UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH

AN ASR PROTOTYPE FOR SPANISH DICTATION

A Degree Thesis

Submitted to the Faculty of the

Escola Tècnica d'Enginyeria de Telecomunicació de Barcelona

Universitat Politècnica de Catalunya

and

AGH University of technology

by

MARTA COSANO SERRA

In partial fulfilment

of the requirements for the degree in

AUDIOVISUAL SYSTEM ON TELECOMMUNICATIONS ENGINEERING

Advisor: M. ASUNCIÓN MORENO BILBAO

Advisor: BARTOSZ ZIÓLKO

Barcelona, January 2020

Abstract

Automatic Speech Recognition (ASR), or speech to text conversion, has been subject to many researchers for decades due to its various applications.

In this project I propose to implement an ASR based on Hidden Markov Model (HMM) and Deep Neuronal Network (DNN) using Kaldi toolkit.

Once a HMM baseline system has been built, I experimented using various transformations and neural networks. Then, more data has been added to the system to try to optimize it. Finally, the results of the transcripts have been analyzed to know where the model fails and propose future improvements.

Resum

El reconeixement automàtic de la parla, també denominat conversió de veu a text, ha sigut motiu de nombroses investigaciones durant dècades degut a les seves diverses utilitats.

En aquest projecte s'ha implementat un model de reconeixement automàtic de la parla en castellà basat en models ocults de Markov (MOM) i xarxes neuronals profundes (DNN) usant l'eina Kaldi.

Una vegada es construeix un sistema de referencia amb MOM, s' experimentarà usant varies transformades i xarxes neuronals. Després, s'afegirà més dades al sistema per tal d'intentar optimitzar-lo. Finalment, s'analitzarà els resultats de les transcripcions per saber en qué falla el model i poder proposar futures millores.

Resumen

El reconocimiento automático del habla, también denominado conversión de voz a texto, ha sido motivo de numerosas investigaciones durante décadas debido a sus diversas utilidades.

En este proyecto se ha implementado un modelo automático del habla en español basado en modelos ocultos de Markov (MOM) y redes neuronales profundas (DNN) usando la herramienta Kaldi.

Una vez se ha implementado un sistema de referencia con MOM, se ha experimentado usando varias transformadas y redes neuronales. Después, se ha añadido más datos en el sistema para intentar optimizarlo. Finalmente, se ha analizado los resultados de las transcripciones para saber en qué falla el modelo y poder proponer futuras mejoras.

Agradecimientos

En primer lugar quiero agradecer a la profesora Sra. Asunción Moreno por la supervisión de este proyecto, por resolverme las dudas que me han podido surgir y por la paciencia durante estos meses.

También, dar las gracias al profesor Bartosz por darme la oportunidad de trabajar en su empresa durante la estancia en Polonia y proporcionarme las herramientas para desarrollar mi trabajo. Además, me gustaría dar las gracias a los compañeros de departamento de reconocimiento de voz, en especial a Tomasz y Maciek, que me han ayudado cuando me he quedado bloqueada, me han guiado en este proyecto y también, me han hecho ser más independiente.

Por supuesto, mi más sincero agradecimiento a mis padres, que siempre han confiado ciegamente en mí y en mis posibilidades, muchas veces cuando yo no lo hacía. Sobretodo, me han aguantado en épocas de exámenes, siempre me han dado ánimos y me han intentado ayudar en lo que han podido. Igualmente, por el esfuerzo económico que han hecho durante todo este tiempo para que yo pudiera estudiar en la universidad y en Barcelona.

Me gustaría agradecer a toda mi familia. Pero en especial “als padrins” que tienen muchas ganas de que acabe la carrera para verme más y siempre se interesan por mis estudios, y a mi yaya, que valoraba mucho tener una carrera universitaria y tan orgullosa estaba de mí.

Finalmente, quiero dar las gracias a Victor por hacerme ver las cosas de diferente manera, por ver las cosas positivamente cuando yo lo veo todo negro, y sobretodo por hacerme bien.

Historial de revisiones y registro de aprobación

Revisión	Fecha	Propósito
0	5/12/2019	Creación del documento
1	16/01/2020	Revisión del documento
2	20/01/2020	Revisión del documento
3	22/01/2020	Revisión del documento
4	23/01/2020	Revisión del documento

LISTA DE DISTRIBUCIÓN DEL DOCUMENTO

Nombre	e-mail
Marta Cosano Serra	martacosanoserra@gmail.com
M. Asunción Moreno	asuncion.moreno@upc.edu
Bartosz Ziólko	bartosz.ziolko@agh.edu.pl

Escrito por:		Revisado y aprobado por :	
Fecha	4/01/2020	Fecha	24/01/2020
Nombre	Marta Cosano Serra	Nombre	M. Asunción Moreno
Posición	Autora proyecto	Posición	Supervisora del proyecto

Índice

Lista de Figuras

Lista de Tablas:

1. Introducción

El reconocimiento automático de voz (RAH) es una rama de la Inteligencia Artificial en la que se incluyen muchas disciplinas. El objetivo es permitir la comunicación entre personas y ordenadores de manera eficiente, pero también la comunicación entre seres humanos, por ejemplo, en la traducción de idiomas.

Una persona cuando escucha tiene en cuenta varios conocimientos sobre el hablante y el tema de conversación. Además, se tiene en cuenta la manera en cómo habla el hablante y la estructura gramatical para predecir palabras.

Uno de los principales problemas de los modelos de reconocimiento de voz es la información que se adquiere de diferentes fuentes, como puede ser la acústica, la fonética, la fonología, la sintáctica, la semántica o la pragmática, en presencia de ambigüedades y variabilidad de hablantes, de canales y de ruido. Pero, los avances en este campo han dado lugar a métodos para conseguir reconocedores automáticos del habla que alcanzan buenos resultados. [1] [2]

1.1. OBJETIVOS

El principal propósito de este proyecto es aprender y desarrollar un sistema de reconocimiento del habla en español utilizando la herramienta de Kaldi. Kaldi es una herramienta de código abierto para reconocimiento de voz escrita en C++, Bash, Perl y Python.

Para poder realizarlo se han fijado una serie de objetivos:

1. Implementar un “baseline system” RAH usando los scripts de ejemplo de Kaldi para la base de datos Fisher Callhome Spanish.
    a. Construir un sistema basado en HMM-GMM usando diferentes transformadas y otro basado en TDNN híbridas para poder compararlos.
    b. Comparar con los resultados de los scripts de ejemplo que tiene Kaldi
2. Mejorar los resultados añadiendo más datos.
3. Evaluar los resultados de las transcripciones.

1.2. REQUISITOS Y ESPECIFICACIONES

Los requerimientos del proyecto son los siguientes:

- Implementar un sistema base de reconocimiento de voz.
- Comparar varios tipos de modelos de RAH.
- Intentar mejorar los resultados.

La especificaciones son:

- Usar la base de datos Fisher and Callhome de LCD y la base de datos en español de Common Voice.
- Implementar el sistema base gracias a la herramienta de software Kaldi.
- Crear un diccionario fonético usando Montreal Forced Alignment Grapheme to Phoneme.
- Implementar varios modelos aplicando diferentes transformadas (LDA, MLLT, SAT, fMLLR) en un modelo basado en HMM-GMM.
- Implementar un modelo basado en TDNN híbrido.
- Comparar resultados usando la medida de calidad Word Error Rate (WER).

1.3. ANTECEDENTES

Este proyecto se ha realizado bajo la supervisión de Asunción Moreno de la Escuela Técnica Superior de Telecomunicaciones de Barcelona y en el departamento de reconocimiento de voz de una compañía en Cracovia (Polonia) llamada Techmo. Esta empresa está derivada de la universidad Akademia Górniczo-Hutnicza University of Science and Technology (AGH UST) y son expertos en el campo del sonido.

Mi proyecto es independiente a los proyectos en los que están trabajando en Techmo actualmente, y es el primer modelo de reconocimiento de voz en español.

La idea principal fue proporcionado por mi tutor en Polonia, Bartos Ziólko; él es profesor de la universidad AGH UST y es CEO de Techmo. Pero mi supervisor, Tomasz Jadczyk, CTO de Techmo, es quién me ha proporcionado más ideas en cómo construir el proyecto.

