Entrenamiento y Evaluación de reconocedores de Voz de Propósito General basados en Redes Neuronales *feed-forward* y Modelos Ocultos de Markov

Eduardo Clemente, Alcira Vargas, Alejandra Olivier, Ingrid Kirschning, Ofelia Cervantes

TLATOA-CENTIA, Depto. Ingeniería en Sistemas Computacionales
Universidad de las Américas-Puebla
Sta. Catarina Mártir, Cholula, Puebla, 72820, México
{is101670, alcira, olivier, ingrid, ocervan}@mail.udlap.mx

Abstract. Este artículo presenta los resultados de un trabajo de tesis que consistió en el desarrollo y comparación de dos nuevos reconocedores de propósito general para el Español Mexicano. Uno de estos reconocedores está basado en la metodología de redes neuronales y el otro en modelos ocultos de Markov. El interés en la comparación de estas metodologías se debe a que sólo se habían comparado reconocedores de propósito específico (dígitos) [2, 8], de ahí que el desarrollo de un reconocedor de propósito general permitirá realizar cualquier tipo de aplicación para el Español hablado en México.

1 Introducción

Los métodos de reconocimiento automático de voz han sido investigados por muchos años, enfocándose principalmente en aplicaciones dedicadas a proporcionar información por teléfono. Hoy en día se ha intensificado dicha investigación y los logros que se han obtenido aún son muy limitados en su uso. Por esta razón se continuan estudiando métodos que aumenten la capacidad de reconocimiento en los sistemas para que sean robustos y tengan una mejor adaptación al medio. A pesar de que siempre se obtendrá un mejor desempeño con reconocedores entrenados para un contexto limitado, es necesario contar también con herramientas de propósito general, dónde un reconocedor de propósito general es aquél que es capaz de clasificar cualquier conjunto de unidades del habla.

El presente trabajo compara dos reconocedores entrenados con diferente metodología pero con los mismos datos. El primer reconocedor fue entrenado como una red neuronal artifical de tipo feed-forward de tres capas (entrada, intermedia y de salida). El segundo fue un modelo oculto de Markov, ambos entrenados con los scripts del CSLU-Tookit del Oregon Graduate Institute. Al final se comparan su desempeño sobre un corpus de propósito específico (dígitos), así como su tiempo de entrenamiento y memoria requerida.

2 Entrenamiento de los Reconocedores

Desde los 90's, el CSLU (*Center for Spoken Language Understanding*) ha estado trabajando en el desarrollo de nuevas herramientas para crear sistemas de lenguaje hablado. El resultado de tal esfuerzo es el CSLU Toolkit, un conjunto integrado de programas y documentación que representa el estado del arte en herramientas de investigación, desarrollo y aprendizaje acerca de sistemas del lenguaje hablado [3].

El Toolkit es un ambiente fácil de entender que integra un conjunto de tecnologías incluyendo reconocimiento de voz, síntesis de voz, y animación facial. Esto también permite el desarrollo rápido y fácil de aplicaciones de sistemas de información utilizados a través del teléfono. La arquitectura del Toolkit tiene tres componentes principales: un conjunto de librerías que contienen módulos de tecnologías de punta, además de los antes mencionados, también cuenta con un shell interactivo de programación y un ambiente de desarrollo rápido de aplicaciones (RAD).

2.1 Pasos para el entrenamiento de redes neuronales con el CSLU-Toolkit

Esta herramienta contiene varias funciones en C, Tcl y Tk que implementan los pasos generales que se necesitan para crear un reconocedor basado en redes neuronales. Los reconocedores son desarrollados usando un enfoque basado en frames con una red neuronal para estimar las probabilidades posteriores. Los pasos desarrollados durante el reconocimiento son:

- La señal es dividida en frames.
- Las características son calculadas para cada frame en una ventana de contexto. Esta ventana de contexto describe el espectro de la voz en el frame central y en un número pequeño de los frames que lo rodean.
- Las características en cada frame son clasificadas en categorías basadas en fonemas usando una red neuronal. Las salidas de la red neuronal son usadas como estimadores de probabilidad, para cada categoría fonética, del frame actual que contiene esa categoría.
- La matriz de probabilidades y un conjunto de modelos de pronunciación es usada por la búsqueda de Viterbi para determinar las palabras que son más probables.

