UNIVERSIDAD POLITÉCNICA DE PUEBLA
Ingeniería en Informática

Proyecto de Estancia Práctica en Desarrollador en Sistemas de Software y Administrador de Redes

Documentación sobre proceso de incorporación de un nuevo idioma español al reconocedor automático de voz CMUSphinx II

Área temática del CONACYT: VII
Ingenierías y tecnologías

Presenta:
Luis Ángel Romero Toxqui

Asesor técnico
M.C. Javier Velázquez Sandoval

Asesor académico
M.C. Rebeca Rodríguez Huesca

Juan C. Bonilla, Puebla, México. 19 de diciembre del 2018

Resumen

Con el presente proyecto se plantea desarrollar un nuevo modelo acústico en español de México utilizando el software de código abierto CMUSphinx de la Universidad de Carnegie Mellon. Su principal función es la decodificación de la voz con la ayuda del modelo oculto de Márkov. Además, se utilizaron 19 personas para poder grabar su voz y utilizarla en el entrenamiento y pruebas del modelo acústico.
Los modelos del lenguaje y acústico se desarrollan para satisfacer las necesidades de un reconocedor automático de habla de múltiples hablantes o de un sistema de control por comandos.
Se obtuvo un modelo de lenguaje que consiste de uno de modelo de lenguaje. En relación con la parte acústica se entrenaron un modelo acústico. Por último, se elaboró un manual para la creación de los modelos acústicos y de lenguaje con el software CMUSphinx.

Índice

1. Introducción

Este capítulo contiene la descripción del problema del sistema CMUSphinx II, la justificación del problema y los objetivos generales y específicos.

1.1. Descripción del problema o necesidad

El objetivo de este proyecto es el entrenamiento de un modelo acústico, y un modelo de lenguaje gramatical para el idioma español de México, que pueda ser utilizado en un sistema de control por comandos; lo anterior por la falta de modelos para el idioma español, el acento, la pronunciación, el tono o simplemente al modelo de lenguaje de la variante dialéctica de esta región.

1.2 Justificación

Elaborar un documento donde se muestra el proceso de entrenamiento de CMUSphinx II, para agregar un nuevo modelo acústico en español de México, utilizando todas las herramientas que nos brindan. En él se describirán los pasos para obtener los archivos instalables, así como la configuración y los elementos de configuración particulares de este sistema. Al final se muestra su puesta en funcionamiento, ejecutando algunos ejemplos para el idioma de español México.

1.3 Objetivo General y Específicos

El objetivo general:

Elaborar un manual de Usuario y Entrenamiento en español de México del reconocedor automático de habla CMUSphinx II para incorporar al conjunto de herramientas de voz de la Universidad Politécnica de Puebla.

El objetivo Especifico:

- Investigar la documentación existente acerca de CMUSphinx II.
- Implantar el CMUSphinx II en un sistema de la Universidad Politécnica de Puebla.
- Ejecutar los ejemplos de demostración del funcionamiento correcto de CMUSphinx II.
- Traducción, Instalación, Configuración y Pruebas del sistema CMUSphinx II.
- Agregar el idioma español al sistema CMUSphinx II.
- Integrar el manual de Usuario y Entrenamiento de CMUSphinx II.

2. Metodología y herramientas

Este capítulo contiene los pasos que se llevan a cabo para la elaboración del entrenamiento de un nuevo modelo acústico español de México junto con las herramientas necesarias.

2.1. Descripción de la metodología empleada.

En este punto es importante aclarar que el RAH (Reconocimiento Automático del Habla) está desarrollado bajo el software de código abierto CMUSphinx [1] el cual es instalado y ejecutado en la computadora [2]. Los modelos del lenguaje y acústico se desarrollan para satisfacer las necesidades de un RAH de múltiples hablantes y de un sistema de control por comandos. El proceso de creación de estos sistemas se puede apreciar en la Figura 1.

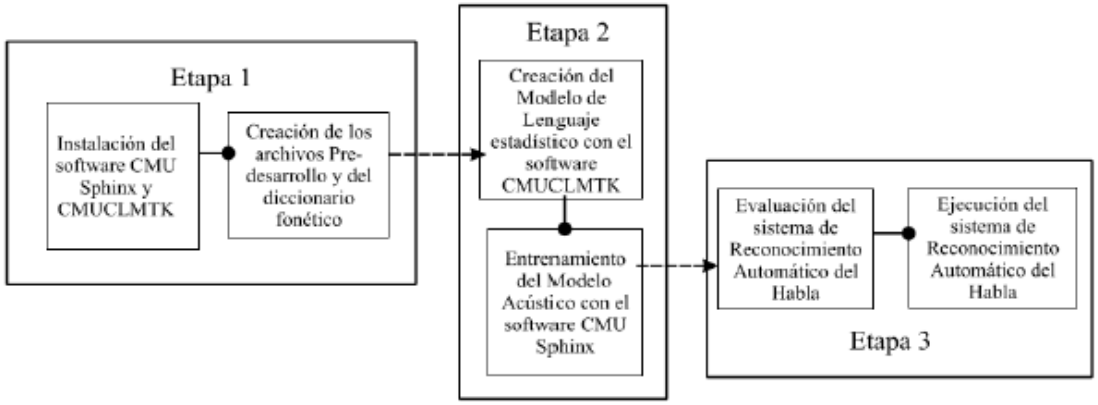

Figura 1: Proceso de desarrollo de un sistema de Reconocimiento Automático del Habla.