1.4. WORK PACKAGE Y DIAGRAMA DE GANTT

<table>
<tr><td colspan="2">Proyecto: An ASR prototype for Spanish dictation</td><td colspan="2">WP ref: WP1</td></tr>
<tr><td colspan="2">Constituyente principal: Documentación</td><td colspan="2"></td></tr>
<tr><td colspan="2" rowspan="2">Breve descrición:

El objetivo es entregar la documentación necesaria a tiempo.

1. Project Proposal and Work plan (versión 1)
2. Project Proposal and Work plan (versión 2)
3. Project Critical Review
4. Final Report template</td><td colspan="2">Fecha inicio: 1/10/2019
Fecha final: 26/01/2020</td></tr>
<tr><td colspan="2">Evento inicio T1: 1/10/2019
Evento final T1: 8/10/2019
Evento inicio T2: 9/10/2019
Evento final T2: 17/10/2019
Evento inicio T3: 19/11/2019
Evento final T3: 1/12/2019
Evento inicio T4: 02/12/2019
Evento final T4: 26/01/2020</td></tr>
<tr><td colspan="2">Tarea interna T1: Project Proposal and Work plan (version 1)
Tarea interna T2: Project Proposal and Work plan (version 2)
Tarea interna T3: Project Critical Review
Tarea interna T4: Final Report template</td><td>Entregables:
T1
T2
T3
T4</td><td>Fechas:
8/10/2019
17/10/2019
30/11/2019
26/01/2020</td></tr>
</table>

Tabla 1: Work Package 1

<table>
<tr><td colspan="2">Proyecto: An ASR prototype for Spanish dictation</td><td colspan="2">WP ref: WP2</td></tr>
<tr><td colspan="2">Constituyente principal: Introducción software</td><td colspan="2"></td></tr>
<tr><td colspan="2" rowspan="2">Breve descrición:

Aprender reconocimiento automático del habla y cómo funciona el software de Kaldi.

Ejecutar los scripts de ejemplo de Kaldi para la base de datos Fisher Callhome Spanish.</td><td colspan="2">Fecha inicio: 1/10/2019
Fecha final: 26/01/2020</td></tr>
<tr><td colspan="2">Evento inicio T1: 1/10/2019
Evento final T1: 26/01/2020
Evento inicio T2: 1/10/2019
Evento final T2: 29/10/2019
Evento inicio T3: 30/10/2019
Evento final T3: 7/11/2019
Evento inicio T4: 8/11/2019
Evento final T4: 26/11/2019</td></tr>
<tr><td colspan="2">Tarea interna T1: Aprender RAH y de la herramienta de Kaldi
Tarea interna T2: Crear el léxico, arreglar transcripciones y preparar ficheros que necesita Kaldi
Tarea interna T3: Entrenar el modelo acústico basado en HMM-GMM con sus distintas transformadas y decodificar.
Tarea interna T4: Entrenar la red neuronal TDNN híbrida y decodificar.</td><td>Entregables:
Resultados %WER de T3
Resultados %WER de T4</td><td>Fechas:

7/11/2019

26/11/2019</td></tr>
</table>

Tabla 2: Work Package 2

Proyecto: An ASR prototype for Spanish dictation	WP ref: **WP3**	
Constituyente principal: Software		
Breve descrición: Añadir la base de datos de Common Voice al modelo TDNN anterior.	**Fecha inicio:** 27/11/2019 **Fecha final:**	
	Evento inicio T1: 27/11/2019 Evento final T1: 9/12/2019 Evento inicio T2: 10/12/2019 Evento final T2: 18/12/2019 Evento inicio T3: 19/12/2019 Evento final T3: 23/1/2020	
Tarea interna T1: Cambiar de entorno y preparar los archivos necesarios. **Tarea interna T2:** Entrenar el modelo TDNN añadiendo Common Voice **Tarea interna T3:** Obtener resultados WER de este modelo	Entregables: Resultados %WER T3	Fechas:

Tabla 3: Work Package 3

Proyecto: An ASR prototype for Spanish dictation	WP ref: **WP4**	
Constituyente principal:		
Breve descrición: Creación de scripts y análisis de resultados de las transcripciones del primer modelo	**Fecha inicio:** 19/12/2019 **Fecha final:** 9/01/2020	
	Evento inicio T1: 21/12/2019 Evento final T1: 31 /12 /2019 Evento inicio T2: 2/1/2020 Evento final T2: 13/01/2020	
Tarea interna T1: Creación de unos scripts para ayudarme a analizar los resultados. **Tarea interna T2:** Análisis de los resultados	Conclusiones de los resultados	13/01/2020

Tabla 4: Work Package 4

Nombre	Duracion	Inicio	Terminado
Project Proposal and Work plan (version 1)	5,875 days	1/10/19 9:00	8/10/19 17:00
Project Proposal and Work plan (version 2)	6,875 days	9/10/19 9:00	17/10/19 17:00
Project Critical Review	8,875 days	19/11/19 9:00	29/11/19 17:00
Final Report template	39,875 days	2/12/19 9:00	24/1/20 17:00
Aprender RAH y Kaldi	83,875 days	1/10/19 9:00	24/1/20 17:00
Preparar ficheros que necesita Kaldi	20,875 days	1/10/19 9:00	29/10/19 17:00
Entrenar modelo HMM-GMM	6,875 days	30/10/19 9:00	7/11/19 17:00
Entrenar modelo TDNN1	12,875 days	8/11/19 9:00	26/11/19 17:00
Cambio de entorno	8,875 days	27/11/19 9:00	9/12/19 17:00
Añadir datos de Common Voice	6,875 days	10/12/19 9:00	18/12/19 17:00
Hacer decodificación de TDNN2	25,875 days	19/12/19 9:00	23/1/20 17:00
Creación de scripts	8,875 days	19/12/19 9:00	31/12/19 17:00
Análisis resultados	7,875 days	2/1/20 9:00	13/1/20 17:00

Figura 1: Milestone

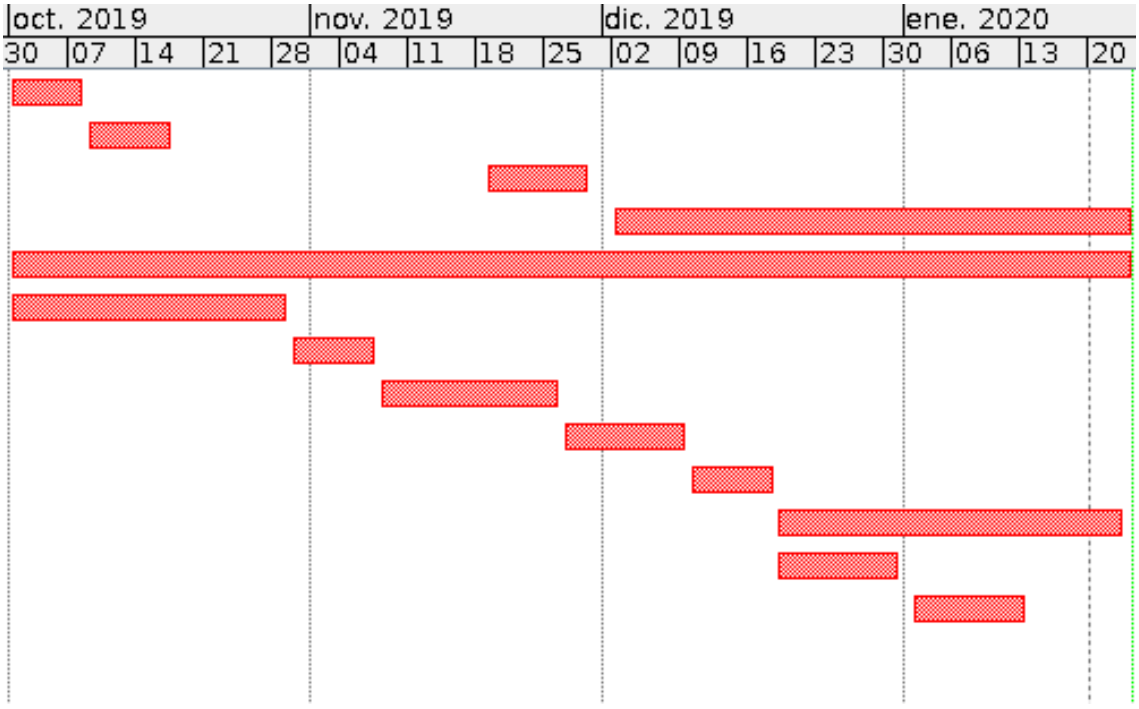

Figura 2: Diagrama de Gantt

1.5. DESVIACIONES DEL PLAN INICIAL Y INCIDENCIAS

Uno de los primeros contratiempos ha sido encontrar bastantes palabras en inglés y algunas de otros idiomas en las transcripciones, además de algunas faltas de ortografía que tuve que corregir.