2.1.1 Entrenamiento y generación de los datos

Una vez que las categorías a entrenar han sido encontradas, y el número de ejemplos por categoría ha sido determinado, los datos actuales serán entrenados y guardados en un archivo de vectores, el cual contendrá para cada ejemplo a entrenar, las características que serán la entrada en la red neuronal y la categoría objetivo.

En el CSLU Toolkit se usan redes feed-forward de 3 capas. El número de nodos de entrada es el número de características espectrales, y el número de nodos de salida es el número de categorías a ser entrenadas.

Cuando se utilizan muchos ejemplos por categoría, es inevitable que algunas de las categorías tengan menos ejemplos que otras, haciendo dificil de aprender dichas categorías. Esta dificultad se presenta debido al factor de que hay mucho más

ejemplos negativos[1] que ejemplos positivos para categorías esparcidas [4]. El número de iteraciones en el entrenamiento usualmente es de 20 a 30 iteraciones para alcanzar el mejor desempeño.

El método de *forced alignment* puede ser usado para generar etiquetas para el entrenamiento. Para generar etiquetas iniciales usando dicho método, se usa un reconocedor de propósito general. También podemos usarlo para re-entrenar una red, en este caso usamos la mejor, la cual obtiene los resultados más altos.

Un método final para mejorar los resultados es usar el entrenamiento "*forward-backward*" o "*embedded*". En el entrenamiento con forward-backward, los destinos de la red neuronal no son valores binarios. Sino probabilidades posteriores. Estas probabilidades son determinadas usando el algoritmo de forward-backward, en el cual una red neuronal previamente entrenada es usada para calcular las probabilidades de observación.

2.1.2 Entrenamiento

Los pasos generales para el procedimiento de entrenamiento son:

- Crear un archivo llamado "corpora", el cual contiene una lista maestra de cada corpus y la localización y formato de los archivos en el corpus.
- Crear un archivo de "información" para el entrenamiento, desarrollo y prueba. Este archivo contendrá toda la información que es necesaria para lo antes mencionado, además de tener el nombre base del reconocedor, el número mínimo de ejemplos requeridos por cada categoría, e información dependiente del corpus.
- Encontrar los datos para el entrenamiento.
- Entrenar y de ser necesario,
- Se puede hacer un re-entrenamiento hasta alcanzar el mejor desempeño de todas las redes, aunque algunas veces la red que resulta de la primera vuelta tiene el mejor desempeño.

Los datos en la fase de entrenamiento, desarrollo (pruebas para ajustar el entrenamiento) y prueba (para obtener la iteración que genera los mejores resultados) de cada reconocedor fueron obtenidos del corpus de propósito general (dominio general). Para el entrenamiento se asignaron 300 locutores, para la partición de desarrollo se usaron 100 locutores y para prueba se usaron todos los locutores del corpus dígitos. De los datos de entrenamiento se toman las muestras para que la red aprenda y es importante que sean suficientes, para asegurar de este modo un mejor modelado y reconocimiento. Los datos de desarrollo se utilizan para escoger la mejor iteración, es importante señalar que éstos datos sean diferentes a los de entrenamiento.

2.1.3 Evaluación de la red

Una vez que se ha entrenado es necesario determinar cual de las iteraciones es la que tiene el mejor desempeño en el conjunto de prueba. Para ello se busca reconocer cada pronunciación en el conjunto de datos de desarrollo usando los pesos de la red de cada iteración. Si el número de palabras en cada pronunciación no es conocida antes,

---

[1] Los ejemplos negativos son ejemplos para los cuales la categoría que está siendo entrenada tiene como valor objetivo 0 y 1 para los ejemplos positivos.

entonces se evalua el desempeño en cada iteración en términos de sustitución, inserción y borrado de errores. Si el número de palabras es conocido con anticipación, entonces sólo se miden los errores de sustitución, con el mismo método. La exactitud de la red siempre se mide como un *100% - (subs+ins+del)*, donde *subs* son los errores de sustitución, *ins* son los errores de inserción y *del* son los errores de omisión, todos ellos en porcentajes. Procedemos a escoger la mejor red con el nivel de exactitud en la palabra. En caso de valores iguales, se selecciona la iteración con el mayor nivel de exactitud en la oración.