Etapa 1: Preparación predesarrollo.

Antes de desarrollar el modelo del lenguaje y el entrenamiento del modelo acústico, es necesario preparar los siguientes elementos:

Un diccionario fonético, de 10 palabras, que contiene los números del cero hasta el nueve, fue utilizado para el entrenamiento del modelo acústico.

Un archivo de texto plano con las frases que se emplearan como comandos del sistema y se usara para la creación del modelo del lenguaje.

Un archivo de texto plano con los fonemas utilizados y que componen cada una de las palabras encontradas en el vocabulario (.Phone), uno con las etiquetas de silencio que el sistema relacionara con los límites de los segmentos hablados de las grabaciones de audio (.Filler).

Los archivos de texto plano de transcripción y de entrenamiento que contendrán las frases grabadas en los archivos de audio por los usuarios para el entrenamiento y prueba del modelo acústico.

Los archivos de audio grabados por los usuarios necesarios para el entrenamiento del modelo acústico. Las duraciones de estos archivos deben sumar en total una hora de grabación como mínimo para que el sistema pueda realizar el entrenamiento del modelo acústico.

Es importante resaltar que los archivos de texto plano de transcripción y el archivo de texto plano que contiene las frases necesarias para la creación del modelo del lenguaje, deben estar escritos en formato Unicode UTF-8, además deben comenzar con la etiqueta < s > y finalizar con la etiqueta < /s >, las cuales deben estar incluidas como representación del silencio en el archivo de extensión Filler.

Etapa 2: Creación del modelo del lenguaje

El modelo del lenguaje es el elemento del RAH que define las probabilidades relacionadas con la aparición y el orden de las palabras en una oración. Permite junto con las características y datos obtenidos de las señales acústicas en el entrenamiento del modelo acústico, decodificar la información de las señales de habla del usuario.

En el toolkit CMUSphinx el modelo del lenguaje se puede crear de la siguiente manera. Un elemento de tipo texto y su extensión es .jsgf o .gram; tal archivo puede ser escrito de manera manual y en formato jsgf; son archivos realizados para representar de manera precisa el orden en el cual deben aparecer las palabras y son utilizados principalmente para sistemas de reconocimiento en comando y control.

Estos archivos son necesarios para crear el archivo ARPA [3] que es el resultado final de la creación del modelo del lenguaje. El archivo del modelo del lenguaje tiene extensión .lm o .arpa, además puede ser de extensión .lm.bin para ser formato binario, el cual permite ejecutar más rápido el reconocedor y es usado para modelos del lenguaje de gran tamaño.

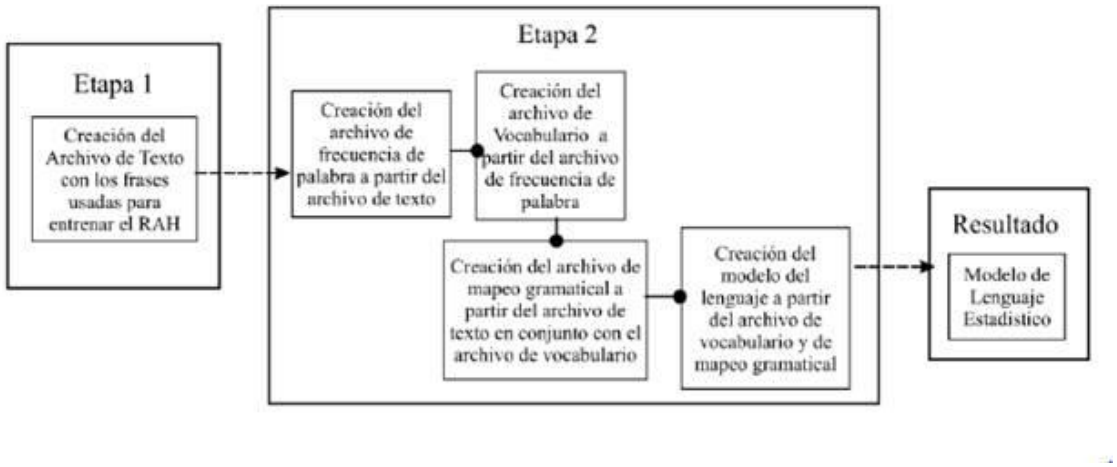

Figura 2: Proceso de creación del modelo del lenguaje estadístico utilizando el software CMUCLMTK.