La falta de experiencia que tenía en proyectos grandes y en el uso de Kaldi me hicieron prolongar el proyecto. Tuve varios problemas que ahora me resultarían más fáciles de resolver, pero en ese momento me hicieron perder tiempo.

Además, cuando tuve acceso al servidor de la empresa para poder entrenar la red neuronal, no todos los días estaba libre para trabajar con él, tenía que esperar mi turno para poder usarlo.Y si el entrenamiento había ido mal tenía que volver a empezar.

Al acabar de entrenar el primer modelo siguiendo los scripts de ejemplo de Kaldi y querer añadir los datos de Common Voice me percaté que tenía algunos ficheros que necesitaba Kaldi que estaban especializados para los datos de LCD, usando 2 canales, ya que eran conversaciones telefónicas, y que no podía crearlos usando unos datos de un solo canal como los de Common Voice. Se decidió cambiar de entorno. El nuevo entorno es el que la empresa usa para su modelo ASR en polaco. En él usan algunos scripts de Kaldi y algunos propios que hacen de él un entorno más adaptable para poder trabajar con modelos ya entrenados, además te permitirme trabajar con una base de datos de un solo canal, como la de Common Voice.

Su modelo se tenía que adaptar al uso de vectores ivectors (vectores que representan características del hablante), ya que mi modelo anterior los usaba para hacer la decodificación. Después, nos dio errores con fonemas posicionales; mi modelo usa fonemas dependientes de posición, tiene en cuenta si es inicio o final de frase, el suyo no. Al final, adaptar su modelo al uso de fonemas dependientes de posición llevaría más tiempo del esperado. Así que, finalemnte, entrené e hice el test con los scripts de Kaldi de ejemplo para la base de datos de Common Voice. Estos cambios de entorno supusieron perder mucho tiempo.

El supervisor de Polonia, muy frecuentemente trabajaba en remoto. La comunicación con éste regularmente era a través del chat de la empresa; las contestaciones a mis preguntas y dudas, en algunas ocasiones, llegaban con retraso y esto hecho hizo que el aprendizaje, y el desarrollo del trabajo no fuera tan constante como me hubiera gustado.

Me propuse varios objetivos, que no he alcanzado, como por ejemplo cambiar los parámetros de la arquitectura de la red neuronal e implementar un modelo “sequence to sequence”. Organicé mis objetivos sin pensar en los contratiempos que me fueron apareciendo, ni que debido a mi falta de experiencia me hacían ir más lenta. También, en Polonia el cuatrimestre se inicia un mes más tarde, en octubre, con lo cual quizás con un mes más hubiese podido alcanzar más objetivos.

2. Estado del arte:

2.1. RECONOCEDOR AUTOMÁTICO DEL HABLA

Un sistema de reconocimiento del habla (RAH) tiene la finalidad de extraer en formato texto los sonidos identificados, los más probables, a partir de la información acústica de los sonidos pronunciados, y independientemente del sistema que se ha usado para grabar la voz, el hablante o el entorno.

Un sistema RAH se rige por la regla de Bayes,es un clasificador Bayesiano. Dado una secuencia de observaciones de la señal de voz X_1^T se pretenden encontrar la secuencia de palabras que mejor defina esas observaciones, es decir, encontrar W_1^N que maximice la probabilidad a posteriori $P(W_1^N \mid X_1^T)$:

$$\left[w_1^N\right]_{\text{opt}} = \underset{w_1^N}{\operatorname{argmax}} \left\{p(w_1^N|x_1^T)\right\}$$

Para ello se aplica la Regla de Bayes :

$$p(w_1^N|x_1^T) = \frac{p(w_1^N) \cdot p(x_1^T|w_1^N)}{p(x_1^T)}$$

El denominador $P(X_1^T)$ se asume independiente de W_1^N, no influye, así que la expresión es equivalente a:

$$\left[w_1^N\right]_{\text{opt}} = \underset{w_1^N}{\operatorname{argmax}} \left\{p(w_1^N) \cdot p(x_1^T|w_1^N)\right\}$$

Finalmente, se concluye que para encontrar la secuencia de palabras óptima que maximice la probabilidad $P(W_1^N \mid X_1^T)$ tenemos que encontrar que maximice el producto de $P(W_1^N)$ y $P(X_1^T \mid W_1^N)$ Donde $P(W_1^N)$ es el modelo de lenguaje, que es la probabilidad a priori de las secuencias de palabras y $P(X_1^T \mid W_1^N)$ es el modelo acústico, que es la probabilidad de una secuencia de observaciones dado una secuencia de palabras. [3]

Estos dos modelos se explican más detalladamente en la sección 2.1.2 y 2.1.3.

Un sistema de reconocimiento de voz tiene la siguiente estructura:

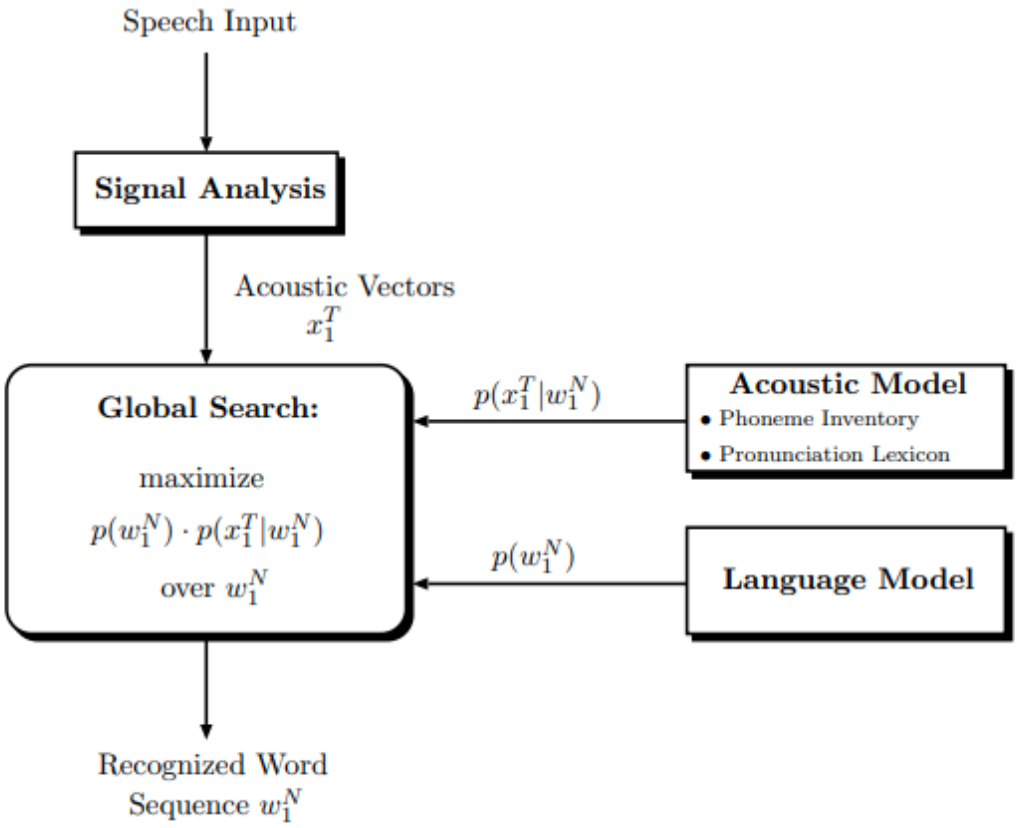

Figura 3: Arquitectura RAH

2.1.1. ANÁLISIS DE LA SEÑAL

El primer paso en un reconocedor del habla es extraer vectores de características de la señal de voz, las técnicas más usadas son Perceptual Linear Prediction (PLP) coefficients y el Mel frequency cepstrum coefficient (MFCC). [3]

En mi caso he usado el MFCC (Mel frequency cepstrum coefficient).

