tiempo.

Un HMM puede ser representado como un grafo dirigido de transiciones/emisiones [Faundez, 00]; esta representación se conoce como arquitectura de un HMM. Las arquitecturas más usadas son:

1. Ergódicas o completamente conectadas en las cuales cada estado del modelo puede ser alcanzado desde cualquier otro estado en un número finito de pasos.

2. Izquierda-derecha, hacia adelante o Bakis, los estados del sistema van de izquierda a derecha (Figura 18). En secuencias biológicas y en reconocimiento de la voz estas arquitecturas modelan bien los aspectos lineales de las secuencias [Pava, 05].

3. Izquierda-derecha paralelas, son dos arquitecturas izquierda-derecha conectadas entre sí (Figura 19).

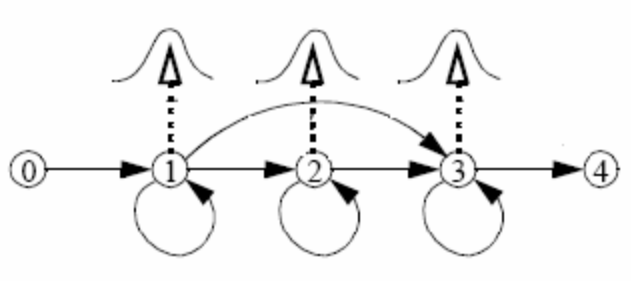


Figura 4. Modelo oculto de Markov (izquierda-derecha).

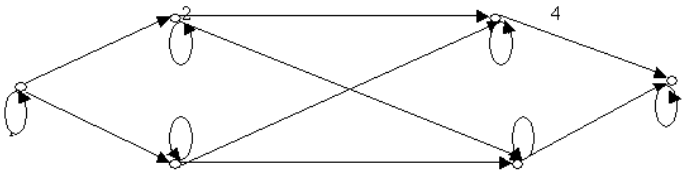


Figura 5. Modelo oculto de Markov paralelo (izquierda-derecha) con 6 estados.

Formalmente, un HMM es especificado como una quintupla (S,K,Π,A,B), en donde S y K son el conjunto de estados y el alfabeto de salida, y Π, A, y B son las probabilidades para el estado inicial, transiciones de estado, y símbolos de emisión, respectivamente.

Existen tres problemas básicos relacionados con los HMM's:

1. Calcular eficientemente P(O|W) la probabilidad de la secuencia de observación O dado el modelo W = (A,B,π) y la secuencia de observación O = (o_1 o_2 ...o_N) .

2. Encontrar la trayectoria más probable q = (q_1q_2...q_N) dado el modelo W y la secuencia de observación O = (o_1 o_2 ...o_N).

3. Encontrar los parámetros del modelo W=(A,B,π) que maximicen las probabilidades de las observaciones, P(O|W).

La forma más simple de resolver el problema 1 consiste en enumerar todas las posibles secuencias de estado de longitud T, en donde en cada tiempo t = 1,2, ...,T se tienen N posibles estados alcanzables, por lo tanto N^T operaciones. Sin embargo, existe un

procedimiento más eficiente para calcular dicha probabilidad, denominado *algoritmo de avance (forward algorithm)*.

En este algoritmo la probabilidad de la secuencia de observación parcial o_1, o_2...o_t en el estado *i* hasta el tiempo *t*, dado el modelo W de N (N +1)(T -1)+N multiplicaciones y N (N -1)(T -1) sumas, entonces resulta N^2T, que es mucho más fácil de computar.

Para el aprendizaje del HMM se deben propagar las probabilidades en forma inversa. El algoritmo de retroceso (backward algorithm) es la versión inversa del algoritmo de forward y se define como la probabilidad de la secuencia de observación parcial desde *t+1* hasta el final, dado el estado *i* en el tiempo *t* y el modelo W es del orden de N^2T operaciones.

La ruta más probable en un HMM es útil para el aprendizaje y para el alineamiento de secuencias con el modelo. La ruta más probable $q = (q_1\ q_2\ ...\ q_T)$ para la secuencia de observación $O = (o_1\ o_2\ ...\ o_T)$ puede ser calculada utilizando el algoritmo de Viterbi. El algoritmo de Viterbi está basado en el método de la programación dinámica.

Finalmente, el problema más difícil de los HMMs es determinar un método para ajustar los parámetros (A,B,π) del modelo para satisfacer los criterios de optimación. No se conoce una forma analítica para fijar los parámetros que maximice la probabilidad de la secuencia de observación. Varios algoritmos están disponibles para el entrenamiento de un HMM, entre ellos, Baum-Welch o EM (Expectation Maximization), GEM (EM Generalizado) y diferentes formas de descenso por gradiente.

El modelado con HMM es usado por muchas herramientas dedicadas al reconocimiento de voz - SPHINX, HTK, CSLU, solo por citar algunas – con las cuales se obtienen muy buenos resultados y además son las más usadas para el reconocimiento de voz.

Redes Neuronales

Las redes neuronales artificiales son modelos matemáticos inspirados en sistemas biológicos que son simulados en una computadora. En general una red neuronal está compuesta por un conjunto de nodos y un conjunto de ligas (Figura 20). Los nodos corresponden a las neuronas y las ligas representan las conexiones y el flujo de datos entre las mismas neuronas. A cada enlace está asociado un peso, que determina la naturaleza e intensidad de la influencia de un nodo sobre otro (Figura 21); así un peso positivo grande corresponde a una excitación fuerte y un peso negativo pequeño corresponde a una inhibición débil. Estos pesos son dinámicamente ajustados durante el entrenamiento.

La principal característica de las redes neuronales es su capacidad para aprender de su ambiente, y mejorar su desempeño a través del aprendizaje. Una red neuronal aprende acerca de su ambiente a través de un proceso interactivo de ajustes de sus pesos sinápticos y niveles de sesgo (bias).

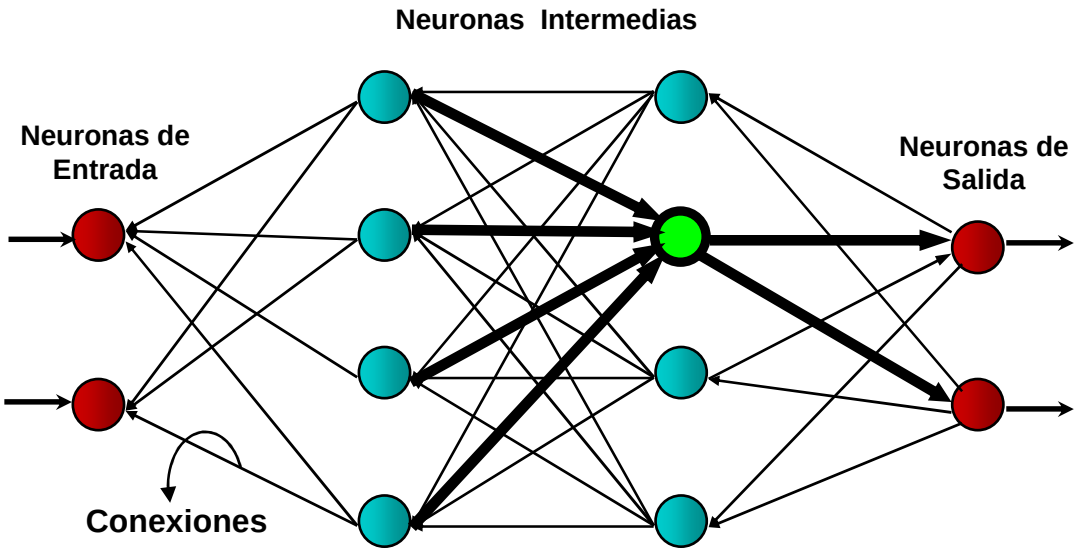


Figura 6. Arquitectura de una red neuronal. [Pava, 05]

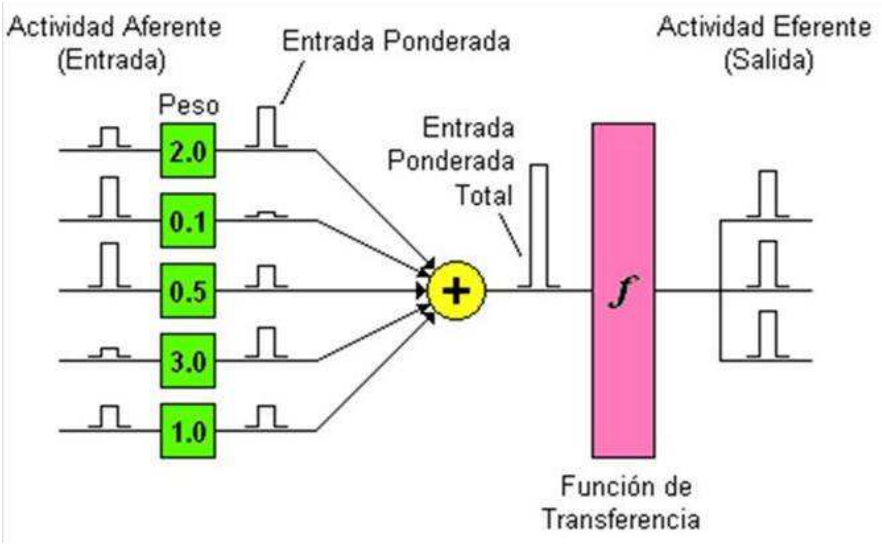


Figura 7. Proceso de una red neuronal. [Pava, 05]

El algoritmo de aprendizaje que utilizan las redes neuronales corresponde al conjunto de reglas bien definidas para la solución de un problema; existe una gran variedad de algoritmos de aprendizaje, los cuales difieren entre sí en la forma como se formulan los cambios en los pesos sinápticos. Entre los modelos más utilizados de redes neuronales se encuentran las redes con alimentación hacia delante (feed-forward), que utilizan alguna variación del método de entrenamiento de retropropagación (back-propagation).

Dentro del proceso de reconocimiento la red neuronal sirve de clasificador. Al igual que en los HMM la red neuronal toma como entradas a las características de la señal de voz, la cuales se pueden extraer con las técnicas LPC, MFCC, PLP, etc., vistas anteriormente. Una vez que se introducen a la red neuronal, el objetivo es clasificar cada unidad del habla de acuerdo con el conjunto de unidades especificadas (silabas, alófonos, etc.); se obtiene como salida una probabilidad por cada unidad fonética. Una vez que se procesa toda la señal de voz, resulta una matriz de probabilidades de tamaño C x F en donde C es el número de unidades que se quiere clasificar y F es el número de segmentos. Entonces se realiza una búsqueda para encontrar la secuencia con mayor probabilidad, para este proceso se utiliza el algoritmo de viterbi.

Actualmente la mayoría de los sistemas reconocedores de voz se basan en técnicas estadísticas porque son las que consumen menos memoria física y además el tiempo de respuesta se reduce considerablemente. No obstante los resultados que se obtienen son mejores en comparación con otras técnicas [Colás, 01].

3.1.2 Modelo del Lenguaje

El modelo del lenguaje, o gramática, permite a un reconocedor de voz asignar la probabilidad *a priori* a una secuencia de palabras, como se vio en el capítulo 1; está representado por *P(W)* en la formula (8). El propósito es predecir la palabra siguiente teniendo como referencia la palabra previa. Con esto se asume que el conocimiento del pasado es una buena guía de referencia para saber lo que sucederá en el futuro [Manning, 99]. En muchas tareas, la identificación de la palabra es difícil porque la entrada es muy ruidosa y ambigua; por ello es que conociendo la palabra previa puede dar un importante acercamiento a la palabra que le seguirá.

Existen muchos métodos para crear estas gramáticas, dentro de los que se encuentran el uso de sistemas basados en reglas (por ejemplo, gramáticas restringidas), y los métodos estadísticos, los cuales calculan un estimado de la probabilidad de las palabras de un gran conjunto de datos textuales; el modelo más utilizado es el N-grama.

Un modelo N-grama utiliza la palabra previa N – 1 para predecir la próxima palabra. Esta hipótesis dice que la probabilidad de una palabra depende sólo de la palabra previa, asunción de Markov. Ahora bien una cadena de Markov puede ser vista como un autómata de pesos de estado finito, en el que el próximo estado depende de una historia finita. Por ende, los modelos del lenguaje basados en N-gramas tienen la capacidad de explotar las propiedades estadísticas del lenguaje en contextos de dos, tres o más palabras.

Las gramáticas restringidas, por su parte, permiten representar las relaciones sintácticas que se establecen entre las palabras del lenguaje. Estas gramáticas están compuestas básicamente de dos partes: 1) por un conjunto de reglas que conforman su parte estructural y 2) por funciones de distribución de probabilidad asociadas a las reglas que

constituyen su parte estocástica. Existen dos problemas fundamentales en el uso de estas gramáticas; por un lado, su aprendizaje, es decir, la obtención de una gramática que represente un lenguaje, y por otro, su integración como modelo de interpretación en sistemas robustos.

3.1.3 Diccionario de Pronunciación

El propósito del diccionario de pronunciación es definir el rango de pronunciación de las palabras dentro del vocabulario. En éste se especifica la secuencia de sonidos (representados mediante un conjunto de símbolos) que componen una palabra. Los símbolos pueden ser definidos específicamente para la tarea de reconocimiento, o bien, obtenidos de un alfabeto fonético; en nuestro caso particular se utilizo el alfabeto fonético mexbet [Cuétara, 04].

La creación de un diccionario de pronunciación es necesaria por varias razones: 1) algunas palabras escritas ortográficamente igual pueden pronunciarse de manera diferente por cada persona; por ejemplo la palabra *competencia* puede ser pronunciada como: /k/ /o/ /m/ /p/ /e/ /t/ /e/ /n/ /s/ /a/ o como /k/ /o/ /m/ /p/ /e/ /t/ /e/ /n/ /s/ /i/ /a/; ambas pronunciaciones deben estar en el diccionario para el apropiado entrenamiento de los modelos acústicos y el apropiado reconocimiento de la palabra cuando sea pronunciada por diferentes personas; 2) la palabra tiene distintos significados, y distintas pronunciaciones de acuerdo al contexto en el que se usa; por ejemplo la palabra *numero*, puede referirse a un número, "El número 1", o al proceso de numerar "Él numeró las cajas". En nuestro caso particular estos problemas no se presentan puesto que gracias a que el corpus DIMEx100 fue etiquetado manualmente se tienen en todos los niveles – T22, T44 y T54 – diferentes pronunciaciones para una sola palabra (para mayor referencia ver el capítulo 2), y además en los niveles T44 y T54 se resuelve el problema (2); ya que se incluye en cada realización el acento correspondiente, evitando así la ambigüedad en la pronunciación.

Los diccionarios de pronunciación se construyen a partir de un corpus, que puede ser el mismo corpus usado durante el entrenamiento de los modelos acústicos o puede ser uno

diferente. Mientras más palabras se tengan en el diccionario y en el modelo de lenguaje, menos restrictivo se vuelve el reconocedor de voz.

Como se ha mencionado la etiquetación manual del corpus DIMEx100 permitió crear un diccionario basto en pronunciaciones, ya que no sólo contiene la forma canónica de las palabras, sino también sus posibles variantes. Un ejemplo del contenido de un diccionario de pronunciación creado con el corpus DIMEx100 en el nivel T44, se presenta a continuación:

A	a
A	a_7
A	e
A_7CIDOS	a_7 s i D o s
A_7GUILA	a_7 g_c g i l a
A_7LBUM	a_7 l b_c b u n
A_7MBITO	a_7 -N b_c b i_7 t_c t o
A_7MBITO	a_7 -N b_c b i t_c t o
A_7NGEL	a_7 -N x e l
A_7NGEL	a -N x e l

3.2 Factores que influyen en el reconocimiento de voz

A lo largo de este trabajo se han visto algunos elementos que influyen de manera importante en el proceso de reconocimiento de voz. La variabilidad de la señal de voz depende tanto de factores intrínsecos al fenómeno de producción de voz como a factores externos al mismo. Dentro de los factores intrínsecos destacan los siguientes:

1. **Variación sociolingüística**, debida fundamentalmente a los distintos acentos o formas de hablar de cada persona. Un ejemplo es la variación en el dialecto; por ejemplo, el español de México que se habla en el centro del país no es el mismo que se habla en el sur del país o en el norte, y dentro del centro del país, el español que

se habla en el centro del DF. (Tepito, Zona Rosa, etc.), difiere del que se habla en el sur del DF (Tlalpan, Coyoacan, etc.).

2. **Variabilidad en la producción de los sonidos**, debido a las distintas velocidades de producción, coarticulación, inclusión de ruidos (apertura y cierre de labios, respiración, sonidos de duda: p.e., *eh, uuh, mmm*), condiciones acústicas (hablar en ambientes ruidosos), contexto de la conversación, estado anímico, etc.