El proceso de creación de un Modelo del Lenguaje Estadístico se inicia al crear en primer lugar el archivo de frecuencia de palabra. Dicho elemento permitirá crear un archivo de vocabulario que obtendrá todas las palabras que componen el texto que se utiliza para la creación del modelo del lenguaje. Con el archivo de texto en conjunto con el archivo de vocabulario, se obtiene el archivo de mapeo gramatical. Con este archivo y los archivos de vocabulario se puede realizar la creación del modelo del lenguaje necesario para el entrenamiento del modelo acústico y el funcionamiento del RAH, este proceso se muestra en la figura 2. Dicho proceso es inherente al software CMUCMLTK y todos los archivos mencionados con anterioridad en esta sección son resultado del funcionamiento del mismo, cabe resaltar que el archivo de vocabulario del cual se habla en esta sección, solo cumple una función en la creación del modelo de lenguaje y no se encuentra de ninguna manera relacionado y su contenido es diferente al del archivo de vocabulario expuesto en la sección de preparación predesarrollo.

Etapa 2: Entrenamiento del modelo acústico

El entrenamiento del modelo acústico se realiza mediante el software SphinxTrain. Para realizar el entrenamiento del modelo acústico se necesita el modelo del lenguaje, el diccionario fonético y los elementos creados en la sección de preparación predesarrollo; estos son principalmente los audios grabados por los usuarios para el entrenamiento del sistema además de los siguientes archivos:

Archivo .phone archivo que contiene la lista de fonemas presentes en el diccionario fonético.

Archivo .filler archivo que contiene la lista de etiquetas “< s >” que representan los silencios y límites de los segmentos hablados de las grabaciones de audio

Archivo _train.fileids lista de archivos de entrenamiento grabados, relacionados con textos de entrenamiento.

Archivo _train.transcription archivo que contiene la lista de textos de entrenamiento.

Archivo_test.fileids lista de archivos de entrenamiento grabados relacionados con textos de prueba.

Archivo _test.transcription archivo que contiene la lista de textos de prueba.

Para iniciar el entrenamiento solo es necesario ejecutar el comando de configuración, el cual creará una carpeta donde se almacenará el modelo acústico entrenado y dentro de esta un archivo de configuración sphinxJrain.cfg. Así, se debe configurar las características y las necesidades del modelo acústico a desarrollar. Se debe incluir dentro de la carpeta del modelo acústico los archivos de predesarrollo que se mencionaron anteriormente y una carpeta donde se almacenaran los audios, los cuales deberán estar ordenados la configuración o la relación expuesta en el Archivo _train.fileids.

En el archivo de configuración se debe modificar las rutas de los archivos de predesarrollo, de igual manera y de ser necesario, se debe modificar el tipo de archivo de modelo del lenguaje con el que se desea decodificar y realizar el entrenamiento, escoger la frecuencia a la que se encuentran grabados los audios (preferiblemente 16 kHz) o modificar el filtro aplicado a las señales de audio; la voz humana se encuentra entre los rangos de 400 Hz a 6.000 Hz, por lo cual es recomendable aplicar un filtro en los 200 Hz para eliminar cualquier posible ruido que pueda perturbar el funcionamiento del reconocedor.

El modelo acústico se entrena en tres etapas como se representa en la figura 3. En la primera parte el sistema verifica los requisitos del entrenamiento con el fin de determinar y comprobar que se encuentren en orden los archivos de predesarrollo necesarios en el entrenamiento. La segunda parte es la extracción de características en la cual se divide la señal en segmentos para realizar un procesamiento de dicha señal, extraer sus características y generar un conjunto de vectores como resultado del análisis realizado por el software SphinxTrain.

En relación con el conjunto de vectores se realiza la tercera parte del entrenamiento; en esta se crea un modelo probabilístico teniendo como referencia a los HMM (Modelo oculto de Márkov), el cual asocia a las probabilidades de las palabras obtenidas de la señal de audio entrante con las probabilidades existentes en el modelo del lenguaje para estas mismas; el resultado de esta etapa se denomina decodificación y tiene como resultado una carpeta que contiene los elementos de un modelo acústico.

El sistema realiza un test de entrenamiento en donde usa los archivos de prueba para verificar el funcionamiento y demostrar los posibles errores del modelo acústico; en la presente investigación estas pruebas son denominadas pruebas de entrenamiento y son el primer filtro de evaluación del sistema. La duración del proceso de entrenamiento puede variar dependiendo de la cantidad de audios a analizar, en promedio una hora de grabaciones tiene una duración de entrenamiento de catorce horas.

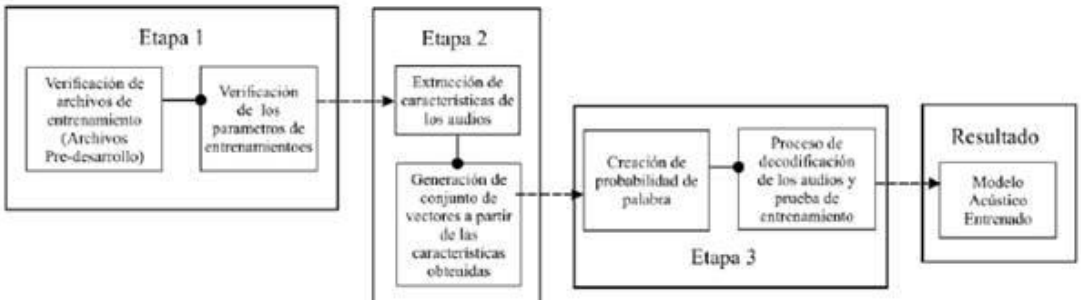

Figura 3: Proceso de entrenamiento del modelo acústico.