El MFCC es una representación perceptual basado en la escala de MEL, la escala que representa la percepción auditiva humana. El MFCC se calcula de la siguiente forma:[4] [5]

- Calcular el número de tramas,normalmente tramas de 25 ms solapadas 10ms.
- Para cada trama:
    1. Se extrae la data, opcionalmente se aplica filtro preénfasis y se multiplica por una ventana, tipicamente de Hamming.
    2. Se aplica la Transformada de Fourier discreta y se obtiene la energía espectral de la señal.
    3. Se aplica los filtros triangulares que siguen la escala de Mel a la potencia espectral obtenida y se suma las energías en cada uno de ellos.

4. Se calcula el logaritmo de la energía de cada frecuencia de MEL.
5. Se aplica la transformada discreta de coseno para cada logaritmo de la energía de la escala de MEL.
6. Finalmente los MFCC son las amplitudes de los espectros resultantes.Típicamente se obtienen entre 12 y 16 coeficientes. A veces, se escalan estos coeficientes para asegurar que estén en un rango razonable.

Otras transformaciones de la señal para mejorar el reconocimiento son: [4]

- **Delta feature computation**. La voz es dinámica y los vectores MFCC no tienen en cuenta cómo cambian los coeficientes a lo largo del tiempo. Para capturar esta información se calcula la derivada de primero y segundo orden ($\Delta + \Delta\Delta$) de los coeficientes espectrales de MFCC. Si tenemos 13 coeficientes de MFCC, con transformación $\Delta + \Delta\Delta$ tendremos un vector de 39 coeficientes MFCC. [4][5]
- **Linear discriminant analysis transform (LDA)** es una técnica para buscar una transformación lineal que reduce las dimensiones de los vectores de características y así maximizar la separabilidad de clase y en consecuencia, poder clasificar mejor.[6]
- M**aximum likelihood linear transform (MLLT),** también conocida como global STC, es una transformación de ortogonalización que hace que las características sean modeladas mejor mediante Gaussianas con una matriz de covarianza diagonal y así se puedan clasificar mejor ya que decorrela el espacio de características. [7] [8]
- **Features Maximum likelihood linear regression (fMLLR)**, también conocida como global Constrained MLLR (CMLLR) es una transformación para normalizar la variabilidad de hablantes. Es decir, generar unos vectores en que las diferencias entre hablantes sean lo menos influyentes posibles. En Kaldi se usa en el entrenamiento SAT (Speaker Adaptative Training). [7]

2.1.2. MODELO ACÚSTICO

El modelo acústico es uno de los componentes más importantes de un sistema de reconocimiento de voz. Su objetivo es encontrar $P(X_1^T \mid W_1^N)$, la secuencia de características acústicas observables X_1^T más probable, a partir de una secuencia de palabras W_1^N.

El modelo para palabras individuales y para secuencia de sentencias es obtenido concatenando los modelos acústicos de unidades fonéticas, siguiendo un diccionario fonético donde cada palabra tiene su pronunciación. [3]

Las unidades acústicas más usadas son los fonemas dependientes de contexto, los n-fonemas modelan el fenómeno de coarticulación (la variación de articulación que un fonema tiene en diferentes contextos) dentro de una palabra y entre palabras. [9]

En reconocimiento de voz de mucho vocabulario se usan n-fonemas, normalmente uno o dos fonemas concatenados, son los trifonemas o los quintofonemas.

Una misma palabra se puede pronunciar con diferente duración y configuración espectral, incluso para un mismo hablante. Las distorsiones temporales de las diferentes pronunciaciones,así como las variación espectral de la señal acústica se puede describir con los Modelos Ocultos de Markov (MOM). [3]

Un MOM es un modelo estadístico que está definido por un conjunto de estados con transiciones que sirve para representar secuencias de datos espaciales o temporales, como la señal de voz. El sistema a modelar es un proceso de Markov de parámetros desconocidos que se calculan a partir de parámetros observables. [10]

La probabilidad $P(X_1^T \mid W_1^N)$ es la suma de todas las posibles secuencias de estados S_1^T para la secuencia de palabras W_1^N . Cada "camino" a través de modelo es una alineación entre la secuencia de observaciones X_1^T y la secuencia de estados S_1^T de MOM.

$$p(x_1|w_1^N) = \sum_{s_1^T} p(x_1^T, s_1^T|w_1^N)$$

Esta ecuación se resuelve con Viterbi. [3]

La manera más frecuente de representar las relaciones entre estados MOM es usar GMM (Gaussian Mixture Model): GMM-HMM.

En este proyecto, además de usar GMM-HMM para crear un modelo acústico se han implementado redes neuronales profundas (DNN).

Las redes neuronales profundas que se han empleado son modelos DNN híbridos. Estas usan como "baseline" un alineamiento forzado producido por GMM-HMM y la salida son las probabilidades de los estados de HMM.[7].

Se dará una explicación más detallada en la sección 3.

2.1.3. MODELO DE LENGUAJE

El modelo P(W_1^N)define cómo se relaciona las palabras entre sí. Más concretamente, consiste en conocer la probabilidad a priori de una secuencia de palabras $W_1^N=W_1W_{2,..}...W_N$ independentemente de las observaciones acústicas. Acepta o rechaza (asigna baja probabilidad) secuencias de palabras.

Debido a que grandes vocabularios generan muchas combinaciones de palabras, no sería practicable su aplicación. La solución es usar modelos de N-gramas. En un modelo N-grama se asume que la probabilidad de un palabra W_N depende únicamente de sus n-1 anteriores palabras. [3]

$$P(w_1,\ldots,w_m)=\prod_{i=1}^{m}P(w_i \mid w_1,\ldots,w_{i-1}) \approx \prod_{i=1}^{m}P(w_i \mid w_{i-(n-1)},\ldots,w_{i-1})$$

$$P(w_i \mid w_{i-(n-1)},\ldots,w_{i-1})=\frac{\operatorname{count}(w_{i-(n-1)},\ldots,w_{i-1},w_i)}{\operatorname{count}(w_{i-(n-1)},\ldots,w_{i-1})}$$

La precisión de un modelo de lenguaje se puede medir con la perplejidad PP. [3]

$$PP=\left[\prod_{n=1}^{N}p(w_n|w_{n-m+1}^{n-1})\right]^{-1/N}.$$

2.1.4. DECISIÓN

En este bloque se elige la secuencia de palabras óptima, la que maximice la probabilidad del modelo de lenguaje y del modelo acústico, a partir de una secuencia de observaciones acústicas.

$$[w_1^N]opt=\underset{w_1^N}{argmax}\,P(w_1^N|x_1^T)=\underset{w_1^N}{argmax}\,(P(w_1^N)P(x_1^T|w_1^N))$$

El decodificador tiene que alinear la secuencia de observaciones X_1^T con todos los posibles estados de secuencia correspondientes a un secuencia de palabras W_1^N. Este proceso de optimización se puede resolver con el algoritmo de Viterbi, aproximando así, la suma de todas las secuencias de estados usando la secuencia de estados más probable. [3]

2.1.5. EVALUACIÓN

Un sistema común para evaluar el sistema de reconocimiento de voz es el Word Error Rate (WER).

Se calcula la WER entre la frase generada por el sistema, la hipótesis y la frase correcta, de referencia.

La expresión de WER es la siguiente:

$$WER = \frac{S + D + I}{N} = \frac{S + D + I}{S + D + C}$$

Donde:

- N es el número de palabras
- S es el número de sustituciones de una palabra por otra
- I es el número de palabras insertadas
- D es el número de palabras eliminadas.
- C es el número de palabras correctas

[11]

2.2. Redes neuronales

2.2.1. Introducción al deep learning

Deep learning es una técnica de Machine Learning que predice y clasifica información. Consiste en redes de neuronas interconectadas, organizadas en capas. La manera en la que están organizadas estas neuronas en capas conforma las diferentes arquitecturas. [12]

La versión más simple de una red neuronal es un perceptrón o neurona artificial simple. La función de una neurona se puede expresar como:

$$y = f\left(\sum (x_i W_i) + b \right)$$

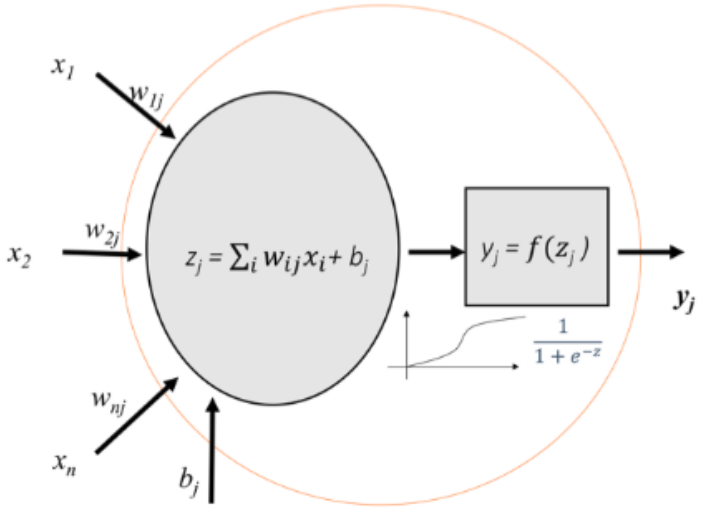

Figura 4: Neurona artificial

Donde Xi es las entradas, W_i es la matriz de pesos , b es el sesgo y sigma es una función de activación.

