Universidad Zaragoza
1542

Trabajo Fin de Grado

Desarrollo y evaluación de herramientas para alineamiento automático de audio y texto con sistemas de reconocimiento automático del habla

Autor

Pablo Gimeno Jordán

Directores

Alfonso Ortega Giménez
Julia Olcoz Martínez

Escuela de Ingeniería y Arquitectura
2016

TRABAJOS DE FIN DE GRADO / FIN DE MÁSTER

Escuela de Ingeniería y Arquitectura Universidad Zaragoza

DECLARACIÓN DE AUTORÍA Y ORIGINALIDAD

(Este documento debe acompañar al Trabajo Fin de Grado (TFG)/Trabajo Fin de Máster (TFM) cuando sea depositado para su evaluación).

D./Dª. Pablo Gimeno Jordán,

con nº de DNI 72997678W en aplicación de lo dispuesto en el art. 14 (Derechos de autor) del Acuerdo de 11 de septiembre de 2014, del Consejo de Gobierno, por el que se aprueba el Reglamento de los TFG y TFM de la Universidad de Zaragoza,

Declaro que el presente Trabajo de Fin de (Grado/Máster) Grado, (Título del Trabajo) Desarrollo y evaluación de herramientas para alineamiento automático de audio y texto con sistemas de reconocimiento automático del habla,

es de mi autoría y es original, no habiéndose utilizado fuente sin ser citada debidamente.

Zaragoza, 20 de Junio de 2016

Fdo: Pablo Gimeno Jordán

A mis padres

Agradecimientos

Han sido muchas las personas que a lo largo de la realización de este Trabajo Fin de Grado me han dedicado una parte de su valioso tiempo. De lo contrario, el trabajo habría resultado mucho más complejo. Sirvan estas breves lineas como muestra de mi agradecimiento.

En primer lugar me gustaría dar las gracias a mis directores, Alfonso Ortega y Julia Olcoz, por sus explicaciones, consejos y correcciones, que han hecho posible la redacción de este Trabajo Fin de Grado. Ha sido un placer trabajar con vosotros y quiero agradeceros la confianza que habéis depositado en mi.

De igual forma, hago extensivo este agradecimiento a todos los miembros de ViVoLab, grupo donde se ha desarrollado este Trabajo Fin de Grado, y a mis compañeros de laboratorio. Su disposición ha sido completa durante los meses en los que se ha desarrollado este trabajo y han sido otro punto de apoyo importante para su realización.

A mi familia, y en especial a mis padres, por su comprensión y apoyo constante durante todos estos años. Sin vosotros no habría podido llegar hasta donde estoy ahora.

Un recordatorio también para todos los compañeros que he conocido en estos 4 años, con los que he compartido clases, prácticas e innumerables momentos de estudio. Por último, no me quiero olvidar de todas aquellas personas con las que tuve la suerte de compartir mi estancia en Göteborg.

A todos, muchas gracias.

Pablo Gimeno Jordán
20 de Junio de 2016

DESARROLLO Y EVALUACIÓN DE HERRAMIENTAS PARA ALINEAMIENTO AUTOMÁTICO DE AUDIO Y TEXTO CON SISTEMAS DE RECONOCIMIENTO AUTOMÁTICO DEL HABLA

RESUMEN

El objetivo del Reconocimiento Automático del Habla (RAH) es, dada una señal de voz, extraer la secuencia de palabras que han sido pronunciadas. Para poder llevar a cabo su tarea correctamente, un sistema de RAH precisa de ciertos conocimientos que obtiene a través de una fase de entrenamiento. Dicho aprendizaje se basa en dos modelos: el Modelo Acústico para caracterizar la señal de voz, y el Modelo de Lenguaje, relativo al vocabulario en ella utilizado. Este Trabajo Fin de Grado toma como punto de partida un motor de RAH para desarrollar y poner a prueba un sistema capaz de alinear el texto del guión de un programa de televisión con su correspondiente audio y obtener una localización temporal precisa de cada una de las palabras locutadas. Bajo esta premisa, se consideran diferentes estrategias de alineamiento.

El principal problema que se nos plantea es la incertidumbre al localizar el texto en el audio, ya que, *a priori* no se tiene ninguna información. Como primera estrategia se propone, realizar un reparto uniforme del texto en el audio del programa. Así, se llevan a cabo una serie de experimentos que permiten caracterizar el sistema de alineamiento y obtener una primera referencia de sus prestaciones.

Para disminuir la ambigüedad en la localización del texto en el audio se incluye un nuevo módulo en el sistema de alineamiento capaz de obtener marcas temporales parciales que sirvan de guía. Tras una nueva serie de experimentos se comprueba que esta estrategia supone una mejora relativa cercana al 12 % respecto de las prestaciones ofrecidas por el sistema base.

Demostrada la eficacia del uso de marcas temporales parciales, y en un intento por mejorar aun más el sistema de alineamiento, se utiliza una herramienta desarrollada para paliar las limitaciones del reconocedor en los finales de palabras, obteniendo una mejora relativa en torno al 20 % respecto del sistema base, que alcanza valores próximos al 23 % cuando se incluye la información de las intervenciones de cada locutor en el sistema de alineamiento.

Por tanto, a la vista de las resultados obtenidos en este Trabajo Fin de Grado, se concluye que el uso de estrategias que permitan reducir la incertidumbre en la localización del texto en el audio resultan adecuadas en este contexto, quedando probada la mejora de prestaciones que suponen en el sistema de alineamiento.

Índice general

Índice de figuras

Índice de tablas

Listado de acrónimos

AR Autorregresivo

BBC *British Broadcast Corporation*

DCT *Discrete Cosine Transform* (Transformada de Coseno Discreta)

GMM *Gaussian Mixture Model* (Modelo de Mezcla de Gaussianas)

HMM *Hidden Markov Model* (Modelo Oculto de Markov)

HTK *Hidden Markov Model Toolkit*

I3A Instituto de Investigación en Ingeniería de Aragón

JSGF *Java Speech Grammar Format*

LPC *Linear Predictive Coding*

MA Modelo Acústico

MAP Maximum A Posteriori

MFCC *Mel Frequency Cepstrum Coefficients* (Coeficientes Mel-Cepstrum)

MGB *Multi-Genre Broadcast*

ML Modelo de Lenguaje

MLE Modelo de Lenguaje Estocástico

NIST *National Institute of Standards and Technology*

pdf *probability density function* (función de densidad de probabilidad)

RAH Reconocimiento Automático del Habla

SCLITE *Score Lite*

TFG Trabajo Fin de Grado

ViVoLab *Voice Input Voice Output Laboratory*

WER *Word Error Rate*

Capítulo 1

Introducción

En este primer capítulo se ofrece una visión general del trabajo desarrollado, introduciendo sus principales conceptos y presentando sus objetivos. Así mismo se contextualizará dentro del entorno en el que se ha realizado, y se definirá la metodología de trabajo seguida. Finalmente, se detallará la estructura de este documento en el que se plasma el trabajo acometido.