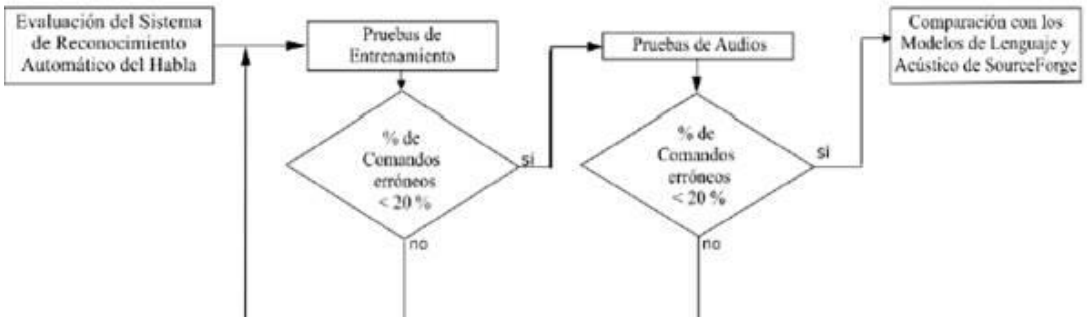

Figura 4: Proceso de evaluación del sistema de RAH.

Etapa 3: Evaluación del sistema de RAH desarrollado

La evaluación de los modelos acústico y de lenguaje se realizó tal y como se muestra en la figura 4, analizando primero los resultados de entrenamiento, los cuales son producto de la parte final del proceso de entrenamiento del modelo acústico, seguidos de pruebas del sistema de RAH con audios.

En caso de que los resultados de entrenamiento no superen el 80 % de acierto o no reconozca todos los comandos bajo los cuales operara el sistema, es necesario realizar nuevamente el entrenamiento del sistema, es decir, verificar que los audios estén grabados correctamente y de ser el caso, grabarlos nuevamente; de ser necesario, cambiar el modelo de lenguaje con el que se está decodificando el sistema o los comandos con los cual este opera; y por ultimo modificar el archivo de configuración de entrenamiento Sphinx_train.cfg para que el sistema entrenado cumpla con las características deseadas. Luego se ejecutará nuevamente el entrenamiento mediante el software SphinxTrain.

2.2. Herramientas utilizadas.

Los siguientes paquetes son necesarios para la capacitación:

- sphinxbase-5prealpha - biblioteca de soporte requerida por Pocketsphinx y Sphinxtrain.
- sphinxtrain-5prealpha - herramientas de entrenamiento de modelo acústico.
- Pocketsphinx-5prealpha - biblioteca del reconocedor escrita en C. [4]

También se requieren los siguientes paquetes externos:

- ActivePerl en Windows.
  Ventajas:
  1. Perl se puede obtener sin costo.
  2. Perl está disponible para prácticamente todos los sistemas operativos y viene incluido en las distribuciones Unix y Linux.
  3. Solo toma unas pocas horas aprender suficiente Perl para escribir tus propios programas.
  4. Los programas de Perl son más cortos y fáciles de entender que los programas escritos en C o Java.

  Desventajas:

  1. Las siguientes versiones posiblemente no tengan compatibilidad con las anteriores. [5]
- ActivePython en Windows.
  Ventajas:
  1. Soporta varias bases de datos.
  2. Es un lenguaje muy poderoso.
  3. Es un lenguaje multiplataforma.
  4. Posee un núcleo de lenguaje relativamente pequeño.
  5. Consta con el apoyo de muy buenas librerías.
  6. Fácil gestión de errores mediante las excepciones.

  Desventajas:

  1. Es un lenguaje interpretado lo que lo vuelve más lento
  2. La programación web en Python es compleja.

3. Resultados

Este capítulo contiene la implementación de cada paso que se describe en el capítulo 2. Metodologías y herramientas.

Etapa 1: Preparación predesarrollo.

Se creó un diccionario fonético que contiene diez palabras los cuales son los números del uno hasta el nueve como se muestra en la Figura 5.