La función de activación define la salida de la neurona. Las más comunes son: el threshold, la sigmoide, la hiperbólica tangente y el rectifier.

Una red multicapa perceptron(MLP) tiene una capa de entrada, uno o más capas de perceptrones, llamadas capas ocultas y una capa final con varios perceptrones. [13]

El algoritmo de aprendizaje de una red neuronal consiste en :

1. Empezar asignando valores aleatorios para los parámetros de la red (W_i , b_i).
2. Coger unos primeros ejemplos del dataset y pasarlos por la capa de entrada.
3. Realizar la propagación forward, donde las neuronas son activadas de izquierda a derecha hasta obtener una predicción.
4. Comparar las predicción obtenida con los valores etiquetados esperados y calcular la función Loss.
5. Realizar backpropagation. El error es propagado de derecha a izquierda y se usa la técnica del gradiente para ajustar los pesos de las conexiones entre neuronas y minimizar la función de loss.
6. Repetir los pasos: aprendizaje reforzado (repetir los pasos y actualizar los pesos después de cada observación) o aprendizaje batch (repetir los pasos y actualizar después de un batch de observaciones). Donde el batch es el tamaño de datos de entrenamiento en una iteración del entrenamiento para actualizar el gradiente.
7. Cuando todos los datos de entrenamiento hayan pasado por la red es una época. Repetir para varias épocas. [12] [13]

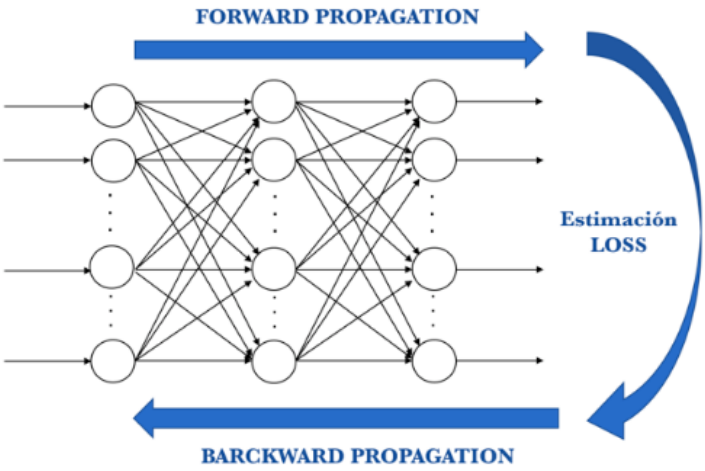

Figura 5: Algoritmo aprendizaje

2.2.2. Red neuronal recurrente

Las redes neuronales recurrentes (RNN) son un tipo de red neuronal donde se crean ciclos de realimentación. Una misma neurona recibe dos entradas, la entrada de la capa anterior y su propia salida en el instante anterior. Dado que la salida depende del tiempo anterior se podría decir que tiene memoria. [14]

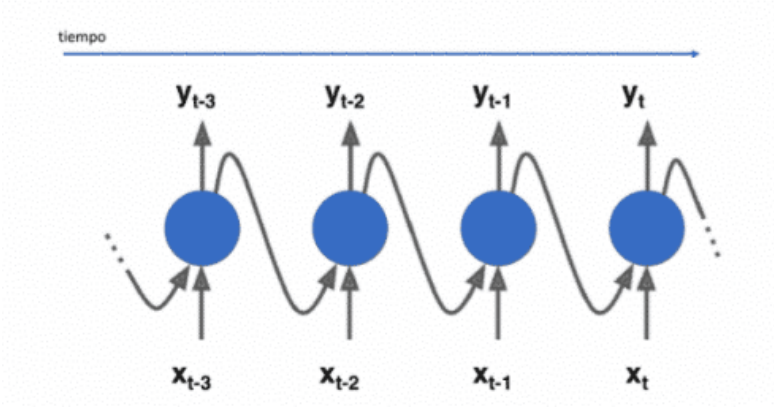

Figura 6: Neuronas de una red neuronal Recurrente

Este tipo de redes son capaces de modelar datos en secuencia, por eso son usadas para el procesado de la lengua,ya que en el habla las palabras vienen condicionadas por las palabras previas. [13]

$$y_t = f\,(W x_t + U y_{t-1} + b\,)$$

Donde Xt es la secuencia de entrada de la capa anterior, W es la matriz de pesos a la entrada, U es la matriz de pesos en el instante de tiempo anterior (y_{t-1}) y b es el sesgo. [14]

En el proyecto se han usado las redes neuronales recurrentes profundas tipo LSTM ya que es el tipo de RNN que aporta mejores resultados. [15]

2.2.3. Long short term memory

Long-Short Term Memory (LSTM) es un tipo de red neuronal recurrente (RNN). Estas tienen más memoria que las redes recurrentes convencionales. Pueden recordar sus entradas durante un largo periodo de tiempo y así extraer dependencias a corto o largo plazo y ver qué información es importante. [14]

LSTM contiene unidades especiales que son bloques de memoria en una capa escondida recurrente. Los bloques contienen celdas de memoria con auto-conexiones que almacenan el estado temporal de la red, además de unidades multiplicativas llamadas puertas de control para controlar el flujo de información. El bloque de memoria está formado por una celda de memoria, una puerta de entrada, una puerta de salida y una puerta de olvidar.

La puerta de entrada controla el flujo de activaciones de entrada en la celda de memoria. La puerta de salida controla el flujo de activaciones de salida de esa celda a las demás de la red. La puerta de olvido controla el grado en que un valor permanece en la celda. Todas ellas se generan con la función sigmoide.

Hay otras versiones de LSTM que contienen “ peephole connections” para que las puertas puedan saber el estado de la celda. [15]

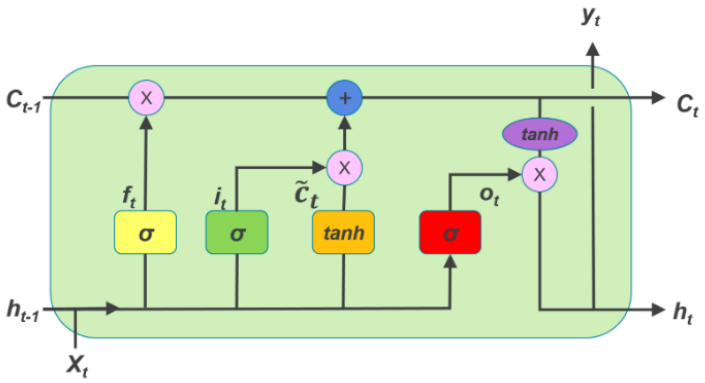

Figura 7: Celda de memoria LSTM

Para calcular una secuencia de salida $h_t = h_1 ... h_T$ o estado oculto, a partir de una secuencia de entrada $x_t = x_1 ... x_T$, se calculan las siguientes ecuaciones: desde t=1 a T.

$$i_t = \sigma(x_t U_i + h_{t-1} W_i + b_i)$$

$$f_t = \sigma(x_t U_f + h_{t-1} W_f + b_f)$$

$$o_t = \sigma(x_t U_o + h_{t-1} W_o + b_o)$$

$$q_t = tanh(x_t U_q + h_{t-1} W_q + b_q)$$

$$c_t = f_t * c_{t-1} + i_t * q_t$$

$$h_t = o_t * tanh(c_t)$$

Donde i_t es la puerta de entrada, f_t es la puerta de olvido, o_t es la puerta de salida que controla la cantidad de memoria que se expone en la salida h_t, q_t es un resultado temporal, c_t es el estado de la nueva celda. El estado oculto o salida de LSTM es h_t , b es el vector de bias de su respectiva puerta, U es la matriz de pesos de la entrada y W es la matriz de pesos de las recurrentes conexiones. [16]

3. Metodología y desarrollo del proyecto

3.1. La base de datos

El primer paso para la construcción de un modelo RAH consiste en la elección de la base de datos. La calidad de la base de datos repercutirá en la del sistema.