2.2 Pasos para el entrenamiento de Modelos Ocultos de Markov con el CSLU-Toolkit (CSLU-HMM)

El ambiente de desarrollo del CSLU-HMM es una colección de bloques construidos con el ánimo de proveer al usuario un ambiente de desarrollo e investigación fácil de usar, además de poderoso para la construcción del estado de arte de los HMM, redes neuronales y reconocedores híbridos.

2.2.1 Arquitectura del CSLU-HMM

El CSLU-HMM soporta un ambiente flexible para varias estrategias de modelado. Se debe tener mucho cuidado para diseñar todos estos componentes para poder operarlo eficientemente y de manera consistente, y se le debe dar una atención especial a su modularidad, portabilidad y extensibilidad.

En la figura 1 se presenta la arquitectura del software del CSLU-HMM. Esta es un extensión del shell de CSLU, también usa una gran variedad de módulos pre-existentes para la computación distribuida, procesamiento de señales de voz, operaciones matemáticas y varios módulos que proveén un completo ambiente de desarrollo para los HMM.

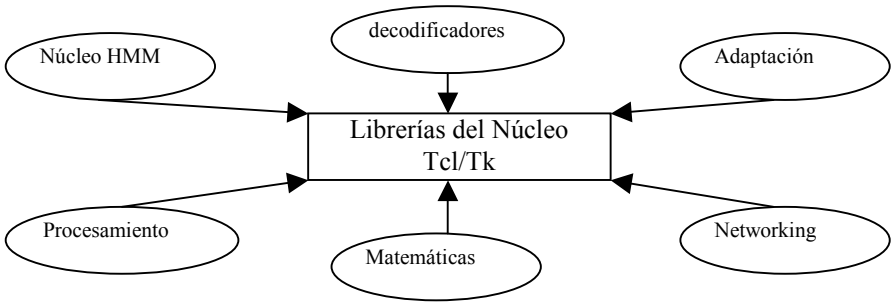

Fig. 1: Arquitectura del del CSLU-HMM

2.2.2 El núcleo del CSLU-HMM

La versión actual del software soporta el modo de entrenamiento avanzado y estándar. Para un entrenamiento básico del modelo, las técnicas son las siguientes:

- Modelo de inicialización usando VQ.

- Modelo entrenamiento basado en el algoritmo EM (expectation / maximization).

El ambiente del CSLU-HMM incluye decodificadores de Viterbi para los modelos estándar de HMM, así como para reconocedores híbridos basados en redes neuronales. Ambas arquitecturas soportan el reconocimiento de gramáticas de estado finito. Los scripts del CSLU-HMM proveén un ambiente de desarrollo y usa una interfaz tradicional de lineas de comando. Cabe aclarar que dichos scripts están en Tcl, lo cual nos permite acondicionarlos a nuestras necesidades.

2.2.3 Preparación de los datos

Esta es una de las partes más importantes para construir un buen reconocedor. Para esto se necesitan archivos de voz asociados con transcripciones de palabra o fonemas, entonces un MLF(Master Label File) se genera, el cual contendrá la misma información pero en un solo archivo. Cada uno de los siguientes pasos son muy específicos y los datos deben estar en un formato ya establecido. Ya que tenemos el formato necesitamos saber la arquitectura de nuestro reconocedor. La arquitectura define las unidades básicas usadas durante el reconocimiento. Dependiendo de la aplicación estas unidades pueden ser palabras, sílabas, fonemas o subfonemas. Todo esto es posible usando el lenguaje de configuración del CSLU-HMM. Un ejecutable del CSLU-HMM, hscript lee la configuración de dicho modelo y genera un conjunto de modelos base. Estos modelos definirán la arquitectura y los valores de los parámetros iniciales para las probabilidades de transición de los estados [5, 6].

2.2.4 Entrenamiento del modelo

Durante la inicialización del modelo el parámetro del estado estima una colocación discreta de datos para cada estado en el HMM. Sin embargo esta colocación a menudo no es la óptima. Los segmentos de las características que están cerca de las transiciones de los estados son difíciles de asociar con un solo estado. Es decir hay una leve asociación con los estados. El algoritmo de forward/backward en contraste con el realineamiento de estados de Viterbi calcula esta asociación probabilística de las característcias contra los estados del HMM. Estas asociaciones probabilísticas son usadas con el algoritmo EM mencionado anteriormente para además mejorar los parámetros iniciales estimados. Para cada segmento el acumulador de los parámetros de las variables son actualizados. Este acumulador es usado para guardar la contribución de cada vector de datos hacia cada mezcla de cada estado.