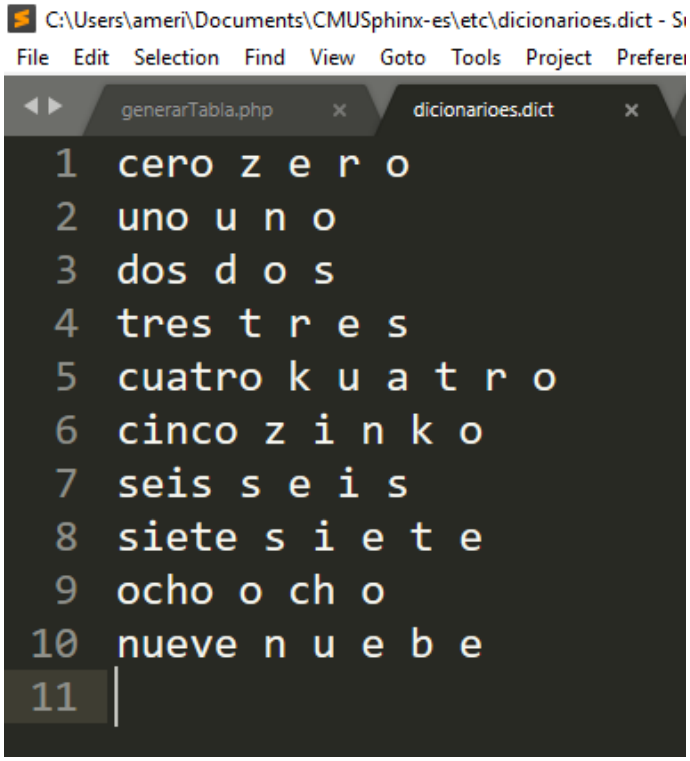

Figura 5: Diccionario fonético, archivo .dict.

Creación de un archivo de texto plano con los fonemas utilizados y que componen cada una de las palabras encontradas en el vocabulario con la extensión .phone, como se muestra en la Figura 6. Uno con las etiquetas de silencio que el sistema relacionara con los límites de los segmentos hablados de las grabaciones de audio (.Filler), como se muestra en la Figura 7.

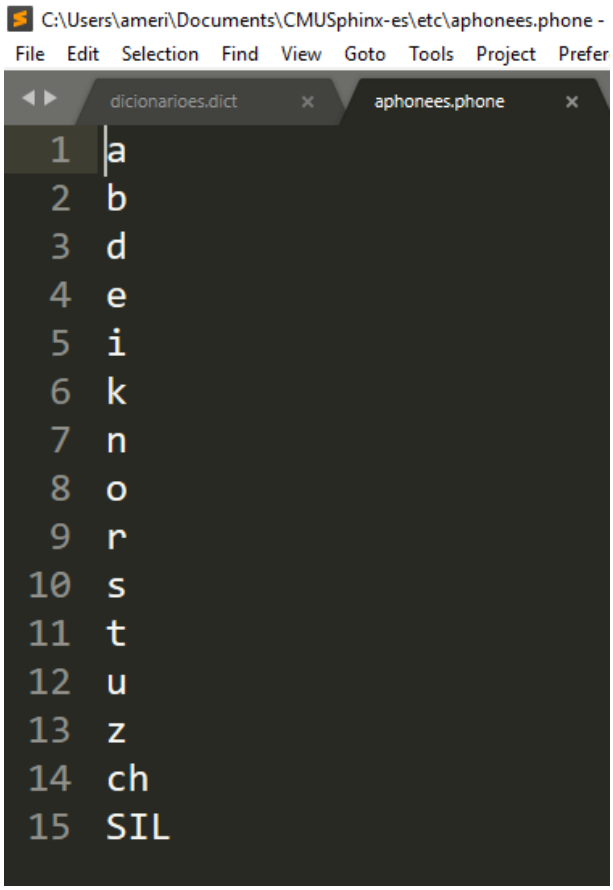

Figura 6: Archivo phone.

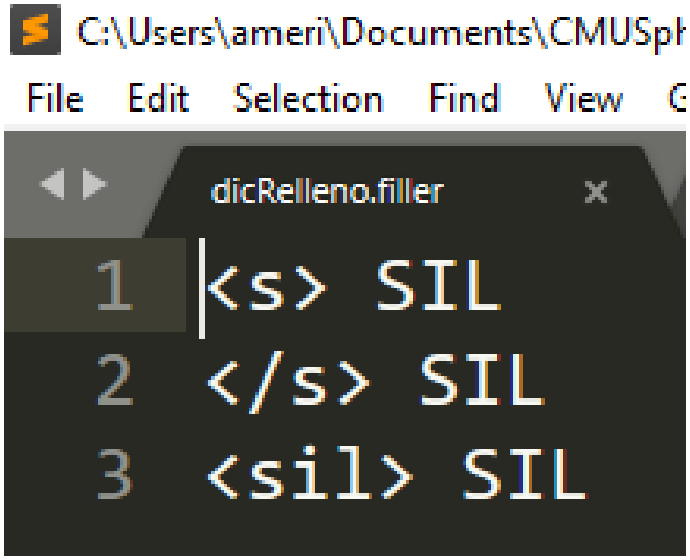

Figura 7: Archivo .filler.

Los archivos de texto plano de transcripción y de entrenamiento que contendrán las frases grabadas en los archivos de audio por los usuarios para el entrenamiento y prueba del modelo acústico, mostradas en la Figura 8 y Figura 9.

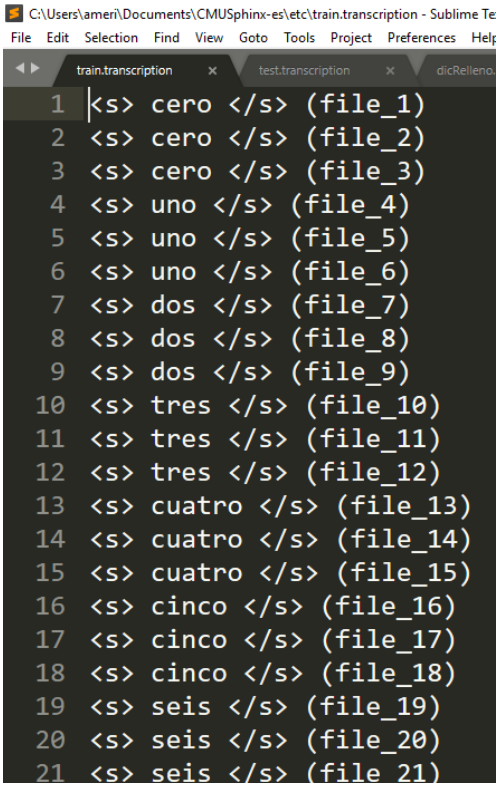

Figura 8: Archivo de entrenamiento formato .transcription.