La base de datos utilizada para este proyecto es “Fisher and Callhome Spanish” [17] en español del catálogo Linguistic Data Consortium (LCD) y la base de datos en español de Common Voice. [18] La primera base de datos se ha usado para tener un primer modelo y la segunda base se ha añadido para mejorar los resultados del primer modelo.

El corpus de Fisher tiene aproximadamente 163 horas de conversaciones telefónicas de 136 hablantes de la zona del caribe y de no-caribe. En total son 819 ficheros de audio de 12 minutos aproximadamente cada uno. En cada conversación se habla de un tema específico, elegido arbitrariamente.

El corpus de Callhome son conversaciones telefónicas de hablantes de norteamérica. Consta de 120 archivos de audio de unos 10 minutos aproximadamente. En total son unas 20 horas.

El corpus Common Voice en español son 221 horas validadas. En él los hablantes son hispanos de diferentes zonas geográficas. Son 3275 archivos de audio de entrenamiento, 2729 de test y 2723 de validación. Cada uno con un máximo de 7 segundos.

Además de los archivos de audio se proporcionan las transcripciones, donde cada locución incluye tiempo de inicio y fin de esa frase en la grabación, y además, para las transcripciones de Fisher incluye el número del canal.

3.2. DIvisión de la base de datos

Para construir el modelo RAH se ha usado la herramienta Kaldi. En Kaldi hay algunos scripts de ejemplo para bases de datos populares, entre ellas la del Fisher and Callhome de LCD. En él, Callhome no se usa ni para el entrenamiento ni para el test. Los siguientes archivos son los de la base de datos de Fisher:

Base de datos	Archivos	Tiempo (horas) aproximadas
Entrenamiento	759	151.8
Validación	20	4
Test	20	4

Tabla 5: Base de datos del primer modelo

Para el segundo modelo, se ha usado la base de datos de Fisher, de Callhome y de CommonVoice :

Base de datos	Archivos	Tiempo (horas) aproximadas
Entrenamiento	4114	171.5
Validación	2759	12.62
Test	2769	12.63

Tabla 6: Base de datos del segundo modelo

3.3. Elaboración del diccionario fonético o léxico

Los ficheros que necesité para crear el RAH son los archivos de audio, las transcripciones y un diccionario fonético.

LCD ofrece el propio léxico para este corpus pero no pudimos disponer de él. Así que implementé un script para extraer de las transcripciones todas las palabras únicas y usé la herramienta Montreal Graphome to Phonema [19] para elaborar un diccionario fonético.

3.4. Modificación de las transcripciones

Al ser una base de datos elaborada en Estados Unidos había palabras en inglés, debido al escaso % de estas palabras en locuciones se decidió eliminarlas.

También modifique algunas pocas palabras que eran propias del francés o del alemán al fonema español , por ejemplo, la palabra “tour” reescrita “tur”.

Finalmente, corregí algunas faltas de ortografía en las transcripciones, como palabras con tildes abiertas y alguna palabra con ç.

4. Resultados

En esta sección se muestra los resultados para las diferentes experimentos que se realizaron.

En un primer momento se ha usado la base de datos de LCD para conseguir un "baseline system" del modelo, después se ha añadido más datos de CommonVoice con el fin de mejorar los resultados WER. Finalmente, se han analizado los resultado de las locuciones transcritas.

4.1. Experimento 1

El primer experimento es la construcción de un "baseline system" a partir de los scripts de ejemplo de Kaldi para la base de datos de Fisher and Callhome en español. Se comprueba la reducción de %WER en las diferentes transformadas que se aplican al sistema HMM-GMM, y después con un sistema HMM-DNN híbrido. Finalmente, se comparan estos resultados con los que vienen por defecto en los scripts de Kaldi para esta base de datos. [20] [21]

Cada entrenamiento de modelo acústico se hace de manera incremental, cada uno está basado en el anterior (menos el primer entrenamiento que se ha de crear desde cero). Los pasos para cada uno siempre son: entrenamiento y alineamiento, que consiste en asignar a cada fonema sus marcas temporales. Después, estos alineamientos son usados para posteriores entrenamientos.

En el primer paso se ha hecho un entrenamiento de monofonemas, fonemas independientes de contexto, con características MFCC y con una base de datos de 10000 frases cortas. Después se calcula el alineamiento del modelo monofonema con un conjunto de 30K locuciones, mono_ali.

El segundo paso se ha hecho a partir de los alineamientos monofonema calculados en el paso anterior, y usando MFCC + Δ+ΔΔ sobre un conjunto de 30000 locuciones mediante los alineamientos monofonema. Se obtiene el primer entrenamiento de trifonema con mono alineamientos, **tri1**, donde ya se tienen en cuenta el contexto. Se hace la alineación de las 30k locuciones con este modelo trifonema, tri1_ali.

A partir de estos últimos alineamientos de trifonemas, en el tercer paso se realiza un entrenamiento usando el conjunto de 30k locuciones obteniendo un modelo trifonema

con alineamiento trifonema, **tri2**. Se prosigue haciendo un alineamiento de este último modelo tri2 con un conjunto de 100K locuciones, tri2_ali.

Se aplica la técnica LDA+MLLT sobre el alineamiento tri2_ali obtenido anteriormente y el conjunto de locuciones de 100000, resultando un modelo trifonema con LDA+MLLT, **tri3a**. Se alinea este modelo tri3a con el conjunto de 100000 locuciones, pero esta vez, se usa un metodo de regresion lineal fMLLR.

Sobre esta alineación fMLLR se realiza un entrenamiento SAT (speaker adaptation training) usando el conjunto de locuciones 100000. A este entrenamiento le llamaremos **tri4a**. Se realiza el alineamiento con tri4a y todo el conjunto de datos de entrenamiento, tri4_ali.

A partir del alineamiento anterior se realiza un entrenamiento SAT (aumentando el número de gaussianas y el de hojas) usando todo el conjunto de locuciones de la base de datos de entrenamiento llamado t**ri5a**. Se hace el alineamiento de tri5a con todo el conjunto de datos de entrenamiento.

Finalmente, se usa este último modelo como entrada para la red neuronal LSTM, construyendo un HMM-DNN híbrido, ya que en el estado del arte vimos que era el tipo de RNN que daba mejores resultados.

En la tabla 7 se comparan los resultados que hemos obtenido con los de los scripts de Kaldi.

Modelos fonéticos	**Resultados %WER de los scripts de Kaldi**	**Nuestros resultados %WER**
tri1 (trifonema con alineamiento monofonema) uso de 30K locuciones	53.70	55.53
tri2 (trifonema con alineamiento trifonema) 30k	53.18	54.76
tri3a (trifonema +LDA + MLLT) 100k	46.95	48.86
tri4a (+SAT +fMLLR) 100k	42.86	43.84
tri5a (+gaussianas y hojas) se usa todo el corpus de entrenamiento	40.48	42.41
TDNN	22.21	22.82

Tabla 7: Resultados %WER de diferentes transformadas

Se puede observar que todas las transformaciones que se han aplicado mejoran los resultados %WER, además, podemos concluir que un sistema TDNN-HMM híbrido obtiene mejores resultado que un sistema basado en GMM-HMM .

En comparación a los resultados por defecto de Kaldi conseguimos unos resultados peores, pero no difieren demasiado de los suyos. El %WER superior llega a ser un 2% más alto cuando usamos HMM-GMM aplicando las diferentes transformaciones. Esta diferencia es inferior cuando usamos TDNN híbridas, nuestro modelo tiene un 0.6% más de error.

4.2. Experimento 2

El segundo experimento consiste en intentar mejorar los resultados, para ello se añadió más datos en el modelo, en concreto los datos de Common Voice, ya que es una base de datos gratuita.

Para realizar este experimento, no se usó los scripts de Kaldi de ejemplo para la base de datos LCD como se había hecho en el primer experimento, ya que en él se usa una base de datos con dos canales y Common Voice sólo usa un canal. Se cambió dos veces de entorno para poder usar las tres bases de datos. Estos cambios de entorno fueron uno de los contratiempos de este trabajo explicado en la sección 1.5.

Finalmente, se usó la “receta” de Kaldi para la base de datos de Common Voice en inglés ya que permite trabajar con una base de datos de un solo canal. Se adaptó el entorno para poder entrenar el sistema com las bases de datos que se quería, Common Voice en español, Fisher y Callhome. Estos fueron los resultados:

Modelos fonéticos	**Resultados %WER usando Common Voice, Fisher y Callhome**
monophone system (shortest 10k)	90.28
tri1 (trifonema con alineamiento monofonema) uso de 20K locuciones	64.23
tri2b (trifonema con alineamiento trifonema) 20k	61.17
tri3b (trifonema +LDA + MLLT + SAT) 20k	53.64
tri4b (+SAT +fMLLR) se usa todo el corpus de entrenamiento	48.54
TDNN	35.8

Tabla 8: Resultados %WER del segundo modelo

Como se puede observar en los resultados, estos no son los esperados, ya que son peores que el primer modelo usando sólo un base de datos.

Estos resultados indican que no se configuró bien los scripts de common Voice en inglés a las bases de datos de LCD y de Common Voice.

Es decir, cada “receta” de Kaldi está especializada a una base de datos en concreto, tiene sus propias configuraciones. Algunas de las configuraciones de estos scripts de ejemplo para la base de datos de Common Voice se han dejado por defecto en este modelo, y difieren a las de los scripts de ejemplo de Fisher and Callhome. Por ejemplo, el número de locuciones usadas en cada entrenamiento del modelo fonético.

4.3. Experimento 3

El tercer experimento consistió en analizar los resultados de las locuciones transcritas en el primer modelo, es decir, analizar cuáles son los errores en el modelo.

Para ello me ayudé de unos scripts que creé a partir de un fichero que crea Kaldi, en el cual salen las alineaciones de las locuciones de referencia y sus hipótesis, los tipos de errores en la alineación y el número de éstos. (Figura 8).

```
20051009_182032_217_fsp-A-002491-002724 ref  yo  soy  de  colombia  sí
20051009_182032_217_fsp-A-002491-002724 hyp  yo  soy  de  colombia  de
20051009_182032_217_fsp-A-002491-002724 op   C   C    C   C         S
20051009_182032_217_fsp-A-002491-002724 #csid 4 1 0 0
```

Figura 8: Ejemplo alineación locuciones

Los scripts extraen listas de palabras sustituidas (Tabla 9), palabras eliminadas y peores frases y con ellos pretendía obtener errores frecuentes. También, observar errores frecuentes; [22] si había locuciones fuera de contexto, si las frases con más errores tenían más ruido en sus transcripciones originales o palabras difíciles de entender o si las sustituciones se debían a que tenían los mismos fonemas.

Analizando estos ficheros y los resultados de las transcripciones ví que las sustituciones erróneas más frecuentes eran:

- en palabras con tildes y sin tildes.
- entre el plural y el singular de un palabra.
- debido a faltas de ortografía en las transcripciones originales

original	sustituida	#veces
mm	mhm	64
mhm	mmm	48
si	sí	45
sí	si	35
que	qué	28
las	la	21

Tabla 9: Ejemplo de sustituciones

```
20051109_210353_450_fsp-A-011509-011681  ref qué buen trabajoj
20051109_210353_450_fsp-A-011509-011681  hyp qué buen trabajo
```

Figura 9: Ejemplo de alineación con faltas de ortografía

En modelos de reconocimiento de voz en español normalmente no se utilizan las tildes para reconocimiento, en otros idiomas indican que es un fonema diferente, en español no. Solo indica si la vocal es tónica. Mi sistema RAH no utiliza características prosódicas, no distingue entre vocales con tilde o sin ella, no las puede detectar.
Como yo no quité las tildes a mis transcripciones y construí el diccionario con ellas, se calculó el %WER de las transcripciones sin tildes y así ver el impacto de éstas en los resultados.

%WER original TDNN1 (con tildes)	22.82
%WER TDNN1 (sin tildes)	21.92

Tabla 10: Comparación %WER con y sin tildes

Además, nos percatamos que tenía muchas frases cortas, de una o dos palabras, que eran “mm “ o “ si si “, y pensamos que estas frases hacían empeorar los resultados numéricos de las transcripciones. Por eso, se decidió calcular el %WER de las transcripciones sin esas frases y concluímos que estas frases nos hacían empeorar los resultados.

%WER original TDNN1	22.82
%WER TDNN1 sin frases cortas	22.67

Tabla 11: Comparación %WER con y sin frases cortas

También vimos que tenía desacoplos entre dialectos, es decir, al haber usado hablantes hispanos con diferentes dialectos tenía verbos mal sustituidos.

```
20051102_180402_391_fsp-A-000581-000697  ref de dónde llamas
20051102_180402_391_fsp-A-000581-000697  hyp de dónde llamás
```

Figura 10: Ejemplo de alineación con desacoplo de dialectos

Finalmente, las frases en que en la referencia hay varios <unk> (la risa, el ruido de fondo, etc en las referencia de locuciones está reescrito como <unk>), y sobre todo frases entre ruidos, no se llegan a reconocer en muchos casos. Por ejemplo:

```
20051113_210221_496_fsp-A-036769-037107 orig blanco papel <laugh> comparado con </laugh>
20051113_210221_496_fsp-A-036769-037107ref blanco papel ***
20051113_210221_496_fsp-A-036769-037107 hyp blanco papel <unk>
```

Figura 11: Ejemplo de alineación con <unk>

5. PRESUPUESTO

Los costes aproximados de este trabajo tienen en cuenta el material y el salario, el software no lo hemos tenido en cuenta ya que hemos usado herramientas que son gratuitas.

MATERIAL:

Se necesita un ordenador potente para trabajar, su coste aproximado es $\frac{700*0,9}{5}$ = 126€ por año, al no haberse usado todo el año,se ha usado $\frac{1}{3}$ de año. $\frac{126}{3}$ = 42€ . Además se necesita un servidor, $\frac{3000*0,9}{5}$ =540€, $\frac{540}{3}$ =180€

La base de datos de Fisher and Callhome se ha conseguido gracias a ser miembro de LCD, aproximadamente 21700 € ser miembro durante un año. $\frac{21700*0,9}{1}$ =19530€

Descripción	€/unidad	Vida útil	Coste
Base de datos Fisher and Callhome de LCD	21.700	1	19530
Ordenador	700	5	42
Servidor	3000	5	180

Tabla 12: Coste material

SALARIO:

Se ha considerado el sueldo como si estuviera contratada por una empresa en España, pero desplazada a Polonia.

Se ha asumido que un sueldo de ingeniero Junior es de 10€/hora y que se ha trabajado de lunes a viernes 8 horas al dia, desde octubre a enero, 16 semanas. Además, se tiene en cuenta los costes de la seguridad social que la compañía tienen que pagar, un 33.4%.

Descripción	€/hora	horas dedicadas	Seguridad social	Coste
Ingeniero Junior	10	640	2137,6	8537,68

Tabla 13: Coste salario

Coste total del proyecto:

Descripción	€/unidad
Material	19752
Salario	8537,68
Total	28289,68

Tabla 14: Coste total

6. Conclusiones y trabajo futuro

Uno de los objetivos de este proyecto ha sido construir un modelo de reconocimiento del habla para el español. Para ello, primero se construyó un sistema base utilizando la herramienta Kaldi. Se ha podido comprobar cómo se optimiza el sistema aplicando diferentes transformadas en un modelo trifonema basado en HMM-GMM y además, cómo se optimiza cuando usamos un modelo TDNN híbrido, usando como entrada el modelo HMM-GMM.

Además, se compararon nuestros resultados con los resultados de los scripts de Kaldi y se pudo observar que nuestro mejor modelo, el modelo TDNN híbrido, no difería mucho de sus resultados.

Seguidamente, se estudió el hecho de añadir más datos en el sistema para intentar mejorar los resultados.

Finalmente, se analizó los resultados de las transcripciones y pudimos encontrar errores frecuentes.

El principal objetivo del proyecto era aprender reconocimiento automático del habla y construir un modelo RAH usando la herramienta Kaldi, y este se ha cumplido. Empecé el trabajo sin haber usado, ni oído hablar sobre esta herramienta y finalizo el trabajo con un buen aprendizaje. No obstante, después de la realización del proyecto puedo afirmar que no se han cumplido todos los objetivos que se propusieron al principio; tales como, mejorar los resultados del primer modelo obtenido, cambiar los parámetros de la arquitectura de la red neuronal TDNN e implementar un modelo “sequence to sequence”.

Una vez concluido el proyecto y analizando los errores del sistema, se pueden proponer futuras mejoras:

- Para evitar desacoplos entre dialectos, es mejor escoger otra base de datos que no tenga palabras en inglés ni hablantes hispanos con diferentes dialectos.
- Es importante elegir la base de datos, no sólo el tipo de hablante, sino la manera en que está grabado. Es preferible que la base de datos que se elija esté grabada en las mismas condiciones,por ejemplo, que todas estén grabadas en uno o en dos canales.
- Dado que no se usan las tildes para reconocimiento del habla español, ya que en el español no son fonemas diferentes, para próximos modelos es mejor usar transcripciones sin las tildes y crear el léxico sin ellas.
- Si se sigue la “receta “ de Kaldi para la base de datos de Fisher and Callhome de LCD, se podría cambiar la distribución de los datos, ya que en los scripts por defecto no se usa la base de datos de Callhome, sólo se usa la de Fisher.

- Otro posible experimento sería variar parámetros de la red neuronal y ver si se puede conseguir mejores resultados.
- Uno de mis objetivos al principio era crear un modelo Sequence to Sequence usando la herramienta OpenSeq2Seq . Sería interesante probar si se obtienen mejores resultados con él.

Bibliografia:

[1] CVC(2003-2020*). Congreso de Sevilla*. “La lengua española y las nuevas tecnologías”. (n.d.). Available: https://cvc.cervantes.es/obref/congresos/sevilla/tecnologias/mesaredon_casacuberta. [Accessed: 20 January 2020]

[2] Forsberg, *Markus (2003) “Why is speech recognition difficult*.”

Available:https://www.researchgate.net/publication/228763868_Why_is_speech_recognition_difficult [Accessed: 20 January 2020]

[3] Wolfgang Macherey(2010) , “Dis*criminative Training and Acoustic Modeling for Automatic Speech Recognition”.*Avaiable: https://www-i6.informatik.rwth-aachen.de/publications/download/704/MachereyWolfgang--DiscriminativeTrainingAcousticModelingforAutomaticSpeechRecognition--2010.pdf [Accessed: 20 January 2020]

[4] *“Kaldi: Feature and model-space transforms in Kaldi.” (n.d.)* Available: http://kaldi-asr.org/doc/transform.html [Accessed: 20 January 2020]

[5] MFCC - Wikipedia, la enciclopedia libre. (2019). Available:https://es.wikipedia.org/wiki/MFCC [Accessed: 20 January 2020]

[6] Haeb-Umbach, R., & Ney, H. (n.d.). *“Linear discriminant analysis for improved large vocabulary continuous speech recognition”*. Available: https://www.math.arizona.edu/~hzhang/math574m/Read/ldaFace.pdf [Accessed: 20 January 2020]

[7] Rath, S. P., Povey, D., Vesel´y, K. V., Honza, J. ", & Cernocky, (n.d.). *“Improved feature processing for Deep Neural Networks”.* Available: https://www.danielpovey.com/files/2013_interspeech_nnet_lda.pdf .[Accessed: 20 January 2020]

[8] Psutka, J. V, & Müller, L. (n.d.). “ *Comparison of various feature decorrelation techniques in automatic speech recognition*”.Available: https://www.researchgate.net/publication/266496766_Comparison_of_various_feature_decorrelation_techniques_in_automatic_speech_recognition [Accessed: 20 January 2020]

[9] *Horno Chéliz, M.C, Ibarretxe Antuño, I Menívil Giró, JL.* - Prensas de la Universidad de Zaragoza, (2016). *“Panorama actual de la ciencia del lenguaje. Primer sexenio de Zaragoza”.* (pp 322).

[10]*”Modelo oculto de Márkov”* - Wikipedia, la enciclopedia libre-(2019)Available: https://es.wikipedia.org/wiki/Modelo_oculto_de_Márkov [Accessed: 20 January 2020]

[11] “*Word error rate*” - Wikipedia. (2019). Available: https://en.wikipedia.org/wiki/Word_error_rate [Accessed: 20 January 2020]

[12] *“The Complete Beginner’s Guide to Deep Learning: Artificial Neural Networks”*. (n.d.). Available: https://towardsdatascience.com/simply-deep-learning-an-effortless-introduction-45591a1c4abb [Accessed: 20 January 2020]

[13] Torres,Jordi (2018) “Deep Learning – *Introducción práctica con Keras”*.
Avaiable: https://torres.ai/deep-learning-inteligencia-artificial-keras/ . [Accessed: 20 January 2020]

[14]Torres,Jordi(2018) *“Redes Neuronales Recurrentes”*.Available: https://torres.ai/redes-neuronales-recurrentes/ [Accessed: 20 January 2020])

[15] Sak, H. H., Senior, A., & Google, B. (2014). “*Long Short-Term Memory Recurrent Neural Network* Architectures for Large Scale Acoustic Modeling.”

Available:https://static.googleusercontent.com/media/research.google.com/es//pubs/archive/43905.pdf [Accessed: 20 January 2020]

[16] Yin, W., Kann, K., Yu, M., SchützeSch, H., & Munich, L. (2017). "*Comparative Study of CNN and RNN for Natural Language Processing*." Available: https://arxiv.org/abs/1702.01923 [Accessed: 20 January 2020]

[17] "*Fisher and CALLHOME Spanish--English Speech Translation"* - Linguistic Data Consortium. (2014). Available: https://catalog.ldc.upenn.edu/LDC2014T23 [Accessed: 20 January 2020]

[18] Common Voice. (n.d.). Available:https://voice.mozilla.org/es/datasets [Accessed: 20 January 2020]

[19] Grapheme-to-Phoneme (G2P) - Montreal Forced Aligner 1.0 documentation. (2018). Available: https://montreal-forced-aligner.readthedocs.io/en/latest/g2p.html [Accessed: 20 January 2020]

[20] kaldi/RESULTS at master · kaldi-asr/kaldi. (2017). Available:

https://github.com/kaldi-asr/kaldi/blob/master/egs/fisher_callhome_spanish/s5/RESULTS [Accessed: 20 January 2020]

[21] [egs] Add more modern DNN recipe for fisher_callhome_spanish (#2951) · kaldi-asr/kaldi@ca32c4e.(2019).
Available:https://github.com/kaldi-asr/kaldi/commit/ca32c4e2bd77544c4ad4aa0bba25f0b812ea1a61
[Accessed: 20 January 2020]

[22] Shivakumar, P. G., Li, H., Knight, K., & Georgiou, P. (2018). "*Learning from Past Mistakes: Improving Automatic Speech Recognition Output via Noisy-Clean Phrase Context Modeling*." Available: https://www.researchgate.net/publication/323027143_Learning_from_Past_Mistakes_Improving_Automatic_Speech_Recognition_Output_via_Noisy-Clean_Phrase_Context_Modeling [Accessed: 20 January 2020]

Glosario

ASR: Automatic Speech Recognition

CEO: Chief Executive Officer

CMLLR: Constrained Maximum Likelihood Linear Regression

CTO: Chief Technical Officer

DNN: Deep Neuronal Network. Red Neuronal profunda.

GMM: Gaussian Mixture Model

HMM: Hidden Markov Model

fMLLR: Features Maximum likelihood linear regression

LCD: Linguistic Data Consortin

LDA: Linear Discriminant Analysis

LSTM: Long Short-Term Memory

MFCC: Mel Frequency Cepstrum Coefficient

MLLT: Maximum likelihood linear transform

MLP: Multilayer perceptron

MOM: Modelos Ocultos de Markov

SAT: Speaker Adatation Trainin

RAH: Reconocimiento automático del habla.

RNN: Recurrent Neuronal Network.

TDNN: Time Delay Neuronal Network.

WER: Word Error Rate