Juntos con el modelo de incialización estas dos técnicas de entrenamiento son usadas para formar un modelo inicial el cual es usado como base para el entrenamiento del reconocedor.

2.2.5 Transcripción automática/forced alignment

Los datos etiquetados a mano son suficientes para crear un modelo inicial para reconocedores basados en fonemas (el cual no es nuestro caso). Para construir reconocedores con más exactitud y más robustos se requieren de más datos. Muchos de los corpus contienen transcripciones a nivel de palabra, lo cual puede ser usado para aumentar los datos entrenados ya existentes.

Las transcripciones de entrada de las palabras son usadas para crear una gramática de estados finita donde cada nodo o estado en la gramática contiene una

palabra y sus variantes de pronunciación son obtenidas de un léxico de pronunciaciones. Entonces el algoritmo estándar de Viterbi es usado para encontrar la mejor ruta posible a través de la gramática. Para cada palabra en la entrada de transcripciones las variantes de pronunciación son leidas de una base de datos de pronunciaciones.

La inicialización y el entrenamiento del modelo usa datos asociados con un modelo particular. Durante estos pasos de entrenamiento se asume que los límites fonéticos son definidos y que no hay interacciones entre los modelos vecinos. La re-estimación de parámetros direcciona estos problemas creando un modelo compuesto desde las transcripciones asociadas con este.

2.2.4 Evaluación

Para establecer el proceso de evaluación primero necesitamos construir una gramática de tareas y una búsqueda asociada con la red usada por el decodificador de *Viterbi*. Ya que tenemos la red de búsqueda, el reconocimiento es mejorado usando la herramienta hmmsearch.tcl la cual soporta modelados completos de trifonemas dependientes del contexto, con palabras, fonemas, o alineamientos de nivel de estados. Por omisión sólo la primera mejor respuesta se regresa. Sin embargo, el lattice[2] de palabra es regresado y también está disponible conteniendo múltiples hipótesis.

Cuando todos los archivos han sido procesados, las transcripciones de salida son comparadas con las respuestas de reconocimiento para evaluar el desempeño del reconocedor.

3 Experimentos y Resultados

En esta sección se presenta el desarrollo y resultados de seis experimentos realizados [14]; tres usando la metodología de redes neuronales y tres con modelos ocultos de Markov. De cada metodología tenemos un experimento independiente del contexto y dos dependientes del contexto. Para realizar una comparación de los diferentes reconocedores desarrollados se tomaron características como tiempo de entrenamiento, desarrollo y pruebas, cantidad de memoria y velocidad de proceso para cada red, así como errores de inserción, omisión y sustitución. Así mismo se presenta la descripción de los corpus utilizados, la metodología de evaluación y finalmente las tablas de desempeño para cada experimento.

3.1 El Corpus

Para construir los reconocedores se utilizaron dos corpus, uno de propósito general y otro corpus de propósito específico (*dígitos)* para realizar pruebas a nivel de palabra. El corpus de propósito general consiste de voz grabada por teléfono por más de 500 locutores. Se diseñó para cubrir vocabularios comunes como dígitos, números

[2] Búsqueda basada en refinamiento sucesivo del modelo del lenguaje. Indica además cuales palabras fueron reconocidas para ser habladas en esos intervalos .

naturales, respuestas si/no, horas, días, fechas, etc. Las condiciones de grabación incluyen formato RIFF y una frecuencia de muestreo de 8 KHz. La base de datos de voz consta aproximadamente de 11.49 hrs de grabación continua y ocupan un total de 743 MB en disco. El corpus *dígitos* fue grabado por 50 locutores y fue grabado por micrófono a 8 KHz. Las series contienen los números del 0-9 más el número 10.

3.2 Diseño de Experimentos

En el proceso de producción del habla cada fonema se ve afectado por su contexto. Para modelarlo es necesario dividir los fonemas en partes, y así modelar el dinamismo de los fonemas:

- Una parte, el fonema es independiente del contexto,
- Dos partes, la primera mitad depende del contexto izquierdo y la segunda del derecho.
- Tres partes, el primer tercio del fonema es dependiente del contexto izq., la parte central es independiente del contexto y la última depende del contexto derecho.