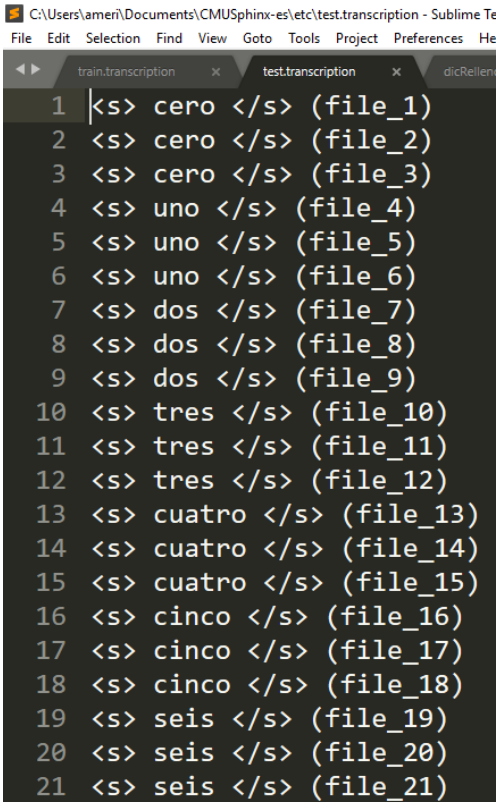

Figura 9: Archivo de prueba en formato .transcription.

Los archivos de audio grabados por los usuarios necesarios para el entrenamiento del modelo acústico mostrados en la Figura 10.

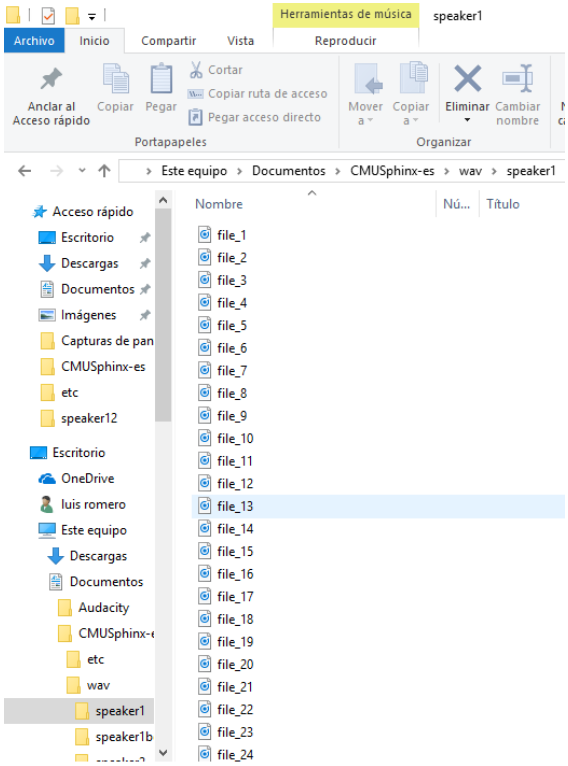

Figura 10: Audios en formato .wav

Etapa 2: Creación del modelo del lenguaje

La creación del modelo de lenguaje se realizó con el archivo modeloIdioma.jsgf, se muestra en la Figura 11.

Figura 11: Archivo de modelo de lenguaje en formato .jsgf.

Etapa 2: Entrenamiento del modelo acústico

Para iniciar el entrenamiento es necesario ejecutar el comando de configuración, como se muestra en la Figura 12, el cual creará una carpeta donde se almacenará el modelo acústico entrenado y dentro de esta un archivo de configuración sphinxtrain.cfg. Así, se debe configurar las características y las necesidades del modelo acústico a desarrollar. Se debe incluir dentro de la carpeta del modelo acústico los archivos de predesarrollo que se mencionaron anteriormente y una carpeta donde se almacenaran los audios, los cuales deberán estar ordenados la configuración o la relación expuesta en el Archivo es_train.fileids.