3. **Ambigüedad en la pronunciación de las palabras**, si aparecen problemas de claridad en la expresión o tenemos palabras similares, el porcentaje de error en el reconocimiento puede ser elevado. Un diccionario es propenso a sufrir graves complicaciones debido a que, implícitamente, hay palabras susceptibles de confusión. Por ejemplo, en un diccionario de más de 20,000 palabras, pueden haber palabras que se diferencien de otra solo por un alófono, por ejemplo las palabras *pasa* y *casa*.

4. **Tamaño del corpus,** con un diccionario de menos de 50 palabras el sistema funciona muy bien, pues la variedad de opciones es baja y la tasa de error será baja también. Si se trabaja con un diccionario más amplio pero una gramática sencilla, tampoco se presentan grandes complicaciones; sin embargo, con un diccionario muy amplio se presenta la ambigüedad y además la cantidad de datos de audio también debe ser considerablemente grande.

Entre los factores externos destacan:

1. **Variabilidad en la cadena de conversión y transmisión de la señal eléctrica**, debido a las diferencias entre las características de los micrófonos, líneas telefónicas, etc.

2. **Variabilidad en el ruido captado con la señal de voz**, debido a la existencia en las proximidades del micrófono de otras fuentes sonoras (TV, radio, carretera, impresoras, otras conversaciones, etc.).

A estos factores de variabilidad acústica habrá que añadir otros factores de variabilidad como son la utilización de palabras no contempladas en el vocabulario, la construcción de frases no permitidas por la gramática del lenguaje, la utilización de abreviaturas, los escenarios semánticos de las palabras, etc. Todo esto hace que el reconocimiento automático del habla no sea un problema tan trivial como a primera vista pueda parecer.

3.3 Herramientas

Hoy en día hay varias herramientas que se pueden obtener vía web de forma gratuita para la creación de reconocedores de voz; algunas de ellas ofrecen la creación de reconocedores de voz en español. Las aplicaciones más utilizadas para este propósito son:

- CSLU Toolkit[3]
- HTK (Hidden Markov Model Toolkit)[4]
- SPHINX

En este trabajo se decidió utilizar *SPHINX,* ya que es considerado uno de los mejores sistemas de reconocimiento de voz en el mundo hoy en día[5], además de que es una herramienta que tiene sus bases en el software libre. SPHINX fue diseñado en la universidad de Carnegie Mellon. Este sistema está basado en los HMM's, al igual que HTK toolkit y CSLU toolkit. SPHINX está compuesto por dos módulos, *SPHINX trainer*, para el entrenamiento de los modelos, y *SPHINX decoder* para el reconocimiento.

SPHINX trainer, está compuesto por un conjunto de programas, cada uno definido para cada tarea, y un conjunto de scripts, que organizan el orden en el cuál cada programa es

[3] http://www.cslu.ogi.edu/
[4] http://htk.eng.cam.ac.uk/
[5] http://cmusphinx.sourceforge.net/html/cmusphinx.php

ejecutado. SPHINX genera modelos acústicos discretos, semicontinuos o continuos (HMM con topología left to rigth) que pueden tener desde 1 hasta 24 gaussianas en cada estado.

SPHINX decoder, contiene algoritmos de programación dinámica como Baum-welch o Viterbi. La versión del *decoder* utilizada en este trabajo (s3.4) está limitada a modelos de lenguaje de bigramas o trigramas y sólo trabaja con modelos acústicos continuos de 3 o 5 estados.

Los ejecutables que *Sphinx decoder* contiene son:

1. **decode**: El decoder de Sphinx-3 s3.2/s3.3/s3.X para procesar archivos cepstrales.
2. **gausubvq**: Para construir subvectores de los clusters de los modelos acústicos.
3. **livedecode**: Ejecutable para las pruebas en vivo.
4. **livepretend**: Ejecutable para las pruebas en modo batch.
5. **align**: Ejecutable para realizar el alineamiento.
6. **allphone**: El reconocedor de fonemas de Sphinx-3.
7. **astar**: El generador del N-mejor en Sphinx-3.
8. **dag**: La aplicación para realizar la búsqueda del mejor patrón en Sphinx-3.

Capítulo 4

Modelo del lenguaje

El modelo del lenguaje permite a un reconocedor de voz asignar la probabilidad *a priori* a una secuencia de palabras. El propósito es predecir la palabra siguiente teniendo como referencia la palabra previa; esto ayuda mucho considerando que la elocución de entrada puede estar acompañada de mucho ruido. Por ello de acuerdo al modelo del canal ruidoso el modelo del lenguaje, *P(W),* representa la probabilidad absoluta de que W ocurra, siendo ésta independiente de cada evento de comunicación.

Existen diferentes problemas a los cuales se enfrenta un sistema ASR, en particular los relacionados con la utilización de palabras no contempladas en el vocabulario de la aplicación, la construcción de frases no permitidas por la gramática del lenguaje, la utilización de abreviaturas, los escenarios semánticos de las palabras, etc., conciernen al modelo del lenguaje. Por ello el modelo del lenguaje debe contemplar el vocabulario que será utilizado en el ambiente para el cual se construirá el reconocedor; pero no es fácil construir un modelo del lenguaje, por todas estas implicaciones y más, el siguiente capítulo está dedicado a todas estas cuestiones, porque para tener un buen reconocimiento de voz se necesita contar con un buen modelo del lenguaje, el cuál es el otro 50% de la efectividad del sistema.

4.1 Factores que influyen en la construcción de un Modelo del Lenguaje

En la sección 3.1.2, se habló de dos tipos de modelos del lenguaje: los estocásticos y los gramaticales. Dado que para este trabajo se utilizaron modelos estocásticos, esta sección se enfocará en los factores que influyen en la construcción de estos modelos.

Los modelos del lenguaje estadísticos son calculados generalmente a partir de grandes corpus de texto, delimitándolos por el tamaño del vocabulario, la longitud del contexto e

incluyendo esquemas para tratamiento de palabras desconocidas [Villaseñor *et al.*, 02]. Los textos pueden proceder de libros de consulta, documentos o informes y de la web, aunque sería ideal que fueran transcripciones del lenguaje hablado.

Uno de los factores determinantes de un modelo del lenguaje es el tamaño del corpus usado durante la fase de entrenamiento. Mientras más grande sea el corpus mayor será el número de contextos de uso de una palabra dada, y por ende, la calidad del modelo del lenguaje será mejor.

Dado que el modelo del lenguaje es un reflejo del corpus de entrenamiento, se deben tomar en cuenta varias cuestiones; si el corpus es suficientemente rico para la tarea especificada, si las palabras se encuentran bien definidas dentro del dominio, si el corpus es representativo de la lengua, etc., por ello se debe establecer en primera instancia su finalidad.

Lo que se busca es obtener un conjunto de textos, propios del dominio de la aplicación, que cubran el ámbito completo de los textos que se espera que produzca el sistema. Por ello un corpus necesita ser cuidadosamente diseñado, ya que si es demasiado específico, para la tarea o dominio, las probabilidades del modelo de ese corpus pueden ser demasiado limitadas lo que llevaría a tener contextos de los cuales no hay información, mientras que si es demasiado general o grande, las probabilidades no serían suficientes para reflejar la tarea o dominio lo que llevaría a hacer predicciones erróneas.

Si bien el modelo del lenguaje se construye a partir de un corpus de entrenamiento, ¿cuál sería la probabilidad de que el modelo genere dicho corpus?; este principio puede ser calculado a través de la entropía y la perplejidad.

La entropía puede verse como la probabilidad de que un modelo sea capaz de generar un cierto corpus; es decir, es una forma de medir la información que hay en una gramática en particular. Por ejemplo, para saber que tan bien una gramática puede generar un lenguaje dado o teniendo dos gramáticas y un corpus saber cual gramática genera mejor dicho corpus.

Para calcular la entropía se requiere tener establecida una variable random *X* que represente lo que se quiere predecir (palabras, letras, partes del habla, etc.), y que tiene una particular función de probabilidad, *p(x)*. La entropía de esta variable *X* es entonces,

$$H(X) = -\sum_{x \in X} p(x) \log_2 p(x) \qquad \textbf{(9)}$$

El resultado de la entropía se mide en bits. Para medir la entropía de un lenguaje se necesita considerar la longitud de las secuencias, es decir, de las oraciones.

Si el lenguaje se toma como un proceso estocástico *L* que produce una secuencia de palabras, su entropía de acuerdo al teorema de Shannon-McMillan-Breiman (Algoet y Cover, 1988; Cover y Thomas, 1991) se define como sigue

$$H(L) = -\lim_{n \to \infty} 1/n \log_2 p(w_1 w_2 \ldots w_n) \qquad \textbf{(10)}$$

Esto es, se puede tomar una sola secuencia suficientemente larga en vez de sumar todas las posibles secuencias, así, una secuencia de palabras suficientemente larga contendría muchas otras secuencias cortas, y cada una de estas secuencias cortas reocurriría en la secuencia larga acorde a sus probabilidades.

La perplejidad (formula 11) mide la incertidumbre de un evento. Al evaluar la perplejidad de un modelo del lenguaje, se calcula la probabilidad media que el modelo asigna a cada palabra del corpus de prueba. Esto es, la perplejidad de un modelo con respecto a una palabra *p1* representa el número de palabras que podrían seguir a *p1*. Así por definición, la perplejidad de un lenguaje depende del propio corpus de entrenamiento; por lo que una menor perplejidad indica que un lenguaje es más predecible, entonces mientras menor sea la perplejidad mayor será la calidad del modelo del lenguaje.

$$2^{-\sum_{i=1}^{N} \frac{1}{N} \log_2 q(x_i)} \qquad \text{ó} \qquad \textbf{(11)}$$

$$\mathbf{2^H} \qquad \textbf{(12)}$$

En la fórmula 12, la *H* representa a la entropía.

Un modelo del lenguaje es una distribución de probabilidad que se realiza sobre un corpus. En base a esto, por ejemplo, se puede encontrar que una frase *x* contenida en un corpus tiene una probabilidad media de 190. Así, este valor daría una perplejidad de 2^{190} por frase, lo cual es un valor muy grande. Sin embargo, se puede normalizar por la longitud de la frase y considerar sólo el número de bits por palabra. De esta manera, si las frases de un corpus constan de un total de 1,000 palabras, y éstas se codifican mediante 7,950 bits, entonces la perplejidad del modelo es $2^{7.95}$ = 247 *por palabra*. Es decir, que se tienen 247 posibilidades por cada palabra, siendo este valor más fácil de manejar.

4.2 Tipos de Modelos del lenguaje

El modelo de lenguaje captura el contexto de uso de un conjunto de palabras y con esta información se calcula la probabilidad de ocurrencia de una secuencia de dos o más términos.

El modelo más simple de una secuencia de palabras consiste en que cada palabra del lenguaje se sigue por cualquier otra palabra con igual probabilidad. Un mejor modelo consiste en asignar la probabilidad de distribución a cada palabra, por ejemplo, la palabra *de* ocurre 5,800 veces dentro del corpus DIMEx100, de 59,812 palabras que contiene el corpus en total; en contraste la palabra *iceberg* ocurre 1 vez en el corpus DIMEx100. De esta manera se pueden utilizar estas frecuencias relativas para asignar la probabilidad, de esta manera se le asignaría el 0.09% a la palabra *de* y el 0.0001% a la palabra *iceberg*. En este contexto además se observó que en el corpus DIMEx100 la palabra *iceberg* no seguía a la palabra *de*. Esto sugiere que se debería tomar en cuenta la probabilidad condicional de una palabra dada la palabra previa. Para obtener estas probabilidades, se requiere calcular todas las posibles palabras previas de cada palabra, para lo cual se requiere de un corpus muy grande. Para resolver este problema se calcula la probabilidad aproximada de una palabra dando todas las palabras previas; por ejemplo, calcular la probabilidad de una palabra dando solamente una sola palabra previa, a este cálculo se le conoce como bigrama; y al calcular la probabilidad de una palabra tomando como referencia dos

palabras previas, se le conoce como trigrama; de esta manera los N-gramas, ven N-1 palabras en el pasado.

La ecuación general para calcular la probabilidad condicional de los N-gramas es,

$$P(w_n \mid w_1^{n-1}) \approx P(w_n \mid w_{n-N+1}^{n-1}) \quad \textbf{(13)}$$

La fórmula 13 muestra que la probabilidad de una palabra w_n dando todas las palabras previas puede ser aproximada por la probabilidad dando solo las N palabras previas.

Un problema aunado a los N-gramas es que son muy dependientes del corpus con el que son entrenados; por lo tanto, dado que los corpus de entrenamiento son finitos es aceptable que no todas las combinaciones de palabras aparezcan y es seguro que se tenga un gran número de casos en donde la probabilidad sea cero. Para resolver este problema se reevalúan algunas de las probabilidades de los N-gramas con valores de cero o con una probabilidad muy pequeña y se les asignan valores diferentes de cero; esta técnica es llamada *smoothing o suavizado*.

Un método básico consiste en asignar una cierta probabilidad al espacio de sucesos no vistos, mediante la aplicación de la *Ley de Laplace, también llamada agregar uno (add-one)*, que consiste en incrementar la frecuencia de todos los sucesos en una unidad [Jurafsky&Martin, 00]. Por ejemplo esta técnica aplicada a un unigrama, la máxima probabilidad de la probabilidad del unigrama puede ser calculada dividiendo el número de veces que aparece la palabra *c,* entre el número de tokens[1] N,

$$P(w_x) = \frac{c(w_x)}{\sum_i c(w_i)} = \frac{c(w_x)}{N} \quad \textbf{(14)}$$

Como el estimado recae en *c*, el ajuste para agregar uno puede ser definido agregando uno a *c* y multiplicarlo por el factor de normalización, *N/(N+V)*, en donde *V* representa el número total de tipos[2] en el lenguaje; se agrega uno al número de veces que aparece la

[1] Un token se refiere al número total de palabras en el corpus.
[2] Un tipo (type) se refiere al número total de palabras distintas dentro del corpus, el cuál hace referencia al tamaño del vocabulario.

palabra de tipo *c*, el número total de tokens debe ser incrementado por el número de tipos, rescribiendo la fórmula se tiene,

$$c_i^* = (c_i + 1)\frac{N}{N+V} \quad \textbf{(15)}$$

La estimación que proporciona el algoritmo agregar-uno es muy dependiente del tamaño del vocabulario y reserva una probabilidad demasiado elevada para los sucesos no vistos, es decir, sobreestima los sucesos no vistos; es por ello que este algoritmo no es comúnmente usado.

Como alternativa, existen otros algoritmos de suavizado que agregan un poco de la masa de probabilidad a los N-gramas con probabilidad cero, entre los más comunes incluyen *backoff* o *deleted interpolation*, ya sea con Witten-Bell discounting o Good-Turing discounting [Jurafsky&Martin, 00].

Witten-Bell Discounting

En este algoritmo, primero se considera una palabra con frecuencia cero, o un N-grama que aún no ha sucedido. Cuando este N-grama sucede por primera vez, esto se ve como un nuevo N-grama. Así la probabilidad de ver un N-grama con frecuencia cero puede ser modelado por la probabilidad de que ese N-grama haya sido visto al menos una vez.

El cálculo de la probabilidad de ver un N-grama por primera vez se realiza contando las veces que vemos un N-grama por primera vez en un corpus de entrenamiento. Esto es muy simple de producir debido a que la cuenta de los N-gramas que han sucedido al menos una vez es el número de tipos de N-gramas en los datos de entrenamiento.

Good-Turing Discounting

Este algoritmo fue descrito por primera vez por Good en 1953, quién dio créditos a Turing por la idea original. En su simple formulación se pretende re-estimar la cantidad de la

masa de probabilidad asignada a los N-gramas con cero o con probabilidades pequeñas, para ver el número de N-gramas con altas probabilidades.

Deleted Interpolation

La técnica elimina sucesivamente cada trigrama del corpus de entrenamiento y estima los mejores valores de λ (interpolación) para el resto de N-gramas en el corpus. El algoritmo utiliza las frecuencias de los distintos N-gramas y les resta 1 para tener en cuenta los sucesos no vistos, evitando así la sobreestimación del modelo. El algoritmo aumenta el valor de λ_i (se refiere a cada uno de los pesos de la interpolación) dependiendo de la frecuencia de aparición del i-grama respecto a los N-gramas correspondientes de distinto orden.

Back-off

Esta técnica consiste en escoger el modelo más apropiado para el contexto actual, es decir, para cada suceso, sólo un factor de ponderación puede ser distinto de cero.