3.2.1 Evaluación de resultados

En los experimentos con redes neuronales se realizan un total de 30 iteraciones y se evalúan las últimas 15 redes escogiéndose solamente la que obtiene el mejor desempeño en el reconocimiento sobre los datos de la fase de desarrollo. En los experimentos de Modelos Ocultos de Markov la fase de entrenamiento consta de 10 iteraciones y se ocupa Vector Quantization y la realineación de Viterbi. De los 10 modelos se escoge el que obtiene el mejor reconocimiento.

Finalmente, para la etapa de evaluación final, se toma la red o el modelo con mayor desempeño resultante de la etapa de desarrollo, probándolos con datos no usados en ninguna de las etapas anteriores.

3.2.2 Evaluación NIST

Los reconocedores fueron evaluados con el software de NIST (National Institute of Standards and Technology). Este software tiene dos propósitos: Primero, alentar a los investigadores a usar medidas estadísticas para resumir sus mejoras y conclusiones; y segundo, proveer una manera estándar para medir el porcentaje de reconocimiento, asegurando de esta forma que las diferencias en resultados publicados por diferentes grupos de investigación son debido al desempeño de sus reconocedores y no a sus algoritmos para calcular los resultados [1]. El software de NIST para evaluar un reconocedor, primero necesita ejecutar un algoritmo de alineación de cadenas para minimizar el número de diferencias (sustituciones, inserciones y eliminaciones) entre los resultados del reconocedor y lo correcto.

A continuación se dará la descripción y características de cada uno de los reconocedores y los resultados obtenidos en los diferentes experimentos que se diseñaron.

3.3 Resultados de los Experimentos

3.3.1 Experimento I

El primer experimento está basado en modelos acústicos independientes del contexto (una parte). Los experimentos independientes del contexto proveen un experimento base sobre el cual se pueden realizar mejoras. Los resultados se ven en la Tabla 1. Los resultados que se muestran se obtuvieron al evaluar los reconocedores sobre 1883 frases de 49 locutores.

Tabla 1: Resultados Experimento I

	Exactitud	*Palabras correctas*	*Error inserción*	*Error borrado*	*Error sustitución*
NN2	*98.2%*	*98.2%*	*5.5%*	*0%*	*1.4%*
HMM2	*71.19%*	*75.5%*	*4.3%*	*3.72%*	*20.78%*

3.3.2 Experimento II

Este segundo experimento se realizó dividiendo los fonemas en 3 y 1 partes.

Tabla 2: Resultados Experimento II

	Exactitud	*Palabras correctas*	*Error inserción*	*Error borrado*	*Error sustitución*
NN1	*98%*	*98%*	*0*	*0*	*1.8*
HMM	*92.14%*	*93.29%*	*1.10*	*2.10*	*4.59*

3.3.3 Experimento III

Este experimento se realizó tomando sólo la mitad de los datos que se ocuparon en el primer experimento. Los resultados del tercer experimento se observan en la Tabla 3.

Tabla 3: Resultados Experimento III

	Exactitud	*Palabras correctas*	*Error inserción*	*Error Borrado*	*Error sustitución*
NN2	*99.1%*	*99.1%*	*2.4*	*0*	*.9*
HMM2	*93.0%*	*93.97%*	*.9*	*1.9*	*4.04*

3.4 Uso de Recursos

En la Tabla 4 se muestra los recursos que se utilizaron respecto a la memoria de cada uno de los experimentos. En general los reconocedores desarrollados utilizando redes neuronales ocuparon el doble de memoria.

Tabla 4: Memoria

Experimento	**Memoria (MB)**

NN I	700
NN II	450
NN III	400
HMM I	300
HMM II	224
HMM III	200

3.4.2 Requerimientos de Tiempo

El tiempo que se muestra en la Tabla 5 es aproximado ya que algunos procesos varían de acuerdo al número de datos que se estén manejando y a la complejidad computacional en cada uno de los pasos de desarrollo de los reconocedores. Todos los experimentos se realizaron en una máquina con 128 MB en ram con procesador Pentium III y 450 MHz. Al realizar una prueba en una máquina con 44 MB en RAM, el tiempo de entrenamiento de los modelos de redes neuronales sobrepasaba las 34 hrs.

Tabla 5: Tiempos para las etapas de entranamiento (train), desarrollo (dev) y prueba (test). (Nota: esta etapa de pruebautiliza todos los elementos del corpus de prueba)

Reconocedor	Entrenamiento	Desarrollo	Prueba
NN I	4 hrs.	30 min.	1 hr.
NN II	3.5 hrs.	30 min.	1 hr.
NN III	2 hrs.	20 min.	45 min.
HMM I	2 hrs.	50 min.	40 min.
HMM II	1.5 hrs.	40 min.	30 min.
HMM III	1.25 hrs.	35 min.	25 min.

Conclusiones

Dados los resultados en cada uno de los experimentos de redes neuronales y de modelos ocultos de Markov, podemos decir que hay dos características en las que nos podemos enfocar: la primera es que se puede tomar como un parámetro importante el tamaño del corpus y en segundo lugar un dominio específico para aplicaciones reales.

Los resultados de la red neuronal fueron satisfactorios para corpus de tamaño pequeño como el de dígitos, pero cuando se hicieron los experimentos con el corpus de teléfono, que es considerado de mediano tamaño, el nivel de reconocimiento bajó. Cabe señalar que el corpus de propósito general fue complementado con otro más pequeño para aumentar el número de ejemplos de cada unidad fonética, sin embargo dicho corpus planteaba varios problemas de etiquetado. Sólo se etiquetó con la herramienta *forced-alignment* y no se hizo un reajuste manual que siempre es necesario después de este proceso para tener mayor exactitud en los límites fonéticos. Debido a ello, se afectó negativamente el desempeño final, pero no de manera significativa, como se puede ver en las tablas de resultados.

Para modelos ocultos de Markov el nivel de reconocimiento fue mejorando con respecto a las evaluaciones hechas con el reconocedor basado en redes neuronales.

Debido a la naturaleza de los HMM que estan está basados en la probabilidad de reconocimiento entre un estado y otro se pueden modelar mejor los efectos coarticulatorios especificando el número de contextos que se quieren manejar para cada unidad (derecho, izquierdo o central). Esto nos da un mayor manejo de cada una de las unidades a reconocer sin tener tantas restricciones que pudieran alterar el reconocimiento que está basado en cada uno de los estados del modelo en general. También el CSLU-HMM permite hacer modelados sujetos a las especificaciones que nosotros propongamos y permitiendo configurar modelos que den una visión real del problema que se abstrae.

En trabajos anteriores donde fueron desarrollados reconocedores basados en HMM se demostró que el nivel de reconocimiento era mayor al de un reconocedor basado en redes [2, 7] utilizando un corpus pequeño de prueba de propósito específico como el de *dígitos*. El objetivo de esta tesis fue el de desarrollar reconocedores con ambos enfoques e incluir un corpus más grande a los que se habían utilizado en proyectos anteriores, además de proponer un esquema de reconocimiento de propósito general. Sin embargo es necesario hacer más pruebas de evaluación con corpus grandes y diferentes al utilizado para el desarrollo, esto con el efecto de tener un mejor conjunto de evaluación.

Bibliografía

1. Castillo, O.: Evaluación de un reconocedor fonético para el español hablado en México. Tesis UDLAP, 1999.
2. Espinosa, M.: Comparación entre un sistema de reconocimiento de voz con el enfoque de redes neuronales y un sistema basado en modelos ocultos de Markov utilizando el CSLU Toolkit. Tesis UDLAP, México, 1998.
3. Fanty, M.: Overview of the CSLU Toolkit. CSLU. OGI.Portland, Oregon U.S.A,1996.
4. Hosom, J., Cole, R., Fanty, M., Schalkwyk, J., Yan, Y., Wei, W.: Training Neural Networks for Speech Recognition. CSLU, OGI, Abril 8, 1998.
5. Jelinek, F.: Statistical Methods for Speech Recognition. The MIT Press, Cambridge Massachusetts, U.S.A. 1998.
6. Schalkwyk, J., Hosom, P., Kaiser, E., Shobaki, K.: CSLU-HMM: The CSLU Hidden Markov Modeling Environment. CSLU, OGI, Marzo 14, 2000.
7. Uraga, E.: Modelado Fonético para un Sistema de Reconocimiento de Voz Continua en Español. Tesis de Maestría, ITESM Campus Morelos, Mayo, 1999.
8. Vargas, A., Munive, N.: Reconocedor Fonético de Dominio Restringido para el Español hablado en México usando el CSLU Toolkit. Tesis UDLAP, México, 1997.