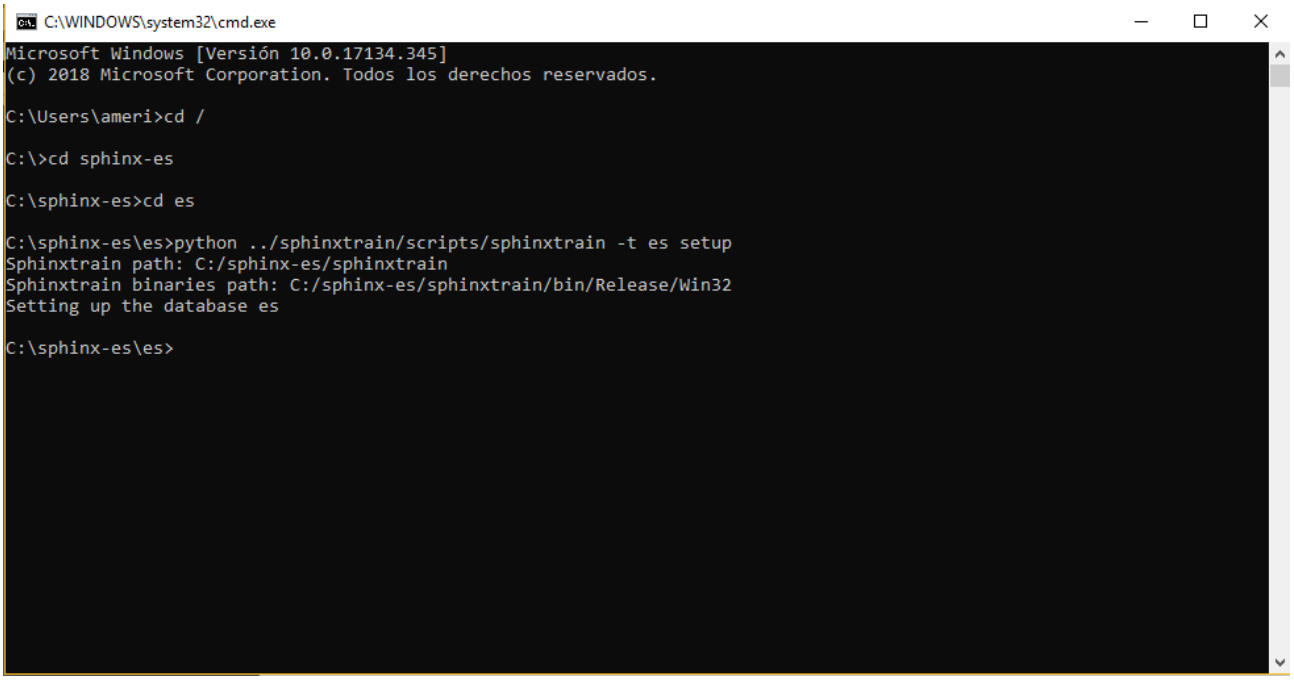

Figura 12: Ejecución del comando de configuración.

En el archivo de configuración sphinx_train.cfg se debe ver las siguientes configuraciones:

$CFG_WAVFILES_DIR = "$CFG_BASE_DIR/wav";

$CFG_WAVFILE_EXTENSION = 'sph';

$CFG_WAVFILE_TYPE = 'nist'; # one of nist, mswav, raw

Los audios de entrenamiento están en formato WAV, se cambió sph a wav, y nist a mswav, como se muestra a continuación:

$CFG_WAVFILES_DIR = "$CFG_BASE_DIR/wav";

$CFG_WAVFILE_EXTENSION = 'wav';

$CFG_WAVFILE_TYPE = 'mswav'; # one of nist, mswav, raw.

En la línea de código $CFG_FINAL_NUM_DENSITIES = 128; cambiamos el ‘128’, puede ser cualquier potencia de 2: 4,8,16,34,64. En este caso ponemos 8.

Una vez que en el archivo sphinx_train.cfg se cambiaron las características y las necesidades para el modelo acústico, se empiece a correr el entrenamiento con el siguiente comando ‘python ../Sphinxtrain/scripts/Sphinxtrain run’, como se muestra en la figura 13.