Los modelos de *Back-off* de N-gramas fueron propuestos por Katz (1987). Estos estiman la probabilidad de un N-grama a partir de los sucesos de orden menor que *N*, según se expresa en la ecuación (16) [Jurafsky&Martin, 00],

$$\hat{P}(w_i|w_{i-2}w_{i-1}) = \begin{cases} P(w_i|w_{i-2}w_{i-1}), & \text{if } C(w_{i-2}w_{i-1}w_i) > 0 \\ \alpha_1 P(w_i|w_{i-1}), & \text{if } C(w_{i-2}w_{i-1}w_i) = 0 \\ & \text{and } C(w_{i-1}w_i) > 0 \\ \alpha_2 P(w_i), & \text{otherwise.} \end{cases} \quad \textbf{(16)}$$

Para los sucesos de baja frecuencia el valor de su probabilidad se calcula aplicando un descuento α, que resta una cierta masa de probabilidad a los sucesos vistos para repartirla entre los no vistos. Para calcular esta función de descuento se utilizan los algoritmos de Good-Turing y Witten-Bell Discounting.

Por otro lado, los modelos basados en gramáticas, se utilizan cuando el objetivo es modelar restricciones y características lingüísticas de orden superior, y además, conseguir que la salida del sistema no sólo sea una secuencia de palabras o categorías sino que además incluya la estructura sintáctica asociada a la misma; para este caso se utilizan otro tipo de gramáticas, como las de contexto-libre, asociando mediante alguno de los métodos conocidos, probabilidades a las distintas reglas. Aunque estas gramáticas son más potentes que los N-gramas, son difíciles de inferir automáticamente y la asociación de probabilidades a las reglas necesita de un proceso posterior a la generación del conjunto de reglas, que suele ser manual.

Actualmente, se están incorporando mecanismos de unificación de rasgos junto con las reglas de estas gramáticas de contexto-libre probabilísticas (PCFG o Probabilistic Context-Free Grammar), dando una potencia superior al modelo gramatical pero aumentando notablemente el tamaño del espacio de búsqueda. Además, es necesario estimar las probabilidades a asociar a las reglas de la gramática, lo que suele tener un costo mayor.

Otro punto importante es que son complicados de integrar en los sistemas de reconocimiento de forma que guíen el proceso acústico. Por todo ello, todavía no ha habido demasiados esfuerzos por integrarlos en sistemas de reconocimiento automático de voz. En contra parte los modelos del lenguaje estadísticos además de ser más flexibles permiten capturar situaciones parecidas al lenguaje hablado, en donde las reglas del lenguaje escrito no siempre son respetadas.

4.3 Metodologías para enriquecer un modelo del lenguaje

El propósito del modelo del lenguaje es reducir el espacio de búsqueda y acelerar el proceso de reconocimiento, por ello se han propuesto varias técnicas para enriquecer el modelo, y por ende el reconocimiento.

El enriquecimiento del modelo del lenguaje se puede hacer desde la recolección de los datos, ya que como se ha visto, el modelo del lenguaje es un reflejo del corpus de entrenamiento.
Para enriquecer el corpus de entrenamiento, primero se necesita identificar las palabras críticas, es decir, las palabras que se encuentran mal representadas en el corpus, o con muy poca ocurrencia dentro del mismo.

Para identificar las palabras críticas se necesita hacer un análisis del corpus, esto es, encontrar la frecuencia de ocurrencia de cada uno de los tipos ya que con ello se asigna la probabilidad de distribución. Una técnica para realizar este análisis se describe a continuación [Villaseñor *et al.*, 03]:

1) Se crea un índice del corpus. Dicho índice indica los términos usados en el corpus y sus frecuencias de ocurrencia. Su representación es mediante un archivo invertido (Kowalski, 97), es decir, una estructura de datos que consta de un diccionario y una lista invertida instrumentados a través de tablas de hash. En el diccionario se almacenan todos los términos extraídos, junto con su frecuencia total de ocurrencia. En la lista invertida se almacena, para cada término, una lista dinámica de las colecciones (podrían ser más de dos) en los que el término fue encontrado y la frecuencia de ocurrencia en cada una de ellas.

2) A partir del índice construido, una frecuencia f_k es asignada a cada uno de los términos. Esta frecuencia indica el número de ocurrencias del término *k* en el corpus. Con base en estas frecuencias se construye una distribución de probabilidad de los términos en el corpus, $\mathbf{D=\{p_k\}}$, en donde, $\mathbf{p_k = f_k \,/\, \Sigma_{j=1} f_j}$, expresa la probabilidad de ocurrencia del término *k* en el corpus, y *n* indica el número de términos existentes en el índice. Así para identificar un término crítico se calcula el promedio como se puede ver en la siguiente formula.

$$d_\mu = \frac{1}{n}\sum_{k=1}^{n} p_k$$

3) Para obtener los ejemplos de las palabras críticas para el enriquecimiento del corpus, se pueden considerar dos fuentes, un corpus de referencia, es decir, un corpus con diálogos reales, o extraer nuevos documentos web.

4) Para enriquecer el corpus de entrenamiento, 1) se construye un nuevo corpus con las frases que contienen las palabras críticas, las cuales se obtuvieron del corpus de referencia o de los nuevos documentos web; 2) se calcula el déficit de ocurrencia de cada palabra crítica. Este déficit indica el número de veces que la palabra $t \in W_c$[3] debe ser incorporada en el corpus de entrenamiento.

$$deficit_t = (P_t^{C_r} - P_t^{C_e}) \times |C_e|$$

En donde, C_r es el corpus de referencia o documentos web y C_e es el corpus de entrenamiento.

5) Se determina cuantas frases se agregaran del nuevo corpus creado con las palabras críticas al corpus de entrenamiento. El número de frases $\hat{r}$ es calculado en función del déficit de ocurrencia de todas las palabras críticas.

$$\hat{r} = max(R),$$

En donde,

$$R = \{r_t | t \in W_c\}$$

$$r_t = \frac{deficit_t}{f_t^{C_s}}$$

Finalmente se construye el corpus de entrenamiento enriquecido C_{e+}. Este paso consiste en agregar *r* veces las frases seleccionadas al corpus de entrenamiento.

Otra propuesta para enriquecer el modelo del lenguaje es incorporando conocimientos lingüísticos. Uno de los principales problemas a los que se enfrenta esta propuesta es el tamaño de los corpora (normalmente textos o transcripciones de conversaciones reales). Normalmente, los corpora son escasos y pequeños, y no suelen representar exhaustivamente los distintos elementos lingüísticos que aparecen en la realidad. Esto

[3] W_c es el conjunto de palabras críticas.

conduce a problemas de cobertura de la gramática, que recaerán en el sistema de reconocimiento de voz.

Sin embargo, se ha demostrado que para la mayoría de las aplicaciones [Colás, 01], no es necesario haber reconocido todos los elementos de la frase de forma correcta, sino reconocer aquellos elementos que aportan información semántica relevante para el entendimiento de la frase. Ello conduce al planteamiento de incorporar probabilidades en la gramática y permitir que el sistema de reconocimiento pueda evolucionar reconociendo sólo segmentos de la frase, entregando diferentes alternativas y realizando un proceso posterior de dicha información.

4.4 Modelo del lenguaje para golem

Dentro del proyecto DIMEx100, uno de los objetivos que se planteo para esta tesis fue la creación de un modelo del lenguaje para un robot desarrollado dentro del proyecto *Golem*[4].

El proyecto *Golem*, por su parte, tiene por objetivo estudiar la integración de información multimodal, principalmente lingüística y de lenguaje natural en español de México hablado, para la navegación de un robot en un espacio limitado; la primera fase del proyecto se enfocó a que el robot diera una visita guiada del Departamento de Ciencias de la Computación del IIMAS, UNAM.

La interacción del robot (agente) con el mundo corresponde con la interpretación de los modelos de diálogo que conforman el dominio conversacional. Los modelos de diálogo se especifican como gráficas de transición recursiva. En cada situación conversacional, el agente puede percibir y realizar sólo un conjunto reducido de intenciones significativas y acciones relevantes. El agente realiza acciones como resultado de las intenciones que percibe de otros agentes. Tanto las intenciones que interpreta el agente como las acciones que realiza son independientes de la modalidad en la que se expresan. El esquema permite

[4] http://leibniz.iimas.unam.mx/~luis/golem/

la definición de situaciones recursivas cuya interpretación equivale a interpretar un modelo de diálogo completo.

El sistema está implementado como un conjunto de agentes computacionales, cada uno de los cuales corresponde a una modalidad de información, ya sea de entrada o de salida. Para la implementación, se seleccionó la arquitectura *Open Agent Architecture* (OAA), la cual se muestra en la Figura 23 y se describe como sigue:

1. Especificación e interpretación de modelos de diálogo.
2. Interpretación del habla.
3. Movimiento de Golem.
4. Síntesis de voz.
5. Despliegue de imágenes y videos.

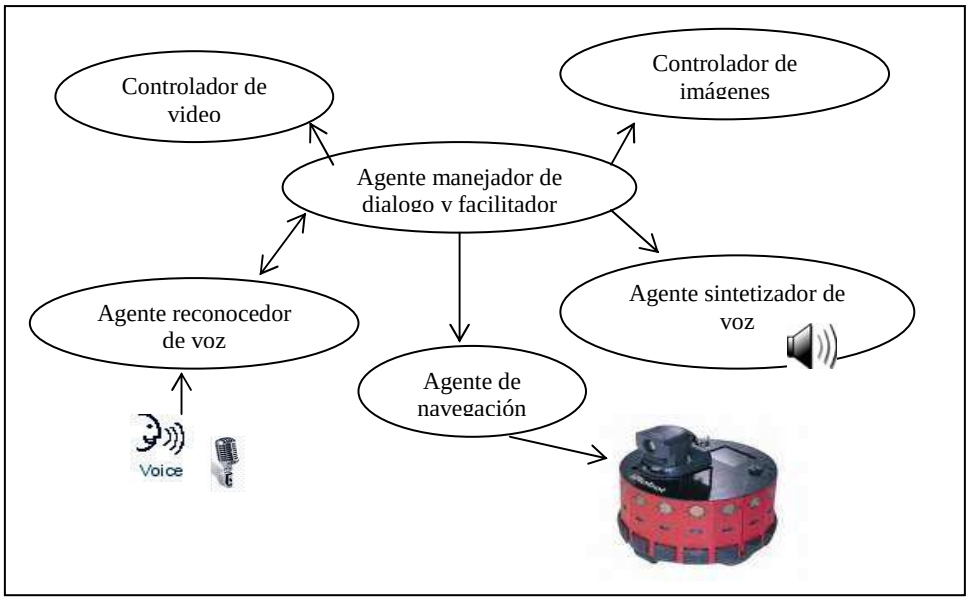


Figura 1. Estructura de agentes de Golem.

Como se puede ver uno de los componentes importantes dentro del proyecto *Golem* es el reconocedor de voz, ya que por medio de él, el usuario interactúa con el robot.

En una primera fase se utilizó el reconocedor de voz creado por Patricia Pavón[5] con el 35% del corpus DIMEx100 en el nivel T22, el cual no tiene un enfoque específico, es decir, es un

[5] http://leibniz.iimas.unam.mx/~luis/DIME/

reconocedor de voz de uso general, lo que significa que las probabilidades no podrían ser suficientes para reflejar la tarea o dominio. Se planteó entonces crear un reconocedor de voz que pudiera reflejar de mejor forma el dominio en el cuál se desenvolverá el robot golem. Dentro de este contexto se propuso crear un modelo del lenguaje con frases específicamente enfocadas a la tarea, y entrenar los modelos acústicos con el 50% del corpus DIMEx100 hasta el momento etiquetado.

El primer paso para la elaboración del modelo del lenguaje es la recolección del corpus rico en frases que pudieran ser utilizadas dentro del contexto en el cuál será utilizado el robot golem.

La construcción de un corpus no es una tarea fácil, por muchos factores como anteriormente se ha explicado, una razón es porque los textos escritos no representan adecuadamente muchos fenómenos del habla espontánea. Una manera de disminuir este problema es usando documentos web como fuente para obtener los datos; esto es porque la mayoría de las personas que contribuyen a crear los documentos web, utilizan un lenguaje informal e incluyen expresiones no gramaticales, similares a las utilizadas en el lenguaje oral espontáneo. Esta situación no solo permite la construcción de corpora muy grandes sino también la creación de corpora con una buena combinación de texto bien escrito gramaticalmente y texto cercano al lenguaje hablado comúnmente.

Por esta razón se decidió utilizar la web como fuente de información para crear el corpus utilizado en la construcción del modelo del lenguaje para el robot golem. En una primera fase se construyeron dos corpus; éstos fueron recolectados con ayuda de una herramienta proporcionada por el INAOE[6], la cual trabaja con una gran cantidad de texto recolectado de Internet.

Para recolectar los corpora se le pasaba como parámetro a la herramienta el número de palabras clave[7] que tenía que contener como mínimo la frase, así el primer corpus

[6] Instituto Nacional de Astrofísica Óptica y Electrónica, Puebla. http://www.inaoep.mx/

[7] Una palabra clave es considerada como una palabra que se utiliza con mucha frecuencia dentro del contexto en el cual se desenvuelve golem.

recolectado contaba con 1,698,647 de frases en donde cada frase contenía mínimo tres palabras clave, este corpus contenía frases no muy largas (50 palabras como máximo), el segundo corpus contaba con 760,424 este corpus contenía frases largas (más de 50 palabras), y contenía cinco palabras clave como mínimo en cada frase. Al crear el modelo del lenguaje para cada corpus y realizar las pruebas correspondientes se llego a la conclusión de que los corpus eran demasiado grandes ya que los resultados no eran lo suficientemente buenos al no generar las hipótesis deseadas. Esto se debió a que la mayoría de las frases contenidas en el corpus no eran el reflejo de las frases utilizadas dentro del contexto en el cual se desenvolvía golem.

Se decidió reducir el tamaño de los corpora y crear nuevos corpora que contuvieran frases más relacionadas con la tarea, para ello se utilizó como base el primer corpus, el cuál contenía frases cortas, además de que en las pruebas fue el corpus que arrojo mejores resultados. A partir de este corpus se crearon 3 corpora, las reglas utilizadas para crear cada uno de ellos son las siguientes:

Primer corpus. Si la frase contenía hasta 10 palabras tenía que contener mínimo 5 palabras clave, si contenía hasta 20 tenía que contar con 10 palabras como mínimo, etc.

Segundo corpus. Si la frase contenía hasta 6 palabras tenía que contener mínimo 3 palabras clave, si contenía hasta 12 tenía que contar con 6 palabras como mínimo.

Tercer corpus. Si la frase contenía hasta 6 palabras tenía que contener mínimo 3 palabras clave, si contenía hasta 12 tenía que contar con 6 palabras como mínimo.

Además se enriqueció cada corpus con frases obtenidas de textos que se utilizaron para explicar cada uno de los proyectos que se desarrollan dentro del departamento. Para cada corpus recolectado se creó un modelo del lenguaje, el cual se probó con el 50% de los datos que se utilizaron para el entrenamiento del reconocedor de voz con el nivel de segmentación T44. Los resultados obtenidos se presentan en el capítulo 5.

Capítulo 5

Desarrollo, experimentos y resultados

5.1 El reconocedor de voz

Para la construcción del reconocedor de voz se utilizó el nivel de etiquetación T44 del corpus DIMEx100. Hasta la elaboración de este trabajo se contaba con el 59% del corpus etiquetado, del cual el 50% del corpus – que corresponde a 50 carpetas - se utilizó para la etapa de entrenamiento de los modelos acústicos y el 9% restante – que corresponde a 9 carpetas - se utilizó para la etapa de reconocimiento (pruebas).

Se decidió utilizar el software SPHINX, que está compuesto por *SPHINX trainner y SPHINX decoder* - para este último se utilizó la versión 3.4 - debido a su disponibilidad para la plataforma linux en la cual se trabajo y su facilidad de uso. Además porque en la primera fase del proyecto DIMEx100 se creó un reconocedor de voz en el nivel de etiquetación T22[1] con el software SPHINX y con el que posteriormente se comparó el nuevo reconocedor construido en el nivel T44.

Debido al aumento en la cantidad de datos etiquetados y revisados del corpus DIMEx100 se volvió a entrenar el reconocedor en el nivel T22 para tener un mismo número de datos entrenados en los dos reconocedores.

En primera instancia se creó el diccionario de pronunciación (apéndice 3), el cuál gracias a la etiquetación manual del corpus DIMEx100, cuenta con varias pronunciaciones para cada palabra, y por lo tanto una mayor diversidad acústica y una mayor probabilidad de reducir la tasa de error.

[1] Para mayor referencia ver el trabajo realizado por Patricia Pérez titulado "Construcción de un reconocedor de voz utilizando Sphinx y el corpus DIMEx100", 2006.
http://leibniz.iimas.unam.mx/~luis/DIME

Antes de utilizar los archivos de audio, estos se modificaron de la siguiente manera:

- Los audios fueron grabados a 44.1 KHz originalmente, pero para utilizarlos con SPHINX se convirtieron a 16 KHz.
- Debido a que los audios fueron grabados de manera controlada, es decir, con el menor ruido posible, se les tuvo que aplicar *dither*[2] a las señales de audio. Esto se debió a que se tenía una onda limpia y al reducir el número de bits en la señal, por la conversión, provoco que la curva se viera escalonada. Por ello la señal perdió resolución y añadir un poco de ruido hizo que su forma escalonada se suavizará creando un sonido más natural.

Después de modificar los archivos de audio se realizó la creación de los modelos acústicos (apéndice 1), éste es el proceso más largo en la construcción de un reconocedor de voz, ya que dependiendo de la cantidad de datos que se utilicen para el entrenamiento, este puede requerir de mucho tiempo.

Las características de los modelos acústicos creados con la herramienta *Sphinxtrain*, se presentan a continuación:

- Modelo Oculto de Markov de 3 estados, continuo.
- 8 gaussianas por estado de cada modelo.
- Trifonemas[3].

Sphinxtrainer, permite crear HMM Markov continuos y semicontinuos. Aunque con los HMM semicontinuos – todos los modelos comparten un conjunto fijo de distribuciones de probabilidad – se reduce en cierta medida el coste computacional debido a que el número de distribuciones de probabilidad no depende del número de modelos, actualmente los HMM continuos producen el mejor rendimiento si cuentan con una cantidad suficiente de datos de entrenamiento pero con un aumento considerable en el cómputo, esto debido a que en los HHM continuos cada estado de cada modelo tiene sus propias distribuciones de probabilidad que modelan las características acústicas de la voz. Sin embargo, se decidió

[2] Aplicar *dither* a una señal significa mezclarle "ruido" de manera controlada.
[3] Trifonemas significa que son fonemas dependientes del contexto, es decir, cada fonema depende de los fonemas vecinos.

utilizar los HHM continuos porque se consideró que se contaba con la cantidad de datos de entrenamiento suficientes y con máquinas con un buen nivel de procesamiento. El equipo utilizado cuenta con las siguientes características: procesador Intel Pentium 4 a 3GHz, 1GB de memoria en RAM, tarjeta de audio AC'97 integrada en una placa Intel 915G/P/GV.

En total se crearon 44 modelos acústicos correspondientes a las unidades alofónicas del nivel de etiquetación T44 del corpus DIMEx100 (ver tablas 10, 11 y 12 del capítulo 2).

Después de construir los modelos acústicos, con la transcripción de los enunciados se creó el modelo del lenguaje de 3-gramas (apéndice 2) y el diccionario de pronunciación. El modelo del lenguaje junto con el diccionario de pronunciación y los modelos acústicos son los elementos principales para la construcción del reconocedor de voz.

5.1.1 Experimentos

Como se ha mencionado a lo largo de este trabajo el proceso de reconocimiento o decodificación consiste en que el sistema identifique una pronunciación dada como alguna de las que ya conoce, este aprendizaje se hizo en la etapa de entrenamiento. Para ello SPHINX *decoder* al final del reconocimiento arroja un archivo que describe todo el proceso; desde que entra la señal de audio, hasta que se genera la frase completa, un ejemplo del contenido de este archivo – llamado *log* – se muestra a continuación:

```
INFO: main_live_pretend.c(93): PARTIAL HYP: <sil>
INFO: main_live_pretend.c(93): PARTIAL HYP: <sil> ES
INFO: main_live_pretend.c(93): PARTIAL HYP: <sil> EN LA
INFO: main_live_pretend.c(93): PARTIAL HYP: <sil> AME27RICA
INFO: main_live_pretend.c(93): PARTIAL HYP: <sil> EN EL CASO
INFO: main_live_pretend.c(93): PARTIAL HYP: <sil> EN EL CASO DE
INFO: main_live_pretend.c(93): PARTIAL HYP: <sil> EN EL CASO DE
INFO: main_live_pretend.c(93): PARTIAL HYP: <sil> EN EL CASO DE LAS QUE27
INFO: main_live_pretend.c(93): PARTIAL HYP: <sil> EN EL CASO DE LA ESCUELA
INFO: main_live_pretend.c(93): PARTIAL HYP: <sil> EN EL CASO DE LA PSICOLOGI27A
INFO: main_live_pretend.c(93): PARTIAL HYP: <sil> EN EL CASO DE LA PSICOLOGI27A
INFO: main_live_pretend.c(93): PARTIAL HYP: <sil> EN EL CASO DE LA PSICOLOGI27A <sil>

Backtrace(s00101)
 LatID    SFrm     EFrm     AScr       LScr      Type
   11       0        5      -28119     -74100    -1        <sil>
  341       6       18      -224247    -115321   0         EN(4)
  934      19       29      -150036     -74575   0         EL
 2280      30       59      -541405    -131043   0         CASO(2)
 2586      60       66      -68884     -41734    0         DE
 2776      67       73      -104679    -103104   0         LA(7)
 5255      74      137      -838249    -218263   0         PSICOLOGI27A
 6013     138      154      -351639     -74100   -1        <sil>
 6078     155      155       0         -23123    0         </s>
            0      155      -2307258   -855363   (Total)

FWDVIT: EN EL CASO DE LA PSICOLOGI27A (s00101)
```

Figura 1. Vista del contenido del archivo de decodificación.

Como se ve en la Figura 23, se va generando una hipótesis parcial, que después da lugar a la frase reconocida.

En esta etapa se realizaron 3 experimentos, los cuales se hicieron en modo *batch*, es decir, se utilizó audio pregrabado.

Experimento 1

El primer experimento consistió en utilizar las 50 carpetas utilizadas durante la etapa de entrenamiento, las cuales equivalen a 2,498 oraciones; con estas oraciones se construyo el modelo del lenguaje y el diccionario de pronunciación. El diccionario de pronunciación cuenta con 8,449 palabras en total (tokens), de las cuales 5,466 palabras son diferentes (tipos). Este primer experimento se realizo para ver la calidad de los modelos acústicos.

Experimento 2

El segundo experimento consistió en utilizar las 9 carpetas restantes (no se utilizaron en el entrenamiento), las cuales equivalen a 450 oraciones. En este experimento se utilizó el modelo del lenguaje del primer experimento. El diccionario de pronunciación además de contar con 8,449 palabras se le agregó las palabras que no se encontraban en él pero que eran utilizadas en alguna oración de las 9 carpetas utilizadas para este experimento; aumentado así a 9,749 palabras en total (tokens), de las cuales 6,357 son diferentes (tipos).

Experimento 3

El objetivo de este experimento era hacer una prueba simulando condiciones normales. En este experimento se le pidió a 8 personas – 1 de ellas grabó en los audios originales del corpus - que leyeran 15 oraciones cada una, estas frases fueron escogidas al azar de las 9 carpetas para pruebas. Las oraciones fueron grabadas con el software *Visual Sound Recorder*; se decidió grabar las frases porque posteriormente se compararían los resultados obtenidos con los del reconocedor en el nivel T22, y para ello se necesitaban utilizar los mismos audios. El diccionario de pronunciación y el modelo del lenguaje fueron los mismos que se utilizaron en el segundo experimento.

Los mismos experimentos se repitieron con el reconocedor en el nivel T22.

5.1.2 Resultados

El software utilizado para medir los porcentajes de error fue *sclite*[4], esta herramienta compara las hipótesis generadas por el reconocedor de voz con el texto de referencia, es decir, con los textos originales. Después de la comparación genera varios reportes en donde se muestra la evaluación a nivel de palabra y a nivel de elocución.

[4] ftp://jaguar.ncsl.nist.gov/current_docs/sctk/doc/sclite.htm#sclite_name_0

```
                DUMP OF SYSTEM ALIGNMENT STRUCTURE

System name:   hipotesis_nuevo_carpetas.txt

Speakers:
    0:

Speaker sentences   0:     #utts: 450
id: (s06301)
Scores: (#C #S #D #I) 7 2 1 1
REF:  el ** LOGRO de la excelencia acade27mica Y libertad de CA27TEDRA
HYP:  el LA HORA  de la excelencia acade27mica * libertad de CATEDRAL
Eval:    I  S                                  D             S

id: (s06302)
Scores: (#C #S #D #I) 5 4 0 1
REF:  precio DEL PETR027LEO SUFRE importante ** ALZA a nivel mundial
HYP:  precio DE  PATRONES   SUS   importante DE LOS  a nivel mundial
Eval:        S   S          S                I  S

id: (s06303)
Scores: (#C #S #D #I) 2 8 0 3
REF:  **** * FLORIDA EL  ESTADO DONDE MA27S   INOCENTES son **** CONDENADOS A  muerte
HYP:  FLOR Y DE      LAS TODOS  LOS   DEMA27S INOCENTE  son COMO DE         LA muerte
Eval: I    I S       S   S      S     S       S             I    S          S
```

Figura 2. Muestra de la alineación de los textos con sclite.

La forma en que *sclite* trabaja es como sigue, primero hace una alineación del texto que contiene las hipótesis con el texto de referencia como se ve en la Figura 24. En esta figura se puede observar cada par de oraciones, es decir, el texto de referencia con su respectiva hipótesis, en donde REF es el texto de referencia y HYP son las hipótesis; además muestra el número de palabras correctas (#C), sustituidas (#S), eliminadas (#D) e insertadas (#I).
Una vez que se alinean los textos y se cuentan por cada oración las palabras insertadas, eliminadas o sustituidas, se calculan los porcentajes de error por cada par de archivos; las formulas que utiliza *sclite* para obtener estos porcentajes se muestran a continuación:

$$\text{Percent of correct words} = \left[\frac{\#\text{ Correct words}}{\#\text{ Reference words}} \right] * 100$$

$$\text{Percent of substituted words} = \left[\frac{\#\text{ Substituted words}}{\#\text{ Reference words}} \right] * 100$$

$$\text{Percent of inserted words} = \left[\frac{\#\text{ Inserted words}}{\#\text{ Reference words}} \right] * 100$$

$$\text{Percent of deleted words} = \left[\frac{\text{\# Deleted words}}{\text{\# Reference words}}\right] * 100$$

Un ejemplo del resultado del cálculo de estos porcentajes se muestra en la siguiente Figura:

SYSTEM SUMMARY PERCENTAGES by SPEAKER

hipotesis_nuevo_carpetas.txt								
SPKR	# Snt	# Wrd	Corr	Sub	Del	Ins	Err	S.Err
	450	4752	59.8	37.0	3.1	18.7	58.8	95.1
Sum/Avg	450	4752	59.8	37.0	3.1	18.7	58.8	95.1
Mean	450.0	4752.0	59.8	37.0	3.1	18.7	58.8	95.1
S.D.	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Median	450.0	4752.0	59.8	37.0	3.1	18.7	58.8	95.1

Figura 3. Porcentajes arrojados por sclite.

En esta Figura se puede observar que la suma del porcentaje de palabras correctas (columna Corr) y palabras erróneas (columna Err) suman más del 100%, esto es porque todo se calcula por separado, es decir, en el cálculo de las palabras erróneas se suman todas las palabras insertadas, eliminadas o sustituidas. Otra cuestión importante es que estos porcentajes son calculados a nivel de palabra, el porcentaje de error de la oración completa se puede observar en la última columna de la Figura 25 (S.Err); la fórmula para obtener este porcentaje se muestra a continuación:

$$\text{Percent of sentence errors} = \left[\frac{\text{\# incorrect ref and hyp pairs}}{\text{\# ref and hyp pairs}}\right] * 100$$

Cabe destacar que si una oración tiene al menos una palabra mal, es decir, contiene palabras insertadas, sustituidas o eliminadas, se considera errónea toda la oración.

Como he venido mencionando los resultados obtenidos con el reconocedor construido en el nivel T44, fueron comparados con los obtenidos con el reconocedor en el nivel T22, por ello en cada evaluación de cada experimento mostraré las tablas comparativas.

Evaluación del Experimento 1

Después de la comparación del archivo con las hipótesis y el archivo de referencia, se obtuvieron los siguientes resultados:

```
                DUMP OF SYSTEM ALIGNMENT STRUCTURE

System name:   resultados/hipotesis_nuevo_todo.txt

Speakers:
    0:

Speaker sentences   0:     #utts: 2498
id: (s00101)
Scores: (#C #S #D #I) 6 0 0 0
REF:  en el caso de la psicologi27a
HYP:  en el caso de la psicologi27a
Eval:

id: (s00102)
Scores: (#C #S #D #I) 8 0 0 0
REF:  de la ciudad de me27xico para el mundo
HYP:  de la ciudad de me27xico para el mundo
Eval:

id: (s00103)
Scores: (#C #S #D #I) 11 0 0 0
REF:  y sin embargo no deja de ser una cuestio27n muy importante
HYP:  y sin embargo no deja de ser una cuestio27n muy importante
Eval:

id: (s00104)
Scores: (#C #S #D #I) 6 0 0 0
REF:  el derecho de la unio27n europea
HYP:  el derecho de la unio27n europea
Eval:

id: (s00105)
Scores: (#C #S #D #I) 8 0 0 0
REF:  mantenimiento de alfombras en la ciudad de me27xico
HYP:  mantenimiento de alfombras en la ciudad de me27xico
Eval:

id: (s00106)
Scores: (#C #S #D #I) 9 0 0 0
REF:  certificados de idiomas en caso de que los posea
HYP:  certificados de idiomas en caso de que los posea
Eval:

id: (s00107)
Scores: (#C #S #D #I) 6 0 0 0
REF:  eso es lo que se refleja
HYP:  eso es lo que se refleja
Eval:

id: (s00108)
Scores: (#C #S #D #I) 8 1 0 0
REF:  EN la mayori27a de los casos se emplea metafo27ricamente
HYP:  ES la mayori27a de los casos se emplea metafo27ricamente
Eval: S

id: (s00109)
Scores: (#C #S #D #I) 9 0 0 0
REF:  fondo de las naciones unidas para la infancia unicef
HYP:  fondo de las naciones unidas para la infancia unicef
Eval:

id: (s00110)
Scores: (#C #S #D #I) 8 0 0 0
REF:  instituto nacional de estadi27stica geografi27a e informa27tica inegi
HYP:  instituto nacional de estadi27stica geografi27a e informa27tica inegi
Eval:
```

Solo se muestra la evaluación de las primeras 10 oraciones de la primer carpeta, el resumen de la evaluación completa se muestra en las siguientes tablas:

Reconocimiento a nivel de palabra T44		
No. de palabras		25,990
Correctas		99.1%
Con error:		1.3%
	Con sustitución	0.6%
	Con inserciones	0.5%
	Con eliminación	0.2%

Tabla 1. Resultados del experimento 1 a nivel de palabra en el nivel T44.

Reconocimiento a nivel de palabra T22		
No. de palabras		25,990
Correctas		99.3%
Con error:		1.1%
	Con sustitución	0.5%
	Con inserciones	0.4%
	Con eliminación	0.2%

Tabla 2. Resultados del experimento 1 a nivel de palabra en el nivel T22.

Reconocimiento a nivel de elocución T44	
No. de elocuciones	2,498
Correctas	91.4%
Con error:	8.6%

Tabla 3. Resultados del experimento 1 a nivel de elocución en el nivel T44.

Reconocimiento a nivel de elocución T22	
No. de elocuciones	2,498
Correctas	92.7%
Con error:	7.3%

Tabla 4. Resultados del experimento 1 a nivel de elocución en el nivel T22.

Evaluación del Experimento 2

Los resultados del experimento 2 se muestran a continuación:

```
id: (s06305)
Scores: (#C #S #D #I) 9 3 1 1
REF:  el proyecto SE CENTRA  especialmente en personas mayores INTERESADAS en la educacio27n *** CONTINUA
HYP:  el proyecto ** SESENTA especialmente en personas mayores INTERESADOS en la educacio27n CON TIENE
Eval:             D  S                                        S                              I   S

id: (s06306)
Scores: (#C #S #D #I) 2 8 1 1
REF:  EL  menor DEBE de ***** SER ASISTIDO Y     PROTEGIDO PARA SER FORMADO
HYP:  DEL menor **** de VERDE SE  ASISTIR  CORTE GIRA      POR  SE  FORMA
Eval: S         D       I     S   S        S     S         S    S   S

id: (s06307)
Scores: (#C #S #D #I) 7 5 0 2
REF:  no obstante tambie27n se la puede ** considerar ****** COMO DISPOSICIO27N SUBJETIVA A  VENDER
HYP:  no obstante tambie27n se la puede DE considerar CO27MO DE   POSICIO27N    SUBJETIVO DE DE
Eval:                                   I             I      S    S             S         S  S

id: (s06308)
Scores: (#C #S #D #I) 6 6 1 1
REF:  ** AUNQUE bien sabemos que estos CONCEPTOS NO       SON FIELES de LA aptitud FI27SICA
HYP:  EN QUE    bien sabemos que estos CONCEPTO  MOCIO27N DE  LAS    de ** aptitud FI27SICAS
Eval: I  S                             S         S        S   S         D          S

id: (s06309)
Scores: (#C #S #D #I) 6 6 0 5
REF:  ** SINO en forma ** DIALE27CTICA es decir LLENO de *** *********** CONTRADICCIONES avances ** Y   RETROCESOS
HYP:  SI NO   en forma DE ELE27CTRICA  es decir LLENA de CON TRADICIO27N ES              avances EN LOS PROCESOS
Eval: I  S            I  S                     S        I   I          S                       I  S   S

id: (s06310)
Scores: (#C #S #D #I) 6 7 0 3
REF:  ** ** SIENDO TAMBIE27N UNA obra de consulta para LOS abogados ** Y       NOTARIOS EN ejercicio
HYP:  SI NO ESTA27 MIRA      NO  obra de consulta para LA  abogados EN MATERIA LOS      EL ejercicio
Eval: I  I  S      S         S                        S            I  S       S        S
```

```
id: (s06311)
Scores: (#C #S #D #I) 8 0 0 0
REF:  almacena las referencias a todos los nodos hijos
HYP:  almacena las referencias a todos los nodos hijos
Eval:

id: (s06312)
Scores: (#C #S #D #I) 4 2 0 2
REF:  universidad auto27noma de SAN luis *** *** POTOSI27
HYP:  universidad auto27noma de LA  luis CON TUS EN
Eval:                           S        I   I   S

id: (s06313)
Scores: (#C #S #D #I) 8 0 0 0
REF:  el sobre incluira27 tambie27n una breve sinopsis escrita
HYP:  el sobre incluira27 tambie27n una breve sinopsis escrita
Eval:

id: (s06314)
Scores: (#C #S #D #I) 2 4 1 1
REF:  NO HUBO  asi27 FRONTERAS que ******* PUDIERAN CONTENERLO
HYP:  ** NUEVO asi27 FRONTERA  que PUDIERA CONTENER NO
Eval: D  S           S             I       S        S

id: (s06315)
Scores: (#C #S #D #I) 8 0 0 0
REF:  comisio27n especial sobre la ordenacio27n del servicio farmace27utico
HYP:  comisio27n especial sobre la ordenacio27n del servicio farmace27utico
Eval:
```

Solo se muestra la evaluación de 10 oraciones de la primera carpeta, el resumen de la evaluación completa se muestra en las siguientes tablas:

Reconocimiento a nivel de palabra T44		
No. de palabras		4,752
Correctas		60.1%
Con error:		57.7%
	Con sustitución	36.9%
	Con inserciones	17.7%
	Con eliminación	3.1%

Tabla 5. Resultados del experimento 2 a nivel de palabra en el nivel T44.

Reconocimiento a nivel de palabra T22		
No. de palabras		4,752
Correctas		60.6%
Con error:		58.7%
	Con sustitución	36.5%
	Con inserciones	19.3%
	Con eliminación	2.9%

Tabla 6. Resultados del experimento 2 a nivel de palabra en el nivel T22.

Reconocimiento a nivel de elocución T44	
No. de elocuciones	450
Correctas	4.9%
Con error:	95.1%

Tabla 7. Resultados del experimento 2 a nivel de elocución en el nivel T44.

Reconocimiento a nivel de elocución T22	
No. de elocuciones	450
Correctas	4.9%
Con error:	95.1%

Tabla 8. Resultados del experimento 2 a nivel de elocución en el nivel T22.

Se puede observar muy claramente que en este experimento los resultados cambiaron drásticamente en comparación con el primero, esto se debe a que estos datos no se utilizaron para el entrenamiento, ni fueron tomados en cuenta para la construcción del modelo del lenguaje, por ello se puede tener una mayor certeza de lo precisos que son los modelos acústicos.

Evaluación del Experimento 3

Los resultados del experimento 3 se muestran a continuación:

```
id: (adolfo15)
Scores: (#C #S #D #I) 9 4 1 0
REF:  asi27 como EL AVISO comercial luz ciencia y tecnologi27a Y el logotipo DEL CIO
HYP:  asi27 como LA MISMO comercial luz ciencia y tecnologi27a * el logotipo DE  SIDO
Eval:            S  S                                          D             S   S

id: (adolfo2)
Scores: (#C #S #D #I) 2 2 2 1
REF:  NOMBRE DE la **** INSTITUCIO27N 0        empresa
HYP:  ****** ** la OBRA DE            ACCIO27N empresa
Eval: D      D     I    S             S

id: (adolfo3)
Scores: (#C #S #D #I) 3 2 1 0
REF:  por EL largo QUE SEA necesario
HYP:  por ** largo DE  LA  necesario
Eval:     D        S   S

id: (adolfo4)
Scores: (#C #S #D #I) 5 2 0 0
REF:  fichero histo27rico CON los mensajes ma27s INTERESANTES
HYP:  fichero histo27rico DE  los mensajes ma27s INTERESANTE
Eval:                     S                     S

id: (adolfo5)
Scores: (#C #S #D #I) 6 2 1 1
REF:  AYUDAS para EL manejo de herramientas informa27ticas y ************ TELEMA27TICAS
HYP:  AYUDA  para ** manejo de herramientas informa27ticas y TELEMA27TICA SE
Eval: S           D                                          I            S

id: (adolfo6)
Scores: (#C #S #D #I) 5 3 0 1
REF:  la comisio27n europea ES la ******* PATROCINADORA DEL evento
HYP:  la comisio27n europea DE la TURNADA PARA          DE  evento
Eval:                       S     I       S             S
```

```
id: (adolfo7)
Scores: (#C #S #D #I) 5 2 0 1
REF:  CONSULTA de ****** NORMATIVIDAD sobre el medio ambiente
HYP:  USO      de NORMAS ACTIVIDAD    sobre el medio ambiente
Eval: S           I      S

id: (adolfo8)
Scores: (#C #S #D #I) 7 4 0 0
REF:  convencio27n sobre la conservacio27n de LAS ESPECIES MIGRATORIAS de animales SILVESTRES
HYP:  convencio27n sobre la conservacio27n de LOS PECES    MIGRATORIOS de animales SILVESTRE
Eval:                                         S   S        S                      S

id: (adolfo9)
Scores: (#C #S #D #I) 5 5 1 6
REF:  ***** ** **** ** EXACTO EL   REMATE ES de la pa27gina ** ** y AHI27 esta27 PUBLICADO
HYP:  PESAR DE TODO LO QUE    RAMA Y      Y  de la pa27gina DE LA y ***** esta27 PUBLICADOS
Eval: I     I  I    I  S      S    S      S                 I  I    D            S
```

Solo se muestra la evaluación de 9 oraciones de una sola persona, el resumen de la evaluación completa se muestra a continuación:

Reconocimiento a nivel de palabra T44		
No. de palabras		1,232
Correctas		54.5%
Con error:		68.8%
	Con sustitución	41.6%
	Con inserciones	23.3%
	Con eliminación	3.8%

Tabla 9. Resultados del experimento 3 a nivel de palabra en el nivel T44.

Reconocimiento a nivel de palabras T22		
No. de palabras		1,232
Correctas		55.8%
Con error:		72.1%
	Con sustitución	40.8%
	Con inserciones	27.8%
	Con eliminación	3.4%

Tabla 10. Resultados del experimento 3 a nivel de palabra en el nivel T22.

Reconocimiento a nivel de elocución T44	
No. de elocuciones	120
Correctas	5%
Con error:	95.0%

Tabla 11. Resultados del experimento 3 a nivel de palabra en el nivel T44.

Reconocimiento a nivel de elocución T22	
No. de elocuciones	120
Correctas	3.3%
Con error:	96.7%

Tabla 12. Resultados del experimento 3 a nivel de elocución en el nivel T22.

Con los resultados obtenidos de los 3 experimentos, en ambos reconocedores, se puede decir, que los modelos acústicos y el modelo del lenguaje son muy aceptables.

5.2 Golem

Para evaluar el modelo del lenguaje creado para Golem se creó un dialogo que asemejara una conversación normal con el robot, es decir, que contuviera frases comunes que se utilizan en el contexto; los diálogos que se grabaron fueron los que simulaban las respuestas del usuario (Visitante).

El diálogo completo se muestra a continuación:

Golem	*Visitante*
Bienvenidos al Departamento de ciencias de la computación. ¿Quieres que te de una visita guiada?	*Sí, por favor*
Este es el departamento de ciencias de la computación, el departamento cuenta con las áreas de Inteligencia artificial y Reconocimiento de patrones ¿Qué área quieres que te explique?	*Inteligencia Artificial por favor.*
Inteligencia artificial tiene las áreas de personal y proyectos. ¿Qué quieres que te explique?	*Primero explícame los proyectos, por favor.*
Ok. Inteligencia artificial tiene los proyectos Dime, Dimex 100, Pitágoras, Golem y Reconocimiento Visual de Objetos. ¿Qué proyecto quieres que te explique?	*El tuyo Golem.*
Perdón, no te entendí.	*Que me expliques el tuyo Golem!*
Ok. Yo soy Golem. Tengo poco más de cinco años y me programaron para darte esta visita guiada. Estoy programado bajo una estructura de agentes abiertos. Tengo un agente reconocedor de voz, un ... ¿Quieres que te explique de mi manejador del diálogo?	*No ya no.*
Terminamos con el proyecto Golem. ¿Quieres que te siga explicando los proyectos de inteligencia artificial?	*Ok.*
Inteligencia artificial tiene los proyectos Dime, Dimex100, Pitágoras y Reconocimiento Visual de Objetos. ¿Qué proyecto quieres ver?	*El de reconocimiento visual de objetos.*
Los seres humanos son capaces de reconocer objetos y además pareciera que lo hacen con gran facilidad ... Este proyecto tiene participantes, objetivo, descripción y resultados. ¿Qué sección quieres que te explique?	*Resultados, por favor.*
Este video es un ejemplo de cómo una computadora ve el mundo y de cómo puede ir reconociendo los objetos que previamente ya conocía ... Terminamos con resultados. ¿Quieres que te siga explicando reconocimiento visual de	*No, gracias.*

objetos?	
Ok. ¿Quieres que te siga explicando los proyectos de inteligencia artificial?	***No.***
... Gracias por visitar el departamento de ciencias de la computación. Hasta luego.	

Este dialogo fue grabado por 5 personas diferentes. Para realizar las pruebas se utilizaron los modelos acústicos del nivel T44 creados con el 50% del corpus y el diccionario de pronunciación con 8,449 tokens y 5,466 tipos.

Los audios se grabaron con la herramienta *SpeechView* a 16,000 khz en formato wav, los cuales posteriormente con el programa *sox* se conviertieron a formato raw.

Las pruebas se realizaron con los tres modelos del lenguaje, los resultados obtenidos se muestran a continuación:

Modelo del Lenguaje[5]	Palabras Correctas	Error por Elocución
LM - 1	71.1%	36.4%
LM - 2	63.2%	45.5%
LM - 3	55.3%	63.6%
LM - 4	39.55	81.8%

Tabla 13. Resultados de reconocimiento con cada uno de los modelos del lenguaje.

[5] LM-1. Modelo del lenguaje con 3,615 oraciones
LM-2. Modelo del lenguaje con 7,473 oraciones
LM-3. Modelo del lenguaje con 11,561 oraciones
LM-4. Modelo del lenguaje que se utilizo para el entrenamiento.

Capítulo 6

Conclusiones

Durante el desarrollo de este trabajo me di cuenta que la creación de un sistema de reconocimiento de voz no es una tarea fácil sobre todo por la disponibilidad de los recursos necesarios – para el español - como son el corpus de entrenamiento y el modelo del lenguaje. Contar con el corpus DIMEx100 como un nuevo recurso para la construcción de tecnologías del habla fue una pieza clave para la conclusión de este trabajo; corpus que sin duda alguna es el reflejo del trabajo de investigación que se está realizando en México en el campo del procesamiento del lenguaje natural.

Contar con un corpus bien cuidado como lo es el corpus DIMEx100, resultó en obtener unos modelos acústicos de excelente calidad, esto lo pudimos comprobar con los resultados obtenidos en el reconocimiento, los cuales arrojaron porcentajes muy aceptables.

En el experimento uno el porcentaje de error a nivel de elocución fue de ≈8% y a nivel de palabra del ≈1%, considerando que se intento obtener la hipótesis de los mismos datos con los cuales fue entrenado el reconocedor, se puede decir que el resultado es muy satisfactorio. En el experimento dos el porcentaje de error fue en aumento – ≈57% a nivel de palabra y ≈95% a nivel de elocución – tomando en cuenta que las pruebas se hicieron con datos que no fueron utilizados durante el entrenamiento y además que el modelo del lenguaje no contemplaba información para poder generar estas frases, podemos decir que los resultados son satisfactorios. En el experimento tres se observo que se mantuvieron los porcentajes obteniendo un ≈68% a nivel de palabra y un ≈95% a nivel de elocución, en este experimento se utilizaron datos no utilizados durante el entrenamiento y además grabados por diferentes personas – ajenas a las que contribuyeron a grabar el corpus - en un ambiente no controlado; con esto se comprobó la calidad del corpus DIMEx100 para la construcción de modelos acústicos.

Como se puede observar estos porcentajes difieren muy poco entre los resultados obtenidos por el reconocedor construido en el nivel T22 y el construido en el nivel T44; con ello se puede concluir que para poder ver la eficiencia de un determinado nivel de granularidad se necesitan más datos de entrenamiento ya que con esta cantidad de datos es difícil apreciar que es mejor si un nivel de granularidad abstracto como lo es el nivel T22 para la construcción de un reconocedor de voz o un nivel de granularidad más especifico como lo es el nivel T44.

Una ventaja más que nos proporciono el uso del corpus DIMEx100 es que nos permitió construir reconocedores de voz de habla continua independientes del hablante debido a que fue grabado por 100 personas diferentes, ya que como vimos a lo largo de este trabajo existen factores externos que afectan el reconocimiento, entre ellos se pueden mencionar a los propios del hablante – como el estado de ánimo -, los relacionados con los medios de grabación – como el micrófono - etc.; además esto hace que el número de usuarios sea ilimitado.

Una ventaja más del uso de un corpus bien cuidado y etiquetado a mano es la obtención de diccionarios de pronunciación con muchas entradas por cada palabra, es decir, con pronunciaciones diferentes de una misma palabra y con ello obtener una mayor certeza de escoger la palabra correcta y que además no se esté sujeto a una sola pronunciación ya que no siempre se pronuncian las palabras de la misma forma sobre todo cuando la velocidad con la que se habla no es la misma, por ejemplo.

Sin embargo, la construcción de un sistema de reconocimiento de voz no solo depende de la buena calidad de los modelos acústicos o del extenso diccionario de pronunciación proporcionados por el corpus, sino también del modelo del lenguaje ya que es el que proporciona el conocimiento a priori para obtener así un buen reconocimiento de voz. Pero como vimos la construcción de un modelo del lenguaje no es una tarea trivial sobre todo porque si creamos un modelo del lenguaje basado en un corpus muy grande o que contenga frases poco relacionadas con el contexto nos arroja resultados poco satisfactorios

ya que la probabilidad de obtener la hipótesis adecuada se ve afectada debido a la ambigüedad o vaguedad en los datos utilizados.
En la construcción del modelo del lenguaje para el robot Golem nos enfrentamos a estos problemas por ello la decisión de crear corpus más pequeños basados en frases más apegadas a las tareas que realizaría Golem; así la idea de incorporar la mayor parte de los diálogos utilizados en una conversación típica con el robot, resultó en la obtención de porcentajes aceptables.

No obstante, a pesar de cuidar los recursos con los que se construirá un sistema de reconocimiento de voz, otros problemas a los que nos enfrentamos es a la hora de realizar el reconocimiento en vivo en donde no hay control del ruido alrededor del hablante, la distancia a la cual se habla al micrófono, la velocidad, el volumen, etc. Por ello la importancia de contar con sistemas de reconocimiento de voz independientes del hablante, que además cuenten con recursos cuidadosamente creados, es necesaria para reducir las fallas del sistema. Con estos problemas nos enfrentamos a la hora de poner en funcionamiento el reconocedor de voz con el robot, pero gracias al cuidado en la construcción de los recursos se obtuvieron muy buenos resultados. Esto se vio reflejado en los resultados arrojados por el reconocedor de voz al utilizar el modelo del lenguaje en conjunto con los modelos acústicos y el diccionario de pronunciación, ya que al utilizar el modelo del lenguaje creado específicamente para el uso del robot Golem se vio un incremento en los resultados del reconocedor en comparación con los resultados obtenidos al utilizar el mismo modelo del lenguaje que para la valoración de los modelos acústicos.

Aún así también se pudo comprobar que aunque nos podamos imaginar todos los posibles escenarios siempre van a aparecer nuevos y por lo tanto nuevas frases; por ello la importancia de tener un corpus lo suficientemente extenso para que si bien no podamos obtener la hipótesis correcta si la más parecida y sobre todo que ésta no pierda por completo la intención de lo que hablante dijo.

La obtención de estos resultados se dio gracias al trabajo y esfuerzo de todas las personas que han colaborado en el proyecto DIME; personas que dedicaron su tiempo a etiquetar y

entrenar a otras para etiquetar el corpus, siendo este el trabajo clave y extenuante dentro del proyecto. No dejando atrás toda la investigación para llegar a los tres niveles de granularidad (T22, T44 y T54) con que se etiqueta el corpus el cuál sin esta base seguramente no habríamos tenido los resultados obtenidos en las pruebas de reconocimiento. Teniendo en cuenta que el trabajo dentro del proyecto no solo se limita a la etiquetación y evaluación del corpus sino también a la integración de los recursos creados para dar vida al robot Golem, se tuvo la oportunidad de visitar el INAOE, Puebla, para la recolección de los datos que posteriormente formaron el corpus para el modelo del lenguaje creado para Golem. Sin duda alguna el formar parte de un equipo como lo es el grupo DIME, dan la oportunidad de colaborar en proyectos de investigación reales abriendo así muchas opciones para construir y aportar a la investigación en México.

Para obtener buenos resultados en la construcción de un sistema de reconocimiento de voz no solo es necesario contar con los recursos, sino también se necesita el conocimiento de las herramientas con las cuales se va a trabajar, en este caso Sphinx, la creación de recursos para la manipulación de la información, para el caso scripts en el shell y programas en java y perl, además contar con un buen equipo para el procesamiento de los datos y sobre todo con buena arquitectura de audio.

En un futuro muy cercano, debido a que casi se completa la etiquetación de todo el corpus DIMEx100, se podrán entrenar reconocedores de voz en los tres niveles de etiquetación (T22, T44 y T54) y así poder evaluar los modelos acústicos y entonces saber cuál nivel de granularidad arroja mejores resultados y así obtener un reconocedor de voz con mucho mejor calidad de reconocimiento.

Por otra parte ya teniendo los mejores resultados con la obtención de los modelos acústicos, entonces se tendría que trabajar en la recolección de los datos necesarios para crear el corpus que dará origen al modelo del lenguaje, que como recordaremos es la segunda parte importante para obtener una mejor calidad en el reconocimiento de la voz. Para ello sería conveniente crear una herramienta que recolecte información de Internet y que además se le puedan aplicar filtros para obtener los datos mucho más exactos. La idea

de recolectar la información a través de Internet, como ya explique en el capítulo 4 de este trabajo, es porque la mayoría de la gente que escribe lo hace de una forma desenfada utilizando frases que se usan comúnmente, además de que las escriben tal y como suenan sin importar las reglas gramaticales y esto es bueno de cierta forma puesto que cuando hablamos no siempre o más bien nunca respetamos las reglas gramaticales.

Otra forma de recolectar la gran cantidad de datos para crear el modelo del lenguaje sería grabar conversaciones con todas las posibles frases que serán utilizadas dentro del contexto, pasar esas oraciones a texto y así tener un modelo del lenguaje enfocado a un cierto contexto; aunque ello involucraría que no sean tomados en cuenta todos los posibles escenarios y además se arriesgaría a no poder generar nuevas frases. Una opción para poder enriquecerlo sería agregarle frases recolectadas a través de internet.

De esta manera tener un reconocedor de voz en español y además de buena calidad resulta ser la mejor herramienta para construir proyectos que ayuden al hombre a facilitar sus tareas diarias ya que no hay como utilizar el lenguaje hablado para comunicarse. Además estos sistemas son de mucha ayuda sobre todo para personas con alguna discapacidad, como por ejemplo para los invidentes o los que no cuentan con manos.

Hoy en día estos sistemas de reconocimiento de voz están tomando su importancia, una prueba de ello es que ya están siendo incorporados en los sistemas operativos y ahora por medio de la voz se le puede dar comandos para realizar operaciones básicas como puede ser cerrar o abrir aplicaciones. Una característica que aun se tiene que mejorar es que para hacer uso de estas herramientas y tener un buen funcionamiento hay que realizar una etapa de entrenamiento, en donde se le tiene que hablar al sistema por un lapso de 10 minutos aproximadamente esto es para que se tengan en cuenta los niveles que alcanza nuestra voz. A este reto nos enfrentamos al construir el reconocedor de voz, por ello se cuido la forma de recolectar los datos, así grabar las frases por 100 hablantes diferentes y además el uso de Sphinx, ayudaron a no necesitar pasar por esta etapa de entrenamiento para poder ser utilizado, aunque aún falta mucho camino por recorrer pero las expectativas que se tienen son muy buenas en base a los resultados que se han obtenido y

solo con el 50% del corpus, falta ver los resultados con el 100% del corpus y además con el trabajo de recolección para crear los modelos del lenguaje.

Apéndice 1

1. Los modelos acústicos

La construcción del reconocedor de voz se llevo a cabo en un ambiente linux Ubuntu, en el cuál se instalaron diversas herramientas utilizadas para preparar los datos requeridos. Estas herramientas son:

- SphinxTrain
- SPHINX decoder versión 3.4
- sox
- java
- perl

Para la creación de los modelos acústicos se utilizó la herramienta *SphinxTrain*, que forma parte de CMU SPHINX. Antes de llevar a cabo el entrenamiento de los modelos acústicos se requiere preparar los datos y el ambiente.

El ambiente se crea automáticamente con un script de *SphinxTrain* llamado ***setup_SphinTrain.pl***, éste genera una serie de directorios, en los cuales se guardan los archivos de configuración, los scripts de la herramienta, los datos necesarios para la creación de los modelos y los archivos de salida tanto los de errores como los temporales y de los modelos.

En total se crean diez directorios:

1. **bin**. Es el directorio en el cuál se copian algunos scripts para la manipulación de datos y se crean ligas hacia varios archivos ejecutables de la herramienta.
2. **bwaccumdir.** Es el directorio en el cuál se guardan archivos temporales que arroja el algoritmo Baum-Welch durante el entrenamiento. Estos contienen la acumulación de vectores para computar medias o varianzas.
3. **etc**. En este directorio se encuentra el archivo de configuración de *SphinxTrain*. Aquí es en donde se colocan los diccionarios, el archivo que contiene la trascripción

del corpus, el archivo de control y la lista de los modelos (en nuestro caso los alófonos del nivel T44).

4. **feat**. En este directorio se depositan los archivos de la extracción de características de los audios.
5. **gifs**. Este directorio contiene dos imágenes una que indica error y otra que indica acierto.
6. **logdir**. En este directorio se guardan todos los archivos log, estos contienen el proceso de entrenamiento.
7. **model_architecture**. En este directorio se encuentran los archivos de definición de los modelos, tanto para los modelos dependientes del contexto como los independientes, así como el archivo que contiene la topología de los modelos. Todos estos archivos definen la estructura de los HMM.
8. **model_parameters**. En este directorio se guardan todos los modelos acústicos.
9. **scripts_pl**. Este directorio contiene los scripts que corren cada uno de los nueve módulos que contiene *SphinxTrain.*
10. **wav**. En éste directorio se copian los archivos wav para el entrenamiento.

Adicionalmente cuando se corre el script del modulo 5 se genera un directorio extra llamado *trees,* en el cuál se guardan los archivos creados durante el módulo cinco del entrenamiento.

Una vez que se crea el ambiente con el que trabajará *SphinxTrain*, se realiza el llenado de los directorios ***etc, feat*** y ***wav.***

Los audios que se colocan dentro del directorio *wav*, tienen que estar grabados a 16KHz, como nuestros audios fueron grabados a 44.1KHz, se tuvieron que convertir. Para ello se utilizó la herramienta ***sox***. Se corrió el siguiente script para realizar la conversión:

```
#!/bin/csh

foreach cambio (`ls audio`)
        sox $cambio -r 16000 wav/$cambio dither
 end
```

En el directorio *feat* se colocan los archivos de características; para realizar la extracción de características se utilizó un script contenido en *SphinxTrain* llamado ***make_feats.pl*** ubicado dentro del directorio **bin**, el comando que se ejecuto fué el siguiente:

./wave2feat -c ctl -di wav -ei wav -do feat -eo mfc -dither yes -mswav yes -verbose yes

En este comando le decimos que los archivos están en formato wav (-ei) y que los archivos de salida deben de estar en mfc (-eo). En ambos casos se especifico la aplicación de dither a los audios.

Dentro del directorio **etc** se encuentra el archivo de configuración de *SphinxTrain* **sphinx_train.cfg**, este archivo contiene las rutas de los archivos y se especifican los valores de algunas variables que son necesarios para realizar el entrenamiento.

A continuación se muestra un ejemplo del archivo de configuración para el reconocedor en el nivel T44.

```
# Configuration script for sphinx trainer                -*-mode:Perl-*-

$CFG_VERBOSE = 1;              # Determines how much goes to the screen.

# These are filled in at configuration time
$CFG_DB_NAME = 'Dimex100_T44';
$CFG_BASE_DIR = '/home/reconocedor/Dimex100_T44';
$CFG_SPHINXTRAIN_DIR = '/usr/local/SphinxTrain';

# Directory containing SphinxTrain binaries
$CFG_BIN_DIR = "$CFG_BASE_DIR/bin";
$CFG_GIF_DIR = "$CFG_BASE_DIR/gifs";
$CFG_SCRIPT_DIR = "$CFG_BASE_DIR/scripts_pl";

# Experiment name, will be used to name model files and log files
$CFG_EXPTNAME = "$CFG_DB_NAME";
$CFG_FEATFILES_DIR = "$CFG_BASE_DIR/feat";
$CFG_FEATFILE_EXTENSION = 'mfc';
$CFG_VECTOR_LENGTH = 13;
$CFG_MIN_ITERATIONS = 1;  # BW Iterate at least this many times
$CFG_MAX_ITERATIONS = 30; # BW Don't iterate more than this, somethings likely
wrong.
```

Algunas de las especificaciones que se pueden hacer son las siguientes:

- **FEATFILE_EXTENSION.** El tipo de los archivos a procesar.
- **MAX_ITERATIONS, MIN_ITERATIONS.** Número máx. y mín. de iteraciones del algoritmo Baum-Welch.
- **STATESPERHMM.** Número de estados del Modelo Oculto de Markov.
- **HMM_TYPE.** Tipo de modelo a generar ya sea continuo o semicontinuo.
- **N_TIED_STATES.** Debe ser un valor entre 500 y 2500, el cuál especifica el número total de distribuciones de estado compartidas del grupo final de HMM entrenados (los modelos acústicos).
- **CONVERGENCE_RATIO.** Es un número que puede ir de 0.1 a 0.001, el cuál especifica la proporción entre la diferencia en probabilidad de la iteración actual y la anterior de Baum-Welch, y la probabilidad total de la iteración anterior. Cuantas más iteraciones de Baum-Welch se ejecuten, mejor se aprenderán las distribuciones de sus datos.
- **GAUSSIANSPERSTATE.** Específica el número de gaussianas por estado que va de 4 a 32; en el caso de tener pocos datos, es recomendable que el número de gaussianas no sea mayor a 8.

Los datos requeridos por *SphinxTrain* para realizar el proceso de entrenamiento son:

- Las señales acústicas (audios) convertidos a algún formato aceptado por SPHINX; en este caso fueron convertidos a MFCC.
- El correspondiente archivo de transcripción.
- El diccionario de pronunciación.
- El diccionario con sonidos de relleno[1].
- Un archivo con la lista de las unidades acústicas para los modelos que se quieren entrenar (en nuestro caso los alófonos del nivel T44).
- Un archivo de control que contenga sólo el nombre (sin extensión) del conjunto de archivos de audio a los cuales se les extrajeron sus características.

[1] Este diccionario contiene las representaciones de los sonidos que no forman parte de una palabra, por ejemplo las etiquetas de ruido o silencio. En este caso en particular, este archivo contiene las representaciones del silencio inicial y final de las oraciones, así como las etiquetas que representan al ruido.

Una vez que se tienen todos los archivos, se realiza el entrenamiento de los modelos acústicos, para ello se corre el script ***run_all.pl,*** el cuál ejecuta cada uno de los nueve módulos que tiene *SphinxTrain*.

- **MODULO 00**. Es el módulo de verificación de los archivos de entrenamiento.
    1. Se asegura de que los diccionarios (tanto de pronunciación como de sonidos de relleno) sean congruentes con la lista de las unidades acústicas.
    2. Observa que no haya entradas repetidas en el diccionario.
    3. Verifica que existan todos los archivos que se listan en el archivo de control.
    4. Observa que el número de líneas en el archivo de trascripción sea el mismo que en el archivo de control.
    5. Determina si la cantidad de datos de entrada es suficiente para empezar el entrenamiento y que todas las palabras en la trascripción aparezcan en el diccionario.

  El script que corre este módulo es: **scripts_pl/00.Verify/verify_all.pl**
- **MODULO 01.** Es el módulo donde se realiza la cuantificación de vectores, esta es una técnica de codificación que ha sido aplicada con éxito tanto a compresión de habla como de imágenes. Consiste en agregar los vectores de características a un solo archivo para después, realizar el cómputo de los centroides en el espacio de vectores. No aplica en el caso de modelos continuos.

  El script que corre este módulo es: **scripts_pl/01.Vector_quantize/slave.VQ.pl**
- **MODULO 02.** Es el módulo de entrenamiento de los modelos independientes del contexto. Aquí se realizan las iteraciones con el algoritmo Baum-Welch.

  El script que corre este módulo es: **scripts_pl/02.ci_shmm/slave_convg.pl**
- **MODULO 03.** Es el módulo donde se ligan los modelos y se crean los llamados trifonemas.

  El script que corre este módulo es:

  scripts_pl/03.makeuntiedmdef/make_untied_mdef.pl
- **MODULO 04.** En este módulo se vuelve a iterar con el algoritmo Baum-Welch y se crean los modelos dependientes del contexto.

  El script que corre este módulo es: **scripts_pl/04.cd_shmm/slave_convg.pl**
- **MODULO 05a**. En este módulo se construyen árboles de decisión. El script que corre este módulo es: **scripts_pl/05.buildtrees/make_questions.pl**

- **MODULO 05b**. En este módulo se construyen árboles de decisión para cada estado del HMM. El script que corre este módulo es:
  scripts_pl/05.buildtrees/slave_treebuilder.pl
- **MODULO 06.** En este módulo se podan los árboles anteriores. El script que corre este módulo es: **scripts_pl/06.prunetree/slave_state_tie_er.pl**
- **MODULO 07.** En éste módulo, se reestrenan los modelos dependientes del contexto hasta determinado número de gaussianas. El script que corre este módulo es: **scripts_pl/07.cd_shmm/slave_convg.pl**
- **MODULO 08.** Es el módulo donde se realiza el borrado de interpolaciones. No aplica en el caso de modelos continuos. El script que corre este módulo es:
  scripts_pl/08.deleted_interpolation/deleted_interpolationl.pl
- **MODULO 09.** En éste módulo se realiza la conversión de los modelos al formato de SPHINX 2. No es necesario en el caso de modelos para SPHINX 3. El script que corre este módulo es: **scripts_pl/s2_models/make_s2_models.pl**

Apéndice 2

1. El diccionario de pronunciación

Para construir el diccionario de pronunciación se utilizó directamente la etiquetación del corpus en sus niveles Tp (palabras) y T44 (alófonos). Para obtener la representación de la palabra en alófonos, se utilizó un programa hecho en java que se ejecuta con un script en shell, lo que hace el programa es tomar el tiempo inicial y final de la palabra en el correspondiente archivo de palabras y busca su alineación en el archivo que contiene las etiquetas de alófonos (T44 en este caso).

A continuación se muestra un ejemplo de los archivos de los cuales se obtienen los datos y los parámetros tomados para crear el diccionario.

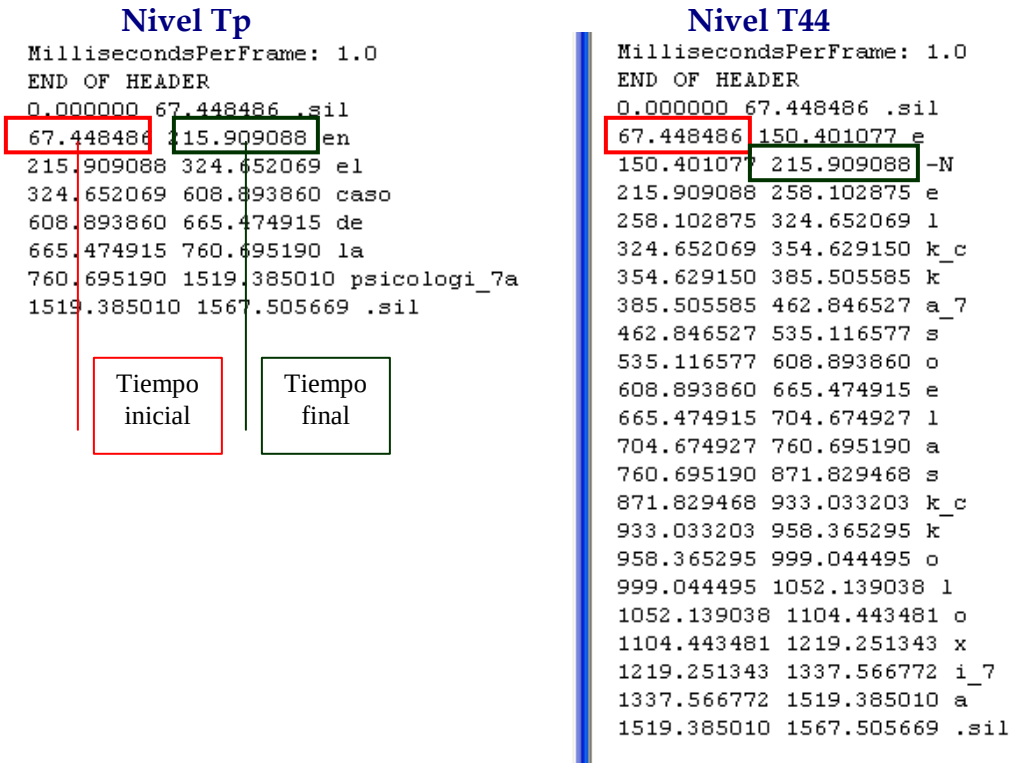


Figura 1. Ejemplo de alineación del nivel Tp con el nivel T44.

El programa en java (***Diccionario.java***) recibe como parámetros el archivo Tp y el archivo T44, la ruta para que encuentre estos archivos se la proporciona el script en shell

(***crea_dic.sh***). El script en shell además de ejecutar el programa en java y proporcionarle las rutas, se encarga de ordenar los datos obtenidos alfabéticamente y de verificar que no se guarden entradas iguales.

El programa ***Diccionario.java*** se muestra a continuación:

```
import java.lang.*;
import java.io.*;
import java.util.*;

public class Diccionario {

   public static void main(String []args) throws IOException{
      String s, s2, diccionario = new String();
      String palabrat44 = "";
      String primera = "";
      Vector periodo = new Vector();
      double punto=1;

      BufferedReader in = new BufferedReader(new FileReader(args[0]));
      BufferedReader t44 = new BufferedReader(new FileReader(args[1]));

      while ((s = in.readLine()) != null){
         String [] token = s.split(" ");
         for ( int k = 0; k < token.length; ++k ){
            periodo.add(token[k]);
         }
      }
      in.close();

      String[] palabra = new String[periodo.size()];
      palabra = (String[])periodo.toArray(palabra);

      for (int i=9;i<palabra.length-3;i+=3){
         while ((s2 = t44.readLine()) != null){
            String[] fonema = s2.split(" ");
            try{
               if (Double.parseDouble(fonema[0]) < Double.parseDouble(palabra[i]) &&
Double.parseDouble(fonema[0]) > 0 && Double.parseDouble(fonema[0]) >= punto){
                  palabrat44 = palabrat44.concat(fonema[2]).concat(" ");
               }else{
                  if(Double.parseDouble(fonema[0]) == Double.parseDouble(palabra[i])){
                     punto = Double.parseDouble(palabra[i]);
                      primera = fonema[2];
                     break;
```

```
                    }
                  }
               }catch(Exception e){}
            }
            System.out.println(palabra[i+1].toUpperCase()+"\t"+palabrat44);
            palabrat44=primera+" ";
        }
        t44.close();
    }
}
```

El script ***crea_dic.sh*** que ejecuta el programa ***Diccionario.java*** se muestra a continuación:

```
#!/bin/csh

javac Diccionario.java

foreach enunciado (`ls ruta_archivos`)
          java Diccionario "tp/individuales/$enunciado" "t44/individuales/$enunciado" >>
Dimex100_T44_prueba.dic
end

cat Dimex100_T44_prueba.dic | sed 's/_7/27/g' > temp.dic

sort -k1 temp.dic | uniq > Dimex100_T44_prueba.dic

cat Dimex100_T44_prueba.dic | uniq | cut -f1 | sort | uniq > agrega1.txt
foreach ordena (`cat agrega1.txt`)
   grep -w ^$ordena Dimex100_T44_prueba.dic >> Dimex100_T44_prueba2.dic
end

cat Dimex100_T44_prueba2.dic | sed 's/r(/r1/g' > etc/Dimex100_T44.dic

rm -rf temp.dic
rm -rf Dimex100_T44_prueba.dic
rm -rf Dimex100_T44_prueba2.dic
rm –rf agrega1.txt
```

En este script se asume que los archivos tanto del nivel Tp como del nivel T44 tienen el mismo nombre.

Lo que se obtuvo al final fue:

```
EN     e
EN(2)   e_7 -N
EN(3)   e l
EN(4)   e n
```

```
EN(5)   e -N
EN(6)   i -N
EN(7)   n
EN(8)   -N
ENAMORADOS      e n a m o r( a_7 D o s
ENAMORAR        n a m o r( a_7 -R
ENANOS  e n a_7 n o s
ENCABEZADO      -N k_c k a V e s a_7 D o
ENCANTO e -N k_c k a_7 -N t_c t o
ENCARGADO       e -N k_c k a -R G a_7 D o
ENCARGADO(2)    e -N k_c k a -R g_c g a_7 D o
```

2. Golem

La creación del diccionario creado para golem se hizo en dos fases: 1) Se copiaron las palabras del diccionario creado para el reconocedor T44 que estaban dentro del vocabulario de golem. 2) Para las palabras que no se encontraban en el diccionario se creó un programa en java (Golem.java) para asignarles su transcripción canónica en el nivel T44.

El programa se muestra a continuación:

```
public class Golem {

   public Golem() {
   }

   public static void main(String[] args) {
      String palabra = args[0];
      String transcripcion = "";

      for(int i=0;i<palabra.length();i++){
         char letra = palabra.toLowerCase().charAt(i);
         switch (letra){
            case 'p':
               if (palabra.toLowerCase().charAt(i+1) != 'o' && palabra.toLowerCase().charAt(i+1) != 'ó' &&
                     palabra.toLowerCase().charAt(i+1) != 'a' && palabra.toLowerCase().charAt(i+1) != 'á' &&
                     palabra.toLowerCase().charAt(i+1) != 'e' && palabra.toLowerCase().charAt(i+1) != 'é' &&
                     palabra.toLowerCase().charAt(i+1) != 'u' && palabra.toLowerCase().charAt(i+1) != 'ú' &&
                     palabra.toLowerCase().charAt(i+1) != 'i' && palabra.toLowerCase().charAt(i+1) != 'í' &&
                     palabra.toLowerCase().charAt(i+1) != 'l' && palabra.toLowerCase().charAt(i+1) != 'r')
                  transcripcion = transcripcion.concat("-B ");
               else
                  transcripcion = transcripcion.concat("p_c p ");
               break;
            case 'k':
               if (palabra.toLowerCase().charAt(i+1) != 'o' && palabra.toLowerCase().charAt(i+1) != 'ó' &&
                     palabra.toLowerCase().charAt(i+1) != 'a' && palabra.toLowerCase().charAt(i+1) != 'á' &&
                     palabra.toLowerCase().charAt(i+1) != 'e' && palabra.toLowerCase().charAt(i+1) != 'é' &&
```

```
            palabra.toLowerCase().charAt(i+1) != 'u' && palabra.toLowerCase().charAt(i+1) != 'ú' &&
            palabra.toLowerCase().charAt(i+1) != 'i' && palabra.toLowerCase().charAt(i+1) != 'í' &&
            palabra.toLowerCase().charAt(i+1) != 'l' && palabra.toLowerCase().charAt(i+1) != 'r')
          transcripcion = transcripcion.concat("-G ");
        else
          transcripcion = transcripcion.concat("k_c k ");
        break;
      case 't':
        if (palabra.toLowerCase().charAt(i+1) != 'o' && palabra.toLowerCase().charAt(i+1) != 'ó' &&
            palabra.toLowerCase().charAt(i+1) != 'a' && palabra.toLowerCase().charAt(i+1) != 'á' &&
            palabra.toLowerCase().charAt(i+1) != 'e' && palabra.toLowerCase().charAt(i+1) != 'é' &&
            palabra.toLowerCase().charAt(i+1) != 'u' && palabra.toLowerCase().charAt(i+1) != 'ú' &&
            palabra.toLowerCase().charAt(i+1) != 'i' && palabra.toLowerCase().charAt(i+1) != 'í' &&
            palabra.toLowerCase().charAt(i+1) != 'l' && palabra.toLowerCase().charAt(i+1) != 'r')
          transcripcion = transcripcion.concat("-D ");
        else
          transcripcion = transcripcion.concat("t_c t ");
        break;
      case 'b':
        if (i == 0)
          transcripcion = transcripcion.concat("b_c b ");
        else{
          if (palabra.toLowerCase().charAt(i+1) != 'o' && palabra.toLowerCase().charAt(i+1) != 'ó' &&
              palabra.toLowerCase().charAt(i+1) != 'a' && palabra.toLowerCase().charAt(i+1) != 'á' &&
              palabra.toLowerCase().charAt(i+1) != 'e' && palabra.toLowerCase().charAt(i+1) != 'é' &&
              palabra.toLowerCase().charAt(i+1) != 'u' && palabra.toLowerCase().charAt(i+1) != 'ú' &&
              palabra.toLowerCase().charAt(i+1) != 'i' && palabra.toLowerCase().charAt(i+1) != 'í' &&
              palabra.toLowerCase().charAt(i+1) != 'l' && palabra.toLowerCase().charAt(i+1) != 'r')
            transcripcion = transcripcion.concat("-B ");
          else{
            if (palabra.toLowerCase().charAt(i-1) == 'm' || palabra.toLowerCase().charAt(i-1) == 'n')
              transcripcion = transcripcion.concat("b_c b ");
            else
              transcripcion = transcripcion.concat("V ");
          }
        }
        break;
      case 'd':
        if (i == 0)
          transcripcion = transcripcion.concat("d_c d ");
        else{
          if (palabra.toLowerCase().charAt(i+1) != 'o' && palabra.toLowerCase().charAt(i+1) != 'ó' &&
              palabra.toLowerCase().charAt(i+1) != 'a' && palabra.toLowerCase().charAt(i+1) != 'á' &&
              palabra.toLowerCase().charAt(i+1) != 'e' && palabra.toLowerCase().charAt(i+1) != 'é' &&
              palabra.toLowerCase().charAt(i+1) != 'u' && palabra.toLowerCase().charAt(i+1) != 'ú' &&
              palabra.toLowerCase().charAt(i+1) != 'i' && palabra.toLowerCase().charAt(i+1) != 'í' &&
              palabra.toLowerCase().charAt(i+1) != 'l' && palabra.toLowerCase().charAt(i+1) != 'r')
            transcripcion = transcripcion.concat("-D ");
          else{
            if (palabra.toLowerCase().charAt(i-1) == 'm' || palabra.toLowerCase().charAt(i-1) == 'n')
              transcripcion = transcripcion.concat("d_c d ");
            else
              transcripcion = transcripcion.concat("D ");
          }
        }
        break;
      case 'g':
        if (i == 0){
```

```
            if (palabra.toLowerCase().charAt(i+1) == 'e' || palabra.toLowerCase().charAt(i+1) == 'i' ||
palabra.toLowerCase().charAt(i+1) == 'é' || palabra.toLowerCase().charAt(i+1) == 'í')
               transcripcion = transcripcion.concat("x ");
            else
               transcripcion = transcripcion.concat("g_c g ");
            //System.out.println(palabra.toLowerCase().charAt(i+1));
         }else{
            if (palabra.toLowerCase().charAt(i+1) != 'o' && palabra.toLowerCase().charAt(i+1) != 'ó' &&
                palabra.toLowerCase().charAt(i+1) != 'a' && palabra.toLowerCase().charAt(i+1) != 'á' &&
                palabra.toLowerCase().charAt(i+1) != 'u' && palabra.toLowerCase().charAt(i+1) != 'ú' &&
                palabra.toLowerCase().charAt(i+1) != 'e' && palabra.toLowerCase().charAt(i+1) != 'é' &&
                palabra.toLowerCase().charAt(i+1) != 'i' && palabra.toLowerCase().charAt(i+1) != 'í' &&
                palabra.toLowerCase().charAt(i+1) != 'l' && palabra.toLowerCase().charAt(i+1) != 'r')
               transcripcion = transcripcion.concat("-G ");
            else{
               if (palabra.toLowerCase().charAt(i-1) == 'm' || palabra.toLowerCase().charAt(i-1) == 'n')
                  transcripcion = transcripcion.concat("g_c g ");
               else{
                  if (palabra.toLowerCase().charAt(i+1) == 'e' || palabra.toLowerCase().charAt(i+1) == 'i' ||
palabra.toLowerCase().charAt(i+1) == 'é' || palabra.toLowerCase().charAt(i+1) == 'í')
                     transcripcion = transcripcion.concat("x ");
                  else{
                     transcripcion = transcripcion.concat("G ");
                  }
               }
            }
         }
         break;
      case 'c':
         if (palabra.toLowerCase().charAt(i+1) == 'h')
            transcripcion = transcripcion.concat("tS_c tS ");
         else{
            if (palabra.toLowerCase().charAt(i+1) == 'e' || palabra.toLowerCase().charAt(i+1) == 'i' ||
palabra.toLowerCase().charAt(i+1) == 'í')
               transcripcion = transcripcion.concat("s ");
            else
               transcripcion = transcripcion.concat("k_c k ");
         }
         break;
      case 'h':
         break;
      case 'f':
         transcripcion = transcripcion.concat("f ");
         break;
      case 's':
         transcripcion = transcripcion.concat("s ");
         break;
      case 'j':
         transcripcion = transcripcion.concat("x ");
         break;
      case 'm':
         if (palabra.toLowerCase().charAt(i+1) != 'o' && palabra.toLowerCase().charAt(i+1) != 'ó' &&
             palabra.toLowerCase().charAt(i+1) != 'a' && palabra.toLowerCase().charAt(i+1) != 'á' &&
             palabra.toLowerCase().charAt(i+1) != 'e' && palabra.toLowerCase().charAt(i+1) != 'é' &&
             palabra.toLowerCase().charAt(i+1) != 'u' && palabra.toLowerCase().charAt(i+1) != 'ú' &&
             palabra.toLowerCase().charAt(i+1) != 'i' && palabra.toLowerCase().charAt(i+1) != 'í')
            transcripcion = transcripcion.concat("-N ");
         else
```

```
        transcripcion = transcripcion.concat("m ");
    break;
case 'n':
    if (palabra.toLowerCase().charAt(i+1) != 'o' && palabra.toLowerCase().charAt(i+1) != 'ó' &&
          palabra.toLowerCase().charAt(i+1) != 'a' && palabra.toLowerCase().charAt(i+1) != 'á' &&
          palabra.toLowerCase().charAt(i+1) != 'e' && palabra.toLowerCase().charAt(i+1) != 'é' &&
          palabra.toLowerCase().charAt(i+1) != 'u' && palabra.toLowerCase().charAt(i+1) != 'ú' &&
          palabra.toLowerCase().charAt(i+1) != 'i' && palabra.toLowerCase().charAt(i+1) != 'í')
        transcripcion = transcripcion.concat("-N ");
    else
        transcripcion = transcripcion.concat("n ");
    break;
case 'ñ':
    transcripcion = transcripcion.concat("n~ ");
    break;
case 'r':
    if (palabra.toLowerCase().charAt(i+1) != 'o' && palabra.toLowerCase().charAt(i+1) != 'ó' &&
          palabra.toLowerCase().charAt(i+1) != 'a' && palabra.toLowerCase().charAt(i+1) != 'á' &&
          palabra.toLowerCase().charAt(i+1) != 'e' && palabra.toLowerCase().charAt(i+1) != 'é' &&
          palabra.toLowerCase().charAt(i+1) != 'u' && palabra.toLowerCase().charAt(i+1) != 'ú' &&
          palabra.toLowerCase().charAt(i+1) != 'i' && palabra.toLowerCase().charAt(i+1) != 'í' &&
          palabra.toLowerCase().charAt(i+1) != 'r')
        transcripcion = transcripcion.concat("-R ");
    else{

        if (palabra.toLowerCase().charAt(i+1) == 'r'){
          transcripcion = transcripcion.concat("r ");
          i++;
        }else
          transcripcion = transcripcion.concat("r( ");
    }
    break;
case 'l':
    if (palabra.toLowerCase().charAt(i+1) == 'l'){
        transcripcion = transcripcion.concat("Z ");
        i++;
    }else
        transcripcion = transcripcion.concat("l ");
    break;
case 'a':
    transcripcion = transcripcion.concat("a ");
    break;
case 'e':
    transcripcion = transcripcion.concat("e ");
    break;
case 'i':
    if (palabra.toLowerCase().charAt(i+1) == 'o' || palabra.toLowerCase().charAt(i+1) == 'ó' ||
          palabra.toLowerCase().charAt(i+1) == 'a' || palabra.toLowerCase().charAt(i+1) == 'á' ||
          palabra.toLowerCase().charAt(i+1) == 'e' || palabra.toLowerCase().charAt(i+1) == 'é' ||
          palabra.toLowerCase().charAt(i+1) == 'u' || palabra.toLowerCase().charAt(i+1) == 'ú')
        transcripcion = transcripcion.concat("j ");
    else
        transcripcion = transcripcion.concat("i ");
    break;
case 'o':
    transcripcion = transcripcion.concat("o ");
    break;
case 'u':
```

```
            if (palabra.toLowerCase().charAt(i-1) == 'g' && (palabra.toLowerCase().charAt(i+1) == 'e' ||
palabra.toLowerCase().charAt(i+1) == 'i' || palabra.toLowerCase().charAt(i+1) == 'í')){
               break;
            }else{
               if (palabra.toLowerCase().charAt(i+1) == 'o' || palabra.toLowerCase().charAt(i+1) == 'ó' ||
                     palabra.toLowerCase().charAt(i+1) == 'a' || palabra.toLowerCase().charAt(i+1) == 'á' ||
                     palabra.toLowerCase().charAt(i+1) == 'e' || palabra.toLowerCase().charAt(i+1) == 'é' ||
                     palabra.toLowerCase().charAt(i+1) == 'i' || palabra.toLowerCase().charAt(i+1) == 'í')
                  transcripcion = transcripcion.concat("w ");
               else
                  transcripcion = transcripcion.concat("u ");
            }
            break;
         case 'á':
            transcripcion = transcripcion.concat("a_7 ");
            break;
         case 'é':
            transcripcion = transcripcion.concat("e_7 ");
            break;
         case 'í':
            transcripcion = transcripcion.concat("i_7 ");
            break;
         case 'ó':
            transcripcion = transcripcion.concat("o_7 ");
            break;
         case 'ú':
            transcripcion = transcripcion.concat("u_7 ");
            break;
         case 'q':
            if (palabra.toLowerCase().charAt(i+1) == 'u'){
               transcripcion = transcripcion.concat("k_c k ");
               i++;
            }
            break;
         case 'v':
            if (i == 0)
               transcripcion = transcripcion.concat("b_c b ");
            else{
               if (palabra.toLowerCase().charAt(i-1) == 'm' || palabra.toLowerCase().charAt(i-1) == 'n')
                  transcripcion = transcripcion.concat("b_c b ");
               else
                  transcripcion = transcripcion.concat("V ");
            }
            break;
         case 'w':
            transcripcion = transcripcion.concat("G u ");
            break;
         case 'z':
            transcripcion = transcripcion.concat("s ");
            break;
         case 'x':
            transcripcion = transcripcion.concat("k_c k s ");
            break;
         case 'y':
            if (palabra.length()-1 == i){
               if (palabra.toLowerCase().charAt(i-1) == 'o' || palabra.toLowerCase().charAt(i-1) == 'ó' ||
                     palabra.toLowerCase().charAt(i-1) == 'a' || palabra.toLowerCase().charAt(i-1) == 'á' ||
                     palabra.toLowerCase().charAt(i-1) == 'e' || palabra.toLowerCase().charAt(i-1) == 'é' ||
```

```
                    palabra.toLowerCase().charAt(i-1) == 'i' || palabra.toLowerCase().charAt(i-1) == 'í')
                transcripcion = transcripcion.concat("j ");
              else
                transcripcion = transcripcion.concat("i ");
            }else
              transcripcion = transcripcion.concat("Z ");
            break;
          default :
            System.out.println("Esta letra no pertenece al vocabulario");
            break;
        }
      }
      System.out.println(palabra + "\t" + transcripcion);
    }

}
```

Para realizar este programa se tomaron en cuenta las reglas de pronunciación del nivel T54 (Tabla 5) y las tablas del nivel T44 (Tablas 10 a 13).

Apéndice 3

1. El modelo del lenguaje

Para la construcción del modelo del lenguaje se utilizó la herramienta CMU Statistical Language Model Tolkit versión 2.05 y las transcripciones del corpus DIMEx100. Las transcripciones se obtuvieron de 50 carpetas del corpus DIMEx100, las cuales equivalen a 2,498 oraciones.

El formato que pide la herramienta de CMU para las transcripciones es el siguiente:

<s> EN EL CASO DE LA PSICOLOGI27A </s>
<s> DE LA CIUDAD DE ME27XICO PARA EL MUNDO </s>

Debe haber un inicio (<s>) y un fin (</s>) por cada oración.
Para facilidad con los comandos, se asume que el archivo que contiene las transcripciones se llama *archivo.trans*.

A continuación se muestran los comandos utilizados para crear el modelo del lenguaje; éstos se encuentran dentro de la carpeta *CMUStisticalLanguageModelTolkit* creada por la herramienta.

1. text2wfreq

cat archivo.trans | **text2wfreq** | **sort –rn –k** 2 > archivo.wfreq

2. wfreq2vocab

cat archivo.wfreq | **wfreq2vocab –gt** 0 > archivo.vocab

El parámetro –gt indica el número de veces que debe aparecer una palabra para ser incluida en el vocabulario.

3. text2wngram

cat archivo.trans | **text2wngram –n** 3 **–temp** /tmp > archivo.w3gram

El parámetro –n indica el orden del modelo del lenguaje, es decir, si es de 3-gramas, 2-gramas, etc., por default es 3. El parámetro –temp indica la ruta en donde se guardarán los archivos temporales que arroja este comando.

4. wngram2idngram

cat archivo.w3gram | **wngram2idngram –n** 3 **–vocab** corpus.vocab **–temp** /tmp > archivo.id3gram

En este comando se incluyen igualmente los parámetros –n y –temp, además del parámetro –vocab que solo indica el archivo que contiene el vocabulario creado anteriormente.

5. idngram2lm

idngram2lm –idngram archivo.id3gram **–vocab** archivo.vocab **–context** con.ccs **–witten_bell –n** 3 **–vocab_type** 0 **–arpa** archivo3g.lm

El parámetro –context permite especificar el archivo que contiene los símbolos de inicio y fin de cada oración, en este caso se incluyen los símbolos <s> y </s>. El parámetro –vocab_type indica el tipo de vocabulario, es decir, si es abierto (1) o cerrado (0). Por último con el parámetro –arpa se especifica el nombre del archivo de salida.

Además este comando permite especificar la técnica de suavizado, entre las opciones disponibles están: Good Turing y Witten bell; ésta última fue la que se escogió para el modelo.

6. lm3g2dmp

lm3g2dmp archivo3g.lm rutaendondeseguarda

Apéndice 4

1. Decodificación o reconocimiento

Para realizar el reconocimiento se utilizó Sphinx *decoder* versión 3.4. Los archivos que necesita Sphinx *decoder* son:

a) El diccionario de pronunciación.
b) El modelo del lenguaje.
c) El archivo con sonidos de relleno.
d) Los modelos acústicos.
e) Los datos de prueba.
f) El archivo que contiene los sub-vectores de las densidades Gaussianas.

Para crear el archivo que contendrá las gaussianas se utilizó la herramienta ***gausubvq*** que se encuentra dentro del paquete de Sphinx3. El comando que se utilizó fue el siguiente:

/Sphinx3/src/programs/gausubvq -mean model_parameters/means -var model_parameters/variances -mixw model_parameters/mixture_weights -svspec 24,0-11/25 -subvq subvq

En donde los datos son obtenidos de los modelos acústicos.

La herramienta que se utilizó para realizar el reconocimiento fue ***livepretend***. El comando utilizado fue:

$**>livepretend** *ctl-file audio-dir arg-file*

En donde,

ctl-file – Este archivo contiene el nombre de los archivos de audio que se utilizaran en las pruebas.

audio-dir – En este directorio se encuentran los archivos de audio que se utilizaran para las pruebas. Estos archivos de audio se encuentran en formato "raw". Como los archivos estaban en formato wav se utilizó "sox". El comando ejecutado fue:

sox audio.wav -r 16000 audio.raw dither

arg-file – En este archivo se encuentran especificadas las rutas de los datos que se necesitan para realizar el reconocimiento, es decir, contiene los parámetros necesarios. Un ejemplo del contenido del archivo se muestra a continuación:

```
-mdef           /home/reconocedores_T44/reconocedor_T44_50%/model_architecture/Dimex100_T44.1000.mdef
-mean           /home/reconocedores_T44/reconocedor_T44_50%/model_parameters/Dimex100_T44.cd_cont_1000/means
-var            /home/reconocedores_T44/reconocedor_T44_50%/model_parameters/Dimex100_T44.cd_cont_1000/varianc
es
-mixw           /home/reconocedores_T44/reconocedor_T44_50%/model_parameters/Dimex100_T44.cd_cont_1000/mixture
_weights
-tmat           /home/reconocedores_T44/reconocedor_T44_50%/model_parameters/Dimex100_T44.cd_cont_1000/transit
ion_matrices
-subvq          /home/reconocedores_T44/reconocedor_T44_50%/subvq
-fdict          /home/reconocedores_T44/reconocedor_T44_50%/etc/Dimex100_T44.filler
-feat           1s_c_d_dd
-upperf         6855.49756
-lowerf         133.33334
-nfilt          40
-nfft           512
-samprate       16000
-dict           /home/reconocedores_T44/reconocedor_T44_50%/etc/Dimex100_T44.dic
-lm             /home/reconocedores_T44/reconocedor_T44_50%/Corpus_T44_50%.mod-3g.lm.DMP
-agc            max
-varnorm        no
-cmn            current
-subvqbeam      1e-02
-epl            4
-fillprob       0.02
-lw             9.5
-maxwpf         10
-beam           1e-60
-wbeam          1e-35
-reportpron     0
-reportfill     0
-outrawdir      /root/dimex100/rawfiles/basura/
"dimex.args" 29L, 1359C                                                    1,1      Comienzo
```

2. Evaluación del reconocimiento.

Para evaluar el porcentaje de reconocimiento se utilizó la herramienta *sclite*; *sclite* necesita de dos archivos; uno contiene las hipótesis, es decir, los datos arrojados por *Sphinx decoder* en la etapa de reconocimiento y uno que contiene las frases originales, es decir, lo que se esperaba que reconociera. El comando utilizado se muestra a continuación:

sctk-2.1/bin/sclite -r referencia -h hipotesis -i rm -o all

En donde,

-r Archivo con los datos de referencia.

-h Archivo con las hipótesis.

-i Indica el formato de los id's de las oraciones.

-o Define los reportes de salida. Con la opción *all* arroja 3 tipos de archivo.

a) .sys – Este archivo contiene el resumen de las palabras que fueron reconocidas con éxito y de las que fueron mal reconocidas; además contiene la información del porcentaje de error por oración (columna ***S.Err***); está información esta dada en porcentajes.

SYSTEM SUMMARY PERCENTAGES by SPEAKER

hipotesis_test3.txt								
SPKR	# Snt	# Wrd	Corr	Sub	Del	Ins	Err	S.Err
	499	5307	98.7	1.0	0.3	0.8	2.1	11.4
Sum/Avg	499	5307	98.7	1.0	0.3	0.8	2.1	11.4
Mean	499.0	5307.0	98.7	1.0	0.3	0.8	2.1	11.4
S.D.	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Median	499.0	5307.0	98.7	1.0	0.3	0.8	2.1	11.4

b) .raw – Este archivo contiene la misma información que el ***.sys*** pero el resumen esta dado en cantidad, es decir, cuentas palabras están correctas, cuantas fueron sustituidas, insertadas y eliminadas.

SYSTEM SUMMARY PERCENTAGES by SPEAKER

hipotesis_test3.txt								
SPKR	# Snt	# Wrd	Corr	Sub	Del	Ins	Err	S.Err
	499	5307	5239	53	15	41	109	57
Sum	499	5307	5239	53	15	41	109	57
Mean	499.0	5307.0	5239.0	53.0	15.0	41.0	109.0	57.0
S.D.	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Median	499.0	5307.0	5239.0	53.0	15.0	41.0	109.0	57.0

c) .pra – Este archivo contiene la información de cómo fue alineado el archivo de las hipótesis con el de referencia.

```
                DUMP OF SYSTEM ALIGNMENT STRUCTURE

System name:   hipotesis_test3.txt

Speakers:
    0:

Speaker sentences   0:     #utts: 499
id: (s07101)
Scores: (#C #S #D #I) 8 0 0 0
REF:  el ciclo comienza el primer domingo de adviento
HYP:  el ciclo comienza el primer domingo de adviento
Eval:

id: (s07102)
Scores: (#C #S #D #I) 7 0 0 0
REF:  calidad del agua tratada en la planta
HYP:  calidad del agua tratada en la planta
Eval:

id: (s07103)
Scores: (#C #S #D #I) 6 0 0 0
REF:  incluido el cambio la conduccion tecnica
HYP:  incluido el cambio la conduccion tecnica
Eval:
```

Bibliografía

[Manning, 99] Christopher D. Manning; Hinrich Schèutze, 1999, *Foundations of Statistical Natural Language Processing*, MIT Press.

[Jurafsky&Martin, 00] Daniel Jurafsky; James H. Martin, 2000, *Speech and Language Processing: An Introduction to Natural Language Processing, Computacional Linguistic, and Speech Recognition,* Prentice Hall, New Jersey.

[Jelinek, 98] F. Jelinek, 1998, *Statistical Methods for Speech Recognition*, MIT Press.

[Pineda *et al*., 04] Luis A. Pineda *et al*., 2004, *DIMEx100: A New Phonetic and Speech Corpus for Mexican Spanish*, Iberamia.

[Cuétara, 04] Cuétara Priede, Javier, 2004, *Fonética de la ciudad de México. Aportaciones desde las tecnologías del habla, tesis de maestría inédita*, México: UNAM.

[Rabiner&Juang, 04] Lawrence R. Rabiner; B. H. Juang, 2004, *Statistical Methods for the recognition and understanding of speech*, fecha de consulta: 01 de Agosto de 2007. Disponible en web:
http://www.caip.rutgers.edu/~lrr/lrr%20papers/354_Statistical%20Methods%20for%20ASR-final-1.pdf

[Faundez, 00] Marcos Faundez Zanuy, 2000, *Tratamiento digital de voz e imagen y aplicación a la multimedia*, Marcombo, Barcelona.

[Villaseñor *et al*., 02] Luis Villaseñor *et al*., 2002, *Comparación léxica de corpus para generación de modelos de lenguaje*, Puebla: INAOE.

[Villaseñor *et al*., 03] Luis Villaseñor *et al*., 2003, *A corpora balancing method for language model construction,* Puebla: INAOE.

[Colás, 01] José Colás Pasamontes, 2001, *Estrategias de incorporación de conocimiento sintáctico y semántico en sistemas de comprensión de habla continua en español*, fecha de consulta: 18 de Agosto de 2007. Disponible en web:
http://elies.rediris.es/elies12/cap241a.htm

[Maldonado, 98] José Luciano Maldonado, 1998, *La estadística como herramienta para el desarrollo de sistemas automáticos reconocedores de habla*, fecha de consulta: 25 de Julio de 2007. Disponible en web:
http://iies.faces.ula.ve/Revista/Articulos/Revista_14/Pdf/Rev14Maldonado.pdf

[Pava, 05] Roberto Pava Diaz, 2005, *Biología Computacional,* fecha de consulta: 30 de Septiembre de 2007. Disponible en web:
http://www.virtual.unal.edu.co/cursos/ingenieria/2001832/lecciones/hmm2.html