1.1. Contexto y trabajo previo

La voz, según el diccionario de la Real Academia Española, es el sonido producido por la vibración de las cuerdas vocales. En términos de señal, se trata de un proceso estocástico no estacionario que se suele modelar como un sistema Autorregresivo (AR), o sistema todo polos. No importa con cual de las dos acepciones nos quedemos, la voz es algo fundamental en nuestro día a día, ya que es nuestro método principal de comunicación. Es por esto que las tecnologías relacionadas con la voz, las denominadas tecnologías del habla, se han desarrollado tan rápido en estos últimos años. Son muchas las aplicaciones de este tipo de tecnologías: una de las principales es el Reconocimiento Automático del Habla (RAH) que, dada una señal de voz, pretende extraer la secuencia de palabras que ha sido pronunciada por el locutor con la mayor precisión posible. El RAH ya está muy integrado en nuestro entorno: la mayoría de los *smartphones* modernos poseen sistemas de reconocimiento de voz cada vez más versátiles y más adaptados al habla espontánea. No obstante, este tipo de tecnologías no se centran únicamente en el reconocimiento de voz. Sus aplicaciones son mucho más amplias y cubren un gran abanico de posibilidades, como puede ser la síntesis de voz o las aplicaciones biométricas para reconocimiento de locutor.

Este Trabajo Fin de Grado (TFG) parte de un núcleo basado en motores de RAH para ir más allá, hacia una aplicación concreta adaptada a la industria audiovisual para la generación automática de subtítulos. El trabajo realizado en esta memoria se enmarca dentro del grupo de tecnologías del habla Voice Input Voice Output Laboratory (ViVoLab)[1], perteneciente al Instituto de Investigación en Ingeniería de Aragón (I3A) de la Universidad de Zaragoza.

También es importante para contextualizar este trabajo introducir el *Multi-Genre Broadcast Challenge* (*MGB Challenge*)[2]. Se trata de una evaluación de reconocimiento del habla, diarización del locutor y alineamiento parcialmente supervisado, usando grabaciones de televisión en inglés. La base de datos está compuesta por aproximadamente 1600 horas de audio tomado de los canales de la *British Broadcast Corporation* (BBC). De las tareas propuestas en esta evaluación, este trabajo se fundamentará en la tarea de alineamiento, que se detallará en

los siguientes capítulos.

1.2. Objetivos y alcance

El principal objetivo de este TFG es el desarrollo y evaluación de sistemas de sincronización audio-texto basados en motores de RAH. Se trata de obtener un alineamiento preciso entre el audio de un programa de televisión y el texto proveniente de la transcripción de su contenido. Esta tarea, que a priori puede parecer sencilla, es de gran interés para la industria audiovisual, ya que un sistema preciso podría dar lugar a importantes reducciones de coste a la hora de producir el subtitulado de contenidos, así como lograr sistemas de indexación y recuperación de información multimedia de altas prestaciones.

El aumento de la demanda de contenidos multimedia ha hecho que aumente también la demanda de contenido subtitulado. Un sistema como el propuesto en este TFG simplificaría la tarea del subtitulado a las empresas involucradas en el sector audiovisual de forma notable, convirtiendo este proceso en algo en gran medida automatizado.

El sistema de alineamiento a implementar resulta complejo y dependiente de varios parámetros que interfieren entre sí, lo cual justifica la necesidad de realizar un estudio para comprobar su influencia a la salida del sistema. Por lo que, resumiendo, el fin último del trabajo es comprobar donde están los límites del sistema de alineamiento y determinar el conjunto de parámetros que maximice la precisión a su salida.

1.3. Metodología de trabajo

El grueso de las herramientas utilizadas para el desarrollo de este TFG han sido desarrolladas por el grupo ViVoLab y otras se encuentran disponibles a modo de *toolkit* de libre distribución. Como núcleo del sistema de alineamiento tenemos el reconocedor de voz ViVoReco, desarrollado por el grupo de investigación y utilizado previamente en varios proyectos. En las siguientes secciones se llevará a cabo una explicación detallada de todas las herramientas que intervienen en el sistema de alineamiento.

Con dichas herramientas, se llevará a cabo una evaluación de diversas estrategias que permitan un correcto alineamiento entre el audio de un programa y su transcripción, muchas veces imprecisa debido a diferencias entre el guión y el texto realmente locutado, o a posibles errores al transcribir. Se partirá de un primer sistema base cuyos parámetros de funcionamiento se optimizarán para posteriormente comprobar las prestaciones al añadir un módulo auxiliar que proporcione marcas temporales incompletas de parte del contenido del programa.

El grupo de investigación dispone de un *cluster* de computación para el desarrollo de sus tareas de investigación. Este *cluster*, compuesto por 8 nodos de computación resulta muy útil a la hora de realizar tareas complejas y paralelizar trabajos. Todas las herramientas citadas anteriormente se utilizaron con este *cluster* como soporte para su ejecución.

Teniendo en cuenta todo lo expuesto, se desarrolló un plan de trabajo donde se explican las diferentes fases necesarias para la consecución de los objetivos expuestos anteriormente:

1. Familiarización con el entorno de trabajo y las herramientas a utilizar.

2. Evaluación de las prestaciones del sistema base: se pretende realizar un barrido sobre

todos los parámetros para comprobar como influyen en el resultado de alineamiento, de manera que encontremos cual es la configuración óptima de parámetros.

3. Evaluación de las prestaciones del sistema al añadir módulos auxiliares con información temporal incompleta: nueva serie de experimentos considerando distintas estrategias de alineamiento, que parten de la configuración óptima del sistema de base con el objetivo de mejorar sus prestaciones. Del mismo modo, se pretende encontrar la mejor de las estrategias y su configuración de parámetros asociada.

4. Interpretación de los resultados experimentales y evaluación crítica de las estrategias de alineamiento: a la vista de los resultados obtenidos será necesario realizar un análisis detallado de las causas y consecuencias de los mismos.

5. Elaboración de la memoria: esta fase y la anterior se solapan en el tiempo. Una vez finalizada la parte experimental, es necesario plasmar por escrito los resultados obtenidos y las conclusiones extraídas al analizarlos.

1.4. Organización de la memoria

Además de este primer capítulo donde se introduce el trabajo realizado, este documento consta de 5 capítulos más y un apéndice, que son descritos a continuación:

- **Capítulo 2 - Reconocimiento automático del habla**: este capítulo fundamenta las bases teóricas del RAH, núcleo principal del sistema de alineamiento a desarrollar.
- **Capítulo 3 - Tarea de alineamiento del *MGB Challenge***: en este capítulo se presenta detalladamente la tarea de alineamiento del *MGB Challenge*, sobre la cual se sustenta este TFG.
- **Capítulo 4 - Alineamiento con el sistema base**: este capítulo presenta una descripción de cada uno los bloques que forman el sistema de alineamiento a evaluar, introduciendo los parámetros que lo rigen. Se lleva a cabo también una primera serie de experimentos para caracterizar dicho sistema.
- **Capítulo 5 - Alineamiento con anclas temporales**: este capítulo introduce una mejora respecto al sistema base incluyendo un módulo auxiliar que proporciona unas marcas temporales parciales generadas de forma automática. Se llevan a cabo nuevos experimentos para evaluar las prestaciones del sistema de alineamiento con esta nueva estrategia.
- **Capítulo 6 - Conclusiones y líneas futuras**: este capítulo recoge las conclusiones extraídas tras la realización de este trabajo y describe brevemente las posibles líneas futuras a seguir.
- **Apéndice A - Resultados del alineamiento con anclas temporales**: se presentan los resultados completos de la primera serie de experimentos realizados con el sistema de alineamiento con anclas temporales.

Capítulo 2

Reconocimiento Automático del Habla

Este capítulo presenta los fundamentos teóricos del Reconocimiento Automático del Habla (RAH), núcleo central del sistema de alineamiento que se evalúa en este trabajo fin de grado.

2.1. Introducción

Un sistema de RAH tiene como objetivo dada una señal de voz, extraer el conjunto de palabras contenidas y transcribirlas en forma de texto. Generalmente, los sistemas de RAH se han clasificado según varias características:

1. En función del **locutor**:
   - **Sistemas dependientes del locutor**: el sistema ha sido entrenado con la voz de una única persona (monolocutor) o de un conjunto reducido de personas (multilocutor).
   - **Sistemas independientes del locutor**: el sistema está preparado para trabajar con cualquier locutor. Se necesitan grandes cantidades de datos que sean capaces de representar suficientemente la variabilidad en la voz.
2. En función del tamaño del **vocabulario** a su salida:
   Se define el vocabulario de un sistema de RAH como el conjunto de palabras que puede dar como salida.
   - **Pequeño vocabulario**: menos de 100 palabras
   - **Vocabulario medio**: entre 100 y 5000 palabras
   - **Gran vocabulario**: más de 5000 palabras
3. En función del **objetivo de reconocimiento**
   - **Palabras aisladas**: la unidad básica de reconocimiento es la palabra. El sistema requiere una pronunciación con suficiente pausa entre el final de una palabra y el comienzo de la siguiente. Este tipo de sistemas es adecuado, por ejemplo, para contextos de menús de opciones (ordenes claras y precisas). En cambio, no tiene sentido en contextos de habla natural o espontanea, ya que representa un modo de comunicación no natural para el ser humano.
   - **Habla continua**: el objetivo es el reconocimiento de frases completas que se pronuncian sin restricciones. El usuario se comunica de forma natural, haciendo que este tipo de sistemas se sitúen más cercanos a la realidad.

2.2. Reconocimiento estadístico

En la actualidad, las técnicas más populares para el RAH son de carácter estadístico y normalmente se basan en una decisión Maximum A Posteriori (MAP), o estimación de la moda de la distribución a posteriori. El primer paso en todos los sistemas de RAH es el de extracción de características de la señal de voz. Aplicando técnicas de procesado de señal que se explicarán más adelante, se obtiene una secuencia de T vectores $\mathbf{O} = [O_1, O_2, ..., O_T]$. A la salida del sistema de RAH se desea obtener la secuencia de N palabras que ha sido pronunciada, y que denominamos $\mathbf{W} = [W_1, W_2, W_3, ..., W_N]$. El sistema de RAH ofrecerá a su salida el conjunto de palabras $\mathbf{W}^*$, que viene determinado según la ecuación (2.1).

$$\mathbf{W}^* = arg \max_{\mathbf{W}} P(\mathbf{W}|\mathbf{O}) \tag{2.1}$$

Para determinar $P(\mathbf{W}|\mathbf{O})$, probabilidad de que se haya pronunciado la secuencia de palabras $\mathbf{W}$ habiendo observado la secuencia acústica $\mathbf{O}$, se hace uso del teorema de Bayes formulado en la ecuación (2.2).

$$P(W|O) = \frac{P(O|W)P(W)}{P(O)} \tag{2.2}$$

Así podemos reescribir la ecuación (2.1) como (2.3).

$$\mathbf{W}^* = arg \max_{\mathbf{W}} \frac{P(\mathbf{O}|\mathbf{W})P(\mathbf{W})}{P(\mathbf{O})} \tag{2.3}$$

Vemos que en la ecuación (2.3) intervienen tres términos distintos:

- $P(\mathbf{O}|\mathbf{W})$: probabilidad de observar la secuencia acústica $\mathbf{O}$ dada la secuencia de palabras $\mathbf{W}$. A menudo se le denomina verosimilitud.
- $P(\mathbf{W})$: probabilidad a priori de que la secuencia de palabras pronunciada sea $\mathbf{W}$, es decir, sin tener ninguna evidencia acústica.
- $P(\mathbf{O})$: probabilidad de observar la secuencia acústica $\mathbf{O}$.

Dado que el término $P(\mathbf{O})$ no afecta a la maximización, ya que es constante respecto de todas las posibles secuencias de palabras $\mathbf{W}$ evaluadas, podemos expresar (2.3) como (2.4), que será la ecuación fundamental que regirá el comportamiento del sistema de RAH.

$$\mathbf{W}^* = arg \max_{\mathbf{W}} P(\mathbf{O}|\mathbf{W})P(\mathbf{W}) \tag{2.4}$$

A la vista de (2.4), la salida del sistema de RAH será aquella que maximice el producto dos términos:

- $P(\mathbf{O}|\mathbf{W})$: equivalente a maximizar la probabilidad de la secuencia acústica $\mathbf{O}$ dada la secuencia de palabras $\mathbf{W}$. Este término vendrá dado por el Modelo Acústico (MA), que contiene la información sobre las unidades acústicas que componen cada palabra (modelo léxico) y la información estadística que caracteriza la secuencia de medidas acústicas.
- $P(\mathbf{W})$: equivalente a maximizar la probabilidad de pronunciar una secuencia de palabras $\mathbf{W}$. Este término vendrá dado por el Modelo de Lenguaje (ML), que contiene la información sobre las posibles transiciones entre una palabra y otra.

Por tanto, todo sistema de RAH se sustenta sobre dos modelos, que son imprescindibles para su correcto funcionamiento, el MA y el ML, descritos con más detalle en las secciones 2.5 y 2.6 de esta memoria.

2.3. Extracción de características

El bloque inicial que compone todos los sistemas de RAH es un procesador de señal que se encarga de adaptar la señal de voz a un formato óptimo para el reconocedor. Este proceso se suele denominar **extracción de características** y permite obtener la serie de secuencias acústicas **O** presentadas en la sección 2.2.

Son varios los parámetros a utilizar en el reconocimiento, teniendo todos ellos en común el uso del análisis localizado. La señal de voz es una señal no estacionaria, por eso es necesario realizar un análisis en ventanas temporales pequeñas que permitan asumir un comportamiento estacionario, tal y como se expresa en la ecuación (2.5):

$$Q_n = \sum_{k=-\infty}^{\infty} T(x[n]w[n-k]) \tag{2.5}$$

Donde $x[n]$ es la señal de voz, $w[n]$ es la ventana de K muestras que se aplica a la señal de voz y $T()$ es la transformación realizada. Algunas de las transformaciones más comunes son:

- Transformada de Fourier
- Predicción lineal (LPC o *Linear Predictive Coding*)
- Coeficientes de Mel-Cepstrum (MFCC o *Mel Frequency Cepstrum Coefficients*)

De todas las anteriores, los coeficientes de Mel-Cepstrum se han convertido en la opción más utilizada en los sistemas actuales, debido a la mejora de prestaciones que presentan respecto al resto.

2.3.1. Coeficientes de Mel-Cepstrum

Los coeficientes Mel-Cepstrum tienen su base en la escala de frecuencias Mel, cuya representación es lineal hasta los 1000 Hz y logarítmica de ahí en adelante. Esto se sustenta en el modelo de percepción auditiva del ser humano, cuya discriminación en frecuencia es mayor para frecuencia bajas que para frecuencias altas.

Sea $x[n]$ la señal digitalizada obtenida a partir de la señal analógica $x(t)$ en el micrófono. Podemos definir el cepstrum complejo de $x[n]$ según (2.6)

$$\hat{x}[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} \ln(X(e^{j\omega n}))e^{j\omega n}d\omega \tag{2.6}$$

donde $X(e^{j\omega n})$ es la transformada de Fourier de la señal $x[n]$. Por tanto, el cepstrum complejo de la señal $x[n]$ se define como la transformada de Fourier inversa del logaritmo neperiano de la transformada de Fourier de la señal $x[n]$.

En la señal de voz la información relevante se encuentra en el módulo, por tanto podemos prescindir de la información de fase de $x[n]$. Para ello, definimos en (2.7) lo que se conoce como el cepstrum real:

$$c[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} \ln|X(e^{j\omega n})|e^{j\omega n}d\omega \tag{2.7}$$

Así, el cepstrum real es la transformada de Fourier inversa del logaritmo neperiano del módulo de la transformada de Fourier de la señal.

Para convertir a la escala Mel se suele aplicar un banco de filtros como el mostrado en la Figura 2.1.

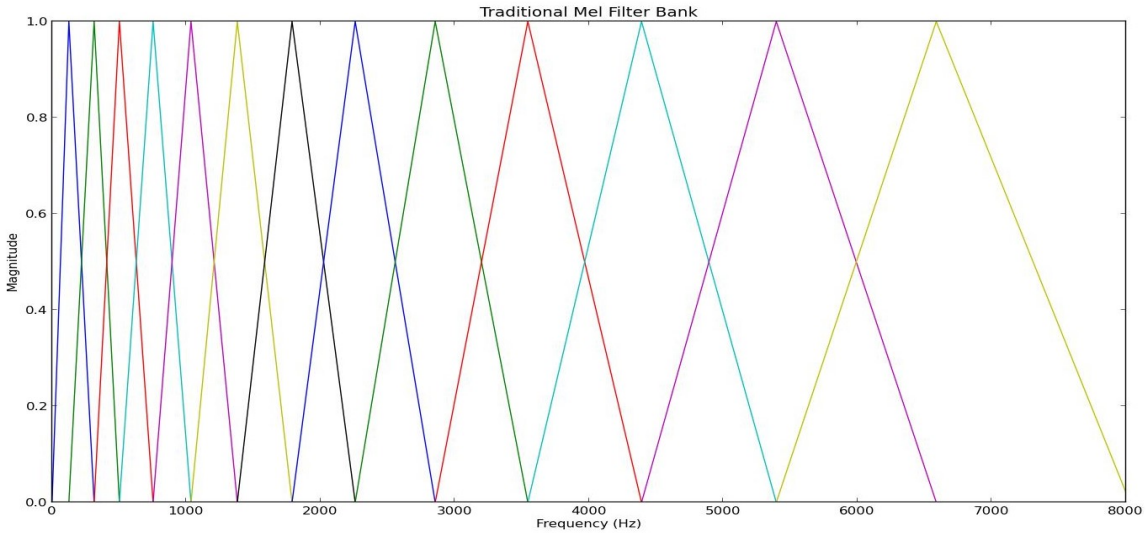

Figura 2.1: Banco de filtros para conversión a escala Mel

Dicho banco de filtros se aplica después de calcular el módulo de la transformada de Fourier de $x[n]$ y antes de realizar el logaritmo neperiano, tal y como se muestra en la Figura 2.2.

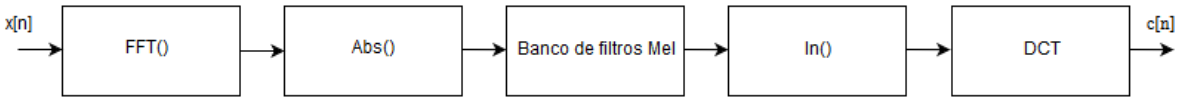

Figura 2.2: Diagrama de bloques para la extracción de características

Finalmente, tras calcular el logaritmo neperiano se aplica la transformada de coseno discreta (DCT o *Discrete Cosine Transform*). Se utiliza la DCT en lugar de aplicar simplemente la transformada inversa de Fourier debido a sus propiedades de compresión y, sobre todo, de decorrelación de la información. Esto hace que, en la mayoría de los sistemas de RAH, sea suficiente utilizar 12 coeficientes MFCC. No obstante, para tener en cuenta las variaciones temporales que se puedan producir en la señal de voz, se suelen tomar también la primera y segunda derivada de los parámetros cepstrales para la representación dinámica de la señal, aportando mas información a largo plazo. Del mismo modo, también resulta útil la información acerca de la energía de la señal, que permite al sistema diferenciar sonidos sordos (menor energía) de sonidos sonoros (mayor energía). Esto hace que el total de parámetros comúnmente utilizados ascienda a 39 MFCC.

2.4. Modelos Ocultos de Markov

Sea $\mathbf{Z} = [Z_1, Z_2, ..., Z_n]$ una secuencia de variables aleatorias, a partir del teorema de Bayes expuesto en (2.2) obtenemos:

$$P(Z_1, Z_2, ...Z_n) = P(Z_1) \prod_{i=2}^{n} P(Z_i | Z_1^{i-1}) \tag{2.8}$$

que se corresponde con la probabilidad conjunta de observar la secuencia $\mathbf{Z}$ y donde $Z_1^{i-1} = Z_1, Z_2, ...Z_{i-1}$. Las variables aleatorias Z forman una cadena de Markov de orden 1 cuando:

$$P(Z_i|Z_1^{i-1}) = P(Z_i|Z_{i-1}) \tag{2.9}$$

La ecuación (2.9) es conocida como la Hipótesis de Markov, según la cuál la probabilidad de una variable aleatoria en un instante depende sólo de la variable aleatoria en el instante anterior. Así, podemos rescribir la ecuación (2.8) de la siguiente forma:

$$P(Z_1, Z_2, ...Z_n) = P(Z_1)\prod_{i=2}^{n} P(Z_i|Z_{i-1}) \tag{2.10}$$

Si descartamos el índice de tiempos i podemos modelar eventos estacionarios como:

$$P(Z_i = s|Z_{i-1} = s') = P(s|s') \tag{2.11}$$

Asociando un estado s a Z_i, la cadena de Markov se puede representar mediante una máquina de estados finitos estocástica, con transiciones entre estados correspondientes a la probabilidad $P(s|s')$. Teniendo esto en cuenta, de (2.10) podemos inferir que la probabilidad de que la cadena de Markov esté en un estado específico en un instante determinado depende únicamente del estado en el instante anterior.

Dada una cadena de Markov de N estados y sea s_t el estado de la cadena de Markov en el instante t, los parámetros que definen esta cadena son:

$$a_{ij} = P(s_t = j|s_{t-1} = i) \quad 1 \leq i, j \leq N \tag{2.12}$$

$$\pi_i = P(s_1 = i) \quad 1 \leq i \leq N \tag{2.13}$$

donde a_{ij} es la probabilidad de transición entre el estado i y el estado j, y π_i la probabilidad de que la cadena de Markov comience en el estado i. Dichas probabilidades de transición y de inicialización verifican las siguientes restricciones:

$$\sum_{j=1}^{N} a_{ij} = 1 \quad 1 \leq i \leq N \tag{2.14}$$

$$\sum_{j=1}^{N} \pi_j = 1 \tag{2.15}$$

Con esta descripción podemos denominar esta cadena de Markov como Modelo Observable de Markov, ya que la salida del proceso es el conjunto de estados en cada instante de tiempo t, donde cada estado se corresponde con un evento que podemos observar Z_i, por lo que hay una correspondencia uno a uno entre la secuencia de eventos observables y la secuencia de estados de la cadena de Markov.

Ahora supongamos una cadena de Markov donde la observación ya no es un valor determinista sino una variable aleatoria X generada según una función de densidad de probabilidad (pdf o *probability density function*) asociada a cada estado. Ya no podemos encontrar una correspondencia unívoca entre la secuencia de salida y la secuencia de estados. Por eso se dice que la secuencia de estados está oculta (*hidden* en inglés). Estos modelos se denominan Modelos Ocultos de Markov o HMM (*Hidden Markov Models*).

La forma más habitual de modelar la producción del habla es mediante el uso de HMM. El sistema debe estimar qué serie de estados produjeron la secuencia de medidas acústicas observada. Formalmente, un HMM queda definido por una serie de parámetros:

- $S = [1, 2, ..., N]$: conjunto de estados que caracterizan el modelo.
- $A = \{a_{ij}\}$: probabilidad de transición entre estados, donde a_{ij} representa la probabilidad de transitar del estado i al estado j.
- $B = \{b_i(x)\}$: conjunto de funciones de densidad de probabilidad, donde $b_i(x)$ es la *pdf* de la observación en el estado i.
- $\pi = \{\pi_i\}$: probabilidad de estado inicial, donde π_i es la probabilidad de estar en el estado i en el instante $t = 1$.

Ya que a_{ij} y π_i son probabilidades, deben satisfacer las siguientes propiedades:

$$a_{ij} \geq 0 \quad b_i(k) \geq 0 \quad \pi_i \geq 0 \quad \forall i, j, k \tag{2.16}$$

$$\sum_{j=1}^{N} a_{ij} = 1 \quad \sum_{i=1}^{N} \pi_i = 1 \tag{2.17}$$

Algo similar ocurre con $b_i(x)$ por tratarse de una función de densidad de probabilidad

$$\int_{-\infty}^{\infty} b_i(x)dx = 1 \quad \forall i \tag{2.18}$$

Por tanto, un HMM queda caracterizado por dos matrices de probabilidad, A y π, y un conjunto de funciones de densidad de probabilidad B. Es necesaria también una constante N que caracteriza el número de estados. La notación que se utilizará para referirnos a un HMM es la especificada en (2.19)

$$\Phi = (A, B, \pi) \tag{2.19}$$

Para la obtención de este modelo hemos tenido en cuenta dos asunciones:

- La hipotesis de Markov para las cadenas de Markov:

  $$P(s_t|s_1^{t-1}) = P(s_t|s_{t-1}) \tag{2.20}$$

  donde s_1^{t-1} representa la secuencia de estados $s_1, s_2, ...s_{t-1}$. Esta asunción implica que la probabilidad de encontrarse en el estado s en un instante de tiempo t sólo depende del estado en el que nos encontrábamos en el estado $t - 1$.

- Independencia de las observaciones de salida anteriores:

  $$P(X_t|X_1^{t-1}, s_1^t) = P(X_t|s_t) \tag{2.21}$$

  donde X_1^{t-1} es la secuencia de salida $X_1, X_2, ..., X_{t-1}$. Esto significa que la probabilidad de emitir un determinado símbolo en un instante de tiempo t depende únicamente del estado s_t y es independiente de las observaciones anteriores.

Estas dos propiedades conllevan una simplificación a la hora de modelar la producción del habla en el RAH. No obstante, en la práctica los HMM resultan adecuados y estas simplificaciones hacen que trabajar con ellos sea más simple, reduciendo en gran medida el número de parámetros que se necesitan estimar.

Dado un HMM $\Phi = (A, B, \pi)$, en el RAH existen tres grandes problemas a los que debemos hacer frente:

1. **Evaluación**: dado un modelo Φ y una secuencia de observaciones $\mathbf{X} = (X_1, X_2, ..., X_T)$ el objetivo es calcular la probabilidad $P(\mathbf{X}|\Phi)$, de obtener la secuencia de observaciones $\mathbf{X}$ dado el modelo Φ. Este problema se suele resolver utilizando el algoritmo *Forward* [3], una forma eficiente de calcular la suma de las probabilidades de todas las posibles secuencias de estados.

2. **Decodificación**: dado un modelo Φ y una secuencia de observaciones $\mathbf{X} = (X_1, X_2, ..., X_T)$ el objetivo es obtener la secuencia de estados $\mathbf{S} = (s_1, s_2, ..., s_T)$ que más se ajusta a las observaciones. Uno de los algoritmos más utilizados para resolver este problema es el algoritmo de Viterbi. Esta técnica busca el camino óptimo en términos de verosimilitud. Se calcula la secuencia de estados que maximiza la probabilidad hasta un instante t y se obtiene el estado siguiente como aquél que da el mejor camino entre el último estado y el conjunto de posibles estados a los que se puede transitar en el estado $t + 1$. Una vez alcanzado el final, el camino definitivo se obtiene mediante un recorrido hacia atrás en la red de decodificación construida. Durante este proceso se suelen aplicar técnicas de poda de caminos donde, según el valor del parámetro del haz de búsqueda o *Beam Search*, se van descartando aquellos caminos cuya verosimilitud acumulada se encuentra por debajo de un umbral, normalmente relativo al camino con máxima verosimilitud.

3. **Estimación**: dado un modelo Φ y un conjunto de observaciones de entrenamiento $\mathbf{X}$ se pretende estimar los parámetros del modelo $\hat{\Phi}$ que maximicen la probabilidad conjunta $P(\mathbf{X}|\Phi)$. Este problema es el más complejo de los expuestos ya que no existe un método analítico que maximice la probabilidad conjunta de los datos de entrenamiento. Un algoritmo que se utiliza habitualmente es el algoritmo de *Baum-Welch* o algoritmo *forward-backward* [3]. Se trata de una técnica iterativa que reestima los parámetros en cada iteración, de forma que $P(\mathbf{X}|\Phi)$ mejora en cada paso. El algoritmo continua hasta que la diferencia de probabilidades entre un paso y el anterior sea lo suficientemente pequeña para considerarse como modelo óptimo. Para esta tarea se necesitan datos de entrenamiento correctamente etiquetados.

De los modelos que utilizan los sistemas de RAH, el MA y el ML, los HMM se utilizan para generar el MA.

2.5. Modelado Acústico

El Modelado Acústico representa la información que tiene el sistema de RAH asociada a las características acústicas de la señal de audio. Tiene como misión determinar la probabilidad $\mathbf{P}(\mathbf{X}|\mathbf{W})$ de una secuencia de observaciones $\mathbf{X}$ dada una secuencia de palabras $\mathbf{W}$. Para ello se utilizan técnicas de aprendizaje que hacen uso de grandes bases de datos de audio. Los HMM constituyen, a día de hoy, la solución mas extendida en términos de MA en los sistemas de RAH. Tal y como se explicó en la sección 2.4, se trata de una máquina de estados finitos a cada uno de los cuales se asigna una *pdf* de salida, que normalmente se suele modelar como una mezcla de Gaussianas (GMM o *Gaussian Mixture Model*) [4]. De forma habitual se considera el fonema (unidad básica del lenguaje que resulta de la abstracción o descripción teórica de los sonidos) como la unidad acústica que el sistema de RAH utilizará para su funcionamiento, y se suele modelar con una secuencia de tres estados que indican el comienzo, el centro y el fin del fonema, resultando el HMM de la Figura 2.3.

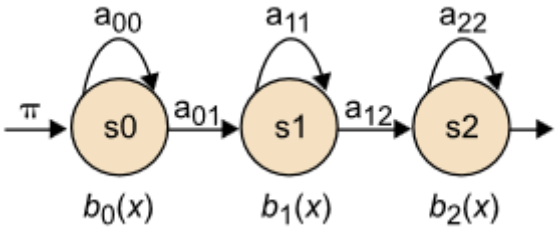

Figura 2.3: HMM para el modelado de un trifonema

2.6. Modelado del Lenguaje

El ML o gramática hace referencia al conocimiento que tiene el sistema sobre la formación de frases con las palabras disponibles en su vocabulario. Indica qué palabras y en qué orden se espera que un usuario las pronuncie, limitando así la búsqueda únicamente a las palabras permitidas en la gramática. Existen fundamentalmente dos tipos de ML: los Modelos de Lenguaje Estocásticos (MLE) y los Modelos de Lenguaje de Gramáticas de Reglas.

2.6.1. Modelos de Lenguaje Estocásticos

Los MLE modelan el lenguaje desde una perspectiva probabilística. Pretenden estimar de forma precisa la probabilidad $P(\mathbf{W})$ para una secuencia de palabras dada $\mathbf{W} = [w_1, w_2, ...w_n]$. Esto, además de mejorar la precisión de los sistemas de RAH, ayuda a reducir el espacio de búsqueda en la decodificación llevada a cabo por el reconocedor.

Sea $P(\mathbf{W})$ la probabilidad de ocurrencia de una secuencia de N palabras, se puede descomponer en producto de probabilidades condicionales según (2.22)

$$P(\mathbf{W}) = \prod_{i=1}^{N} P(w_i|h_i) \tag{2.22}$$

Donde $h_i = w_1^{i-1}$ representa la historia previa a la palabra w_i y corresponde a la secuencia de palabras $w_1 w_2 ... w_{i-1}$. Estos modelos se suelen representar mediante cadenas de Markov de orden 1. Según lo expuesto en la sección 2.4, la probabilidad de un estado sólo depende del estado anterior, es decir de la palabra anterior. Sin embargo, podríamos considerar mayores dependencias si asociamos los estados a la historia h_i de la palabra w_i. En función de la memoria que consideremos podemos encontrarnos con diferentes modelos:

- $h_i = w_{i-1}$: modelo de bigramas de palabras, donde la probabilidad de que una palabra tenga lugar depende sólo de la palabra anterior, es decir $P(w_i|w_{i-1})$.
- $h_i = w_{i-1}w_{i-2}$: modelo de trigramas de palabras, en el que la probabilidad de que una palabra tenga lugar depende de las dos palabras precedentes, es decir $P(w_i|w_{i-1}w_{i-2})$.
- $h_i = w_{i-1}w_{i-2}...w_{i-n+1}$: modelos de n-gramas de palabras, donde la probabilidad de observar una palabra dependerá de las $n-1$ palabras anteriores.

Actualmente los MLE basados en n-gramas son los más utilizados en RAH. Modelan el lenguaje como una fuente de Markov de orden $n-1$ condicionando la probabilidad de observar una palabra a las $n-1$ anteriores. Para generar estos modelos se utilizan grandes bases de datos de texto de entrenamiento. Si utilizamos un conjunto de datos demasiado pequeño el modelo no podrá ver todas las posibles combinaciones de palabras, asignándo a los eventos

no vistos una probabilidad muy pequeña o incluso nula. Esto supone un problema, ya que si $P(W) = 0$ el reconocedor nunca considerará W como una posible transcripción. Para asegurar que todas las secuencias tienen una probabilidad de ocurrencia distinta de cero se utilizan diferentes técnicas conocidas como suavizado o *smoothing* [3].

2.6.2. Modelos de Lenguaje de Gramáticas de Reglas

Los ML basados en gramáticas de reglas basan su funcionamiento en el conocimiento de la estructura sintáctica de las sentencias de una lengua e intentan modelarla. Son apropiados para sistemas con un dominio semántico restringido, en los que tanto el vocabulario como la estructura semántica de las frases están limitadas. Las restricciones jerárquicas que existen entre las palabras permitidas por el sistema pueden especificarse utilizando distintos formatos. Concretamente destacaremos JSGF (*Java Speech Grammar Format*) [5], que serán los ML utilizados por nuestro sistema de RAH.

Linea	Contenido
1	# JSGF V1.0 ISO8859-1 en;
2	grammar com.foo.commands
3	
4	public <basicCmd> = <startPolite> <command>;
5	
6	<startPolite> = (please | kindly | could you);
7	
8	<command> = <action> <object>;
9	<action> = /10/ open | /2/ close | /1/ delete | /1/ move;
10	<object> = [the | a] (window | file | menu);

Tabla 2.1: Descripción de gramáticas JSGF

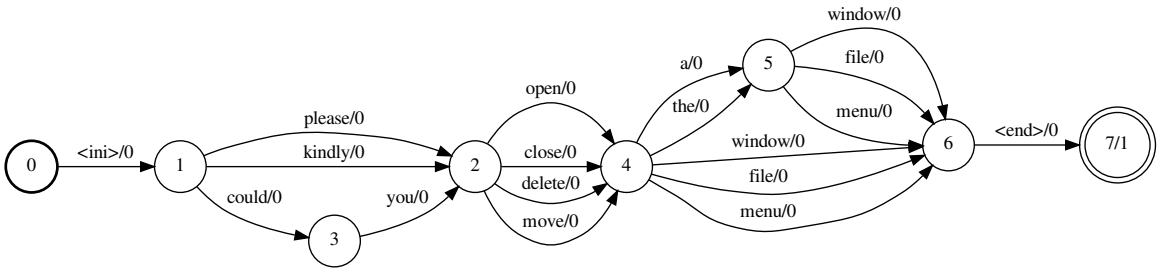

Figura 2.4: Máquina de estados finitos de la gramática *commands*

JSGF define un conjunto de operadores que permiten construir modelos gramaticales. Supongamos un contexto simple de manejo de una interfaz de ventanas mediante comandos de voz, cuya descripción de gramáticas viene dada en la Tabla 2.1. Vemos que se define una única regla pública (<basicCmd>), que a su vez se compone de otras reglas privadas: la fórmula de cortesía de inicio (<startPolite>) y la definición del comando (<command>), que viene dado a su vez por la acción (<action>) a realizar y el objeto (<object>). Las reglas de acciones y objetos son conjuntos de posibles alternativas que el usuario puede pronunciar, además la regla de acciones tiene un conjunto de pesos que indica que el comando "*open*" es más probable que los demás. Por tanto, la regla <command> define una secuencia de palabras en la que

una acción puede ir precedida de forma opcional por un artículo ("*a*" o "*the*"), pero a la que siempre seguirá un objeto. Por lo tanto, teniendo en cuenta la gramática definida, algunas de las posibles secuencias que el usuario podría pronunciar son: "*open a window*", "*close file*", "*could you open the menu*", "*please delete the file*". Otra forma de observar la gramática de una forma más visual es mediante una máquina de estados finitos[6], como la mostrada en la Figura 2.4, correspondiente a la gramática *commands*.

2.7. Errores en RAH

En los sistemas de RAH se consideran tres tipos diferentes de errores:

- **Sustituciones**(S): el sistema da como salida una palabra distinta de la palabra correcta.
- **Borrados**(B): el sistema omite una palabra correcta en la frase reconocida.
- **Inserciones**(I): el sistema añade una palabra extra en la frase reconocida.

Teniendo en cuenta estos tres tipos de errores, una medida que se suele utilizar para evaluar las prestaciones de un sistema de RAH es el *Word Error Rate* (WER).

$$WER(\,\%) = \frac{S + B + I}{M} \times 100 \tag{2.23}$$

donde M representa el número total de palabras en la frase correcta. En general, para calcular el WER primero es necesario alinear la secuencia de palabras reconocidas con la secuencia de palabras correcta. No obstante, dicho alineamiento no resulta tan sencillo puesto que realizar una correspondencia directa entre palabras de la frase correcta y palabras de la frase reconocida no es una tarea trivial, tal y como se ilustra en la Tabla 2.2. Supongamos la frase correcta "son coches de juguete" y la frase reconocida "coches de juguete rojos". *A priori*, si

Frase correcta	Frase reconocida
son	coches
coches	de
de	juguete
juguete	rojos

Tabla 2.2: Correspondencia entre frase correcta y frase reconocida

comparamos palabra por palabra la tasa de error es del 75 % con tres sustituciones ("coches" por "son", "de" por "coches" y "juguete" por "de"). Sin embargo, si alineamos ambos textos correctamente y calculamos su WER vemos que es del 50 %, con un borrado("son") y una inserción ("rojos").

Generalmente, la tarea de evaluación en sistemas de RAH se llevan a cabo mediante el uso de algoritmos de programación dinámica, normalmente integrados en *toolkits* de *scoring* como las herramientas software SCLITE (*Score Lite*) del NIST (*National Institute of Standards and Technology*) [7].

Capítulo 3

Tarea de alineamiento del *MGB Challenge*

En capítulo se describirá detalladamente la tarea de alineamiento propuesta en la evaluación del *MGB Challenge* [8], sobre la cual se sustenta este trabajo. Se describirán los datos proporcionados para la tarea, los elementos que la componen y las métricas utilizadas para evaluar los resultados.

3.1. Introducción

El *MGB Challenge* es una evaluación de reconocimiento del habla, diarización del locutor y alineamiento, utilizando grabaciones de televisión de la *British Broadcasting Corporation* (BBC) [2]. Para la tarea de alineamiento, los participantes reciben un fichero de audio y los subtítulos tal y como fueron desarrollados en los sistemas de producción de la BBC, o en ocasiones el guión del programa, una fase previa a los subtítulos. La tarea consiste en identificar los tiempos precisos de cada palabra pronunciada en la locución en entornos acústicos muy variados. Los subtítulos han de tomarse como una aproximación a esta tarea, ya que pueden no reflejar exactamente las palabras locutadas, bien sea por fallos en la transcripción o por que se ha realizado un resumen para que la representación en pantalla sea adecuada. Por tanto, los participantes deben identificar aquellas palabras que realmente fueron dichas además de determinar cuándo.

3.2. Terminología utilizada

Para describir adecuadamente la tarea de alineamiento es necesario introducir la siguiente terminología:

- ***Script***: el *script* es el conjunto de transcripciones que se entrega a los participantes de la tarea de alineamiento. Las palabras que aparecen en él pueden no reflejar completamente las palabras de la locución, y es posible que contenga errores de transcripción. Este *script* no contiene ninguna información de marcas temporales, ni información sobre la identidad del locutor. Estos ficheros se encuentran disponibles únicamente en formato normalizado, procesado que se aplica a los subtítulos originales en números y caracteres especiales para mantener la consistencia a lo largo de la base de datos.

- **Referencia**: contiene una transcripción precisa con tiempos de inicio y final para cada palabra, basada en transcripciones manuales de los audios, y utilizada como alineamiento perfecto contra el que contrastar la hipótesis.

- **Hipótesis**: es el resultado del sistema automático de alineamiento con el que se experimenta en este TFG. La salida contiene las palabras del *script* encontradas en el audio y su localización precisa en el tiempo. También puede eliminar palabras del *script* que no hayan sido locutadas.

3.3. Consideraciones a la tarea de alineamiento

En términos generales, la tarea de alineamiento requiere encontrar de forma precisa el tiempo de inicio y el tiempo de final de cada palabra locutada en el audio. En la práctica, las palabras que aparecen en el *script* no tienen una correspondencia directa con la locución. Podemos encontrar dos casos posibles:

1. Las palabras que aparecen en el *script* no fueron realmente locutadas.
2. Las palabras que aparecen en la referencia no aparecen en el *script*.

Según como tratemos estos dos casos podemos distinguir dos tipos de alineamiento:

- **Alineamiento restringido al *script***: las palabras que no fueron locutadas pero aparecen en el *script* son eliminadas de la salida del sistema, mientras que las palabras que aparecen solo en la referencia no son añadidas a la salida.
- **Alineamiento apoyado en el *script***: la diferencia es que las palabras que aparecen únicamente en la referencia pueden ser añadidas al *script*. De esta forma la tarea a desarollar es similar a una tarea de transcripción.

La tarea de alineamiento del *MGB Challenge* es de alineamiento restringido al *script*. Por tanto, no será necesario preocuparse de las palabras que aparecen únicamente en la referencia.

Cabe destacar que las secciones de audio donde dos o más locutores hablan al mismo tiempo serán ignoradas en la evaluación. Se eliminarán todas las regiones donde cualquier fragmento de voz se superponga, al menos parcialmente. No obstante, se ha de procesar el fichero de audio y el *script* en su totalidad, teniendo en cuenta los fragmentos que se solapen solo en el momento de calcular los resultados de alineamiento del sistema.

3.4. Métricas de error

Una vez obtenida la salida del sistema de alineamiento es necesario evaluar los resultados obtenidos. Con este objetivo, el *MGB Challenge* define tres métricas diferentes que se explican a continuación:

$$Precision = \frac{N_{match}}{N_{hyp}} \tag{3.1}$$

$$Recall = \frac{N_{match}}{N_{ref}} \tag{3.2}$$

$$F = 2\frac{Precision \times Recall}{Precision + Recall} \tag{3.3}$$

donde N_{match} representa el número de palabras de la hipótesis que han sido correctamente alineadas, N_{hyp} el número de palabras totales de la hipótesis y N_{ref} el número de palabras totales de la referencia.

La ecuación (3.1) es una medida de la precisión de nuestro sistema. Nos informa de cuántas palabras han sido alineadas correctamente respecto del total de las palabras alineadas en la hipótesis. La ecuación (3.2), en cambio, nos da información acerca de cuántas palabras ha sido capaz de alinear nuestro sistema, comparando el numero de palabras correctamente alineadas de la hipótesis con las palabras de la referencia. La medida F se obtiene combinando las ecuaciones (3.1) y (3.2) y representa una medida global de la calidad del sistema de alineamiento que se utilizará para comparar con los sistemas diseñados por otros participantes en el *MGB Challenge*. Todas las métricas pueden tomar valores entre 0 y 1, ambos inclusive.

Es necesario también definir cuándo consideramos que una palabra ha sido correctamente alineada y cuándo no. En este tipo de evaluaciones es habitual considerar unos márgenes de guarda respecto a la referencia por diferentes motivos. Por un lado, los tiempos establecidos en la referencia pueden no ser del todo precisos ya que están sujetos en última instancia a la precisión de la revisión realizada por humanos. Por otro lado, es razonable permitir una tolerancia alrededor de los margenes establecidos por la referencia, de modo que palabras que en la hipótesis se hayan situado muy próximas a sus valores temporales de referencia, sigan siendo consideradas aciertos del sistema de alineamiento.

De este modo, la evaluación del *MGB Challenge* considera que una palabra está correctamente alineada cuando el tiempo de inicio de la palabra de la hipótesis se encuentra dentro de la ventana temporal fijada por el tiempo de inicio de la palabra de la referencia ± 100ms y cuando el tiempo final de la palabra de la hipótesis se encuentra dentro de la ventana temporal fijada por el tiempo final de la palabra de la referencia ± 100ms. En la Figura 3.1 se ejemplifica este método para una palabra alineada correctamente.

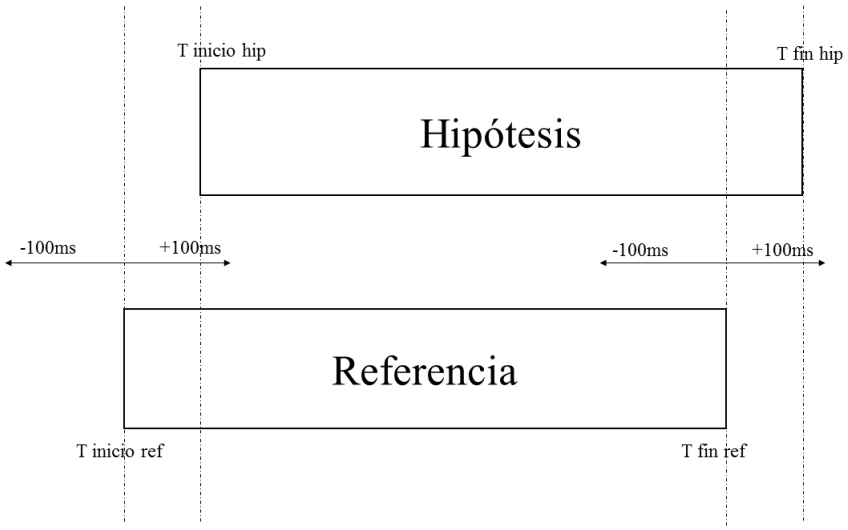

Figura 3.1: Representación del método de evaluación de palabras correctas

3.5. Herramienta de evaluación

Para facilitar la obtención de las métricas de error y que los resultados de todos los participantes estén normalizados, la organización del *MGB Challenge* proporciona una herramienta de evaluación. Esta herramienta alinea la hipótesis de salida del sistema de alineamiento frente a la referencia, elimina los fragmentos de solape y calcula las palabras correctamente alineadas. Finalmente muestra por pantalla las métricas de error obtenidas, tal y como puede observarse en la Figura 3.2.

```
================================================================================
Alignment scoring results
================================================================================
Date                          : Monday, 25. April 2016 06:28PM
Scoring script version        : v0.3
--------------------------------------------------------------------------------
Reference script              : mgb-challenge-2015/20080522_190000_bbctwo_jonathan_meades_magnetic_north_files/.... .stm
Reference alignment           : mgb-challenge-2015/20080522_190000_bbctwo_jonathan_meades_magnetic_north_files/.... .ctm
System output                 : out_ctm/20080522_190000_bbctwo_jonathan_meades_magnetic_north/140000/myFULL.ctm
Filtering UEM                 : mgb-challenge-2015/20080522_190000_bbctwo_jonathan_meades_magnetic_north_files/.... .uem
--------------------------------------------------------------------------------
... reading reference
... aligning script to reference
... reading uem file - mgb-challenge-2015/20080522_190000_bbctwo_jonathan_meades_magnetic_north_files/overlap.uem
... reading system output
... aligning script to system output
... reading uem file - mgb-challenge-2015/20080522_190000_bbctwo_jonathan_meades_magnetic_north_files/overlap.uem
... matching
--------------------------------------------------------------------------------
20080522_190000_bbctwo_jonathan_meades_magnetic_north #ref 3329/3503 #hyp 3395/3402 match 3395/3402 prec 0.7973 rec 0.8132 f 0.8052
--------------------------------------------------------------------------------
Final results
--------------------------------------------------------------------------------
Effective #words in reference  : 3329
Effective #words in hypothesis : 3395
Match count                    : 2707
--------------------------------------------------------------------------------
Precision                      : 0.7973
Recall                         : 0.8132
F                              : 0.8052
================================================================================
```

Figura 3.2: Salida de la herramienta de evaluación del sistema de alineamiento

3.6. Base de datos

El *MGB Challenge* trabaja con programas de televisión pertenecientes a la BBC. La base de datos completa se compone de aproximadamente 