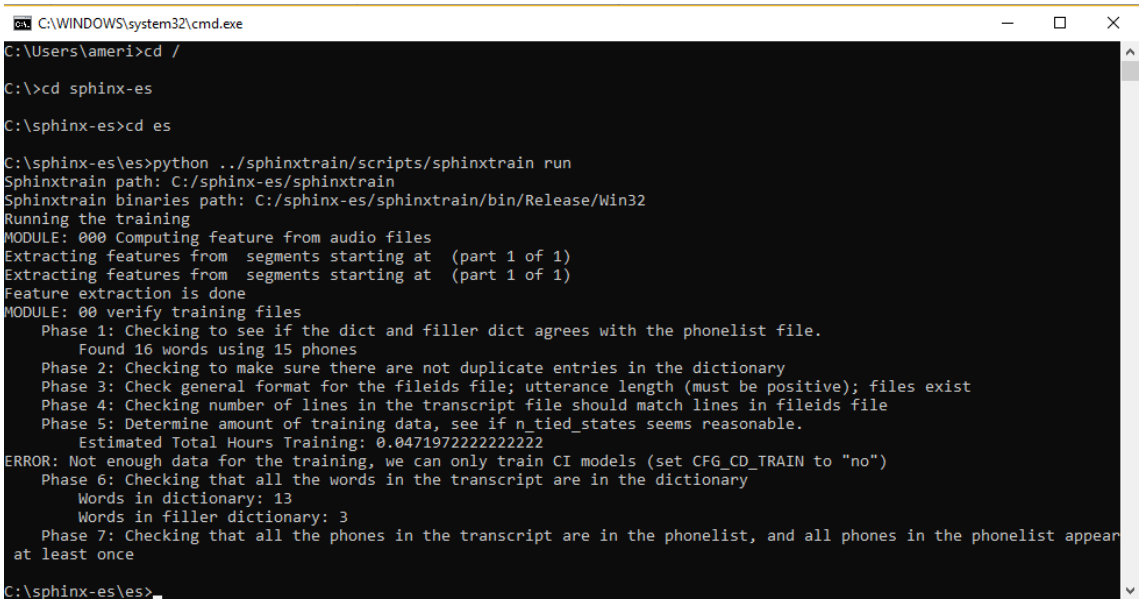

Figura 13: Ejecución del comando para correr el entrenamiento.

Etapa 3: Evaluación del sistema de RAH desarrollado.

Para la evaluación del modelo acústico primero se analiza los resultados de entrenamiento, los cuales son producto de la parte final del proceso de entrenamiento del modelo acústico. Generando un archivo donde viene cada modulo del entrenamiento con su resultado como se muestran en la figura 14.

Figura 14: Archivo con los resultados del entrenamiento.

Para la realización de las pruebas del modelo acústico se utilizo la herramienta de Sphinx4, donde se le pasa por parámetro la ubicación del modelo acústico, el diccionario y el modelo gramatical. Además, de poner la ubicación de los archivos de audio en formato WAV.

4. Conclusiones y recomendaciones

Como conclusión en el desarrollo del modelo acústico con las herramientas de CMUSphinx me permiten aprovechar las nuevas tecnologías y así como los recursos que nos ofrecen, cabe mencionar que al empezar a desarrollar el modelo acústico no tenía conocimientos sobre las herramientas de CMUSphinx por lo cual se me dificulto el desarrollo del modelo acústico, sin embargo, pedí ayuda a profesores y compañeros y así como a mi asesar, para poderme guiar, obtener conocimientos sobre las herramientas de CMUSphinx y desarrollar el modelo acústico requerido.

Se obtuvo un modelo de lenguaje que consiste del archivo de modelo gramatical (.jsgf). Además, se elaboró un manual para la creación de los modelos acústicos y de lenguaje con el software CMUSphinx.

Con la elaboración de este proyecto adquirí conocimientos sobre CMUSphinx y sus herramientas que proporciona como Pocketsphinx, Sphinxtrain, Sphinxbase y Sphinx4. Además, de lograr sobreponerme a los errores que se generaban al crear los archivos de predesarrollo y durante el entrenamiento del modelo acústico. Es posible usar el modelo acústico creado, en Sphinx4 para que logre reconocer los numero en español y posterior mente entrenar con un diccionario mas grande, para ir aumentando las palabras que pueda reconocer.

Como recomendación, es utilizar la versión 2 de Python, además, de contar con un mínimo de cinco horas de grabaciones para hacer el entrenamiento.

El número de participantes en el proceso de entrenamiento del modelo acústico influye bastante en la calidad del procesamiento del reconocedor de voz. Para obtener una mejor respuesta del reconocedor automático de voz es importante usar un diccionario largo para la etapa de entrenamiento.

4. Referencias bibliográficas

[1] P. Lamere, P. Kwok, E. B. Gouv, R. Singh, W. Walker, y P. Wolf, The CMU sphinx-4 speech recognition system, Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, Hong Kong, 2003. Disponible en: https://s3.amazonaws.com/academia.edu.documents/30790465/sphinx4.pdf?AWSAccessKeyId=AKIAIWOWYYGZ2Y53UL3A&Expires=1539135093&Signature=3x8DIQlZz28aJgiCJa0QfROh36M%3D&response-content-disposition=inline%3B%20filename%3DThe_CMU_SPHINX-4_speech_recognition_syst.pdf.

[2] M. Raab, R. Gruhn y E. Noeth, “A scalable architecture for multilingual speech recognition on embedded devices”. Speech Communication, vol. 53, no. 1, January 2011, pp. 62-74. Disponible en: https://www.sciencedirect.com/science/article/pii/S0167639310001342.

[3] L. Villasenor, M. Montes, M. Perez, D. Váufreydáz, Comparación léxica de corpus para generación de modelos de lenguaje, IBERAMIA workshop on Multilingual Information Access and Natural Language, 2002. Disponible en: http://www-prima.inrialpes.fr/Vaufreydaz/Telechargement/Villasenor02a.pdf.

[4] URL: https://cmusphinx.github.io/ Página principal de CMUSphinx, en ella se puede consultar la documentación de entrenamiento de CMUSphinx para el nuevo modelo acústico. Fecha de consulta 09/Octubre/2018.

[5] Jules J. Berman. (2011). Perl: The Programming Language. Jones & Bartlett Publishers. Disponible en: https://books.google.com.mx/books?id=maDGMelhpg8C&printsec=frontcover&hl=es#v=onepage&q&f=false

Universidad Politécnica de Puebla
Ingeniería en Informática

Luis Ángel Romero Toxqui
Javier Velázquez Sandoval
Rebeca Rodríguez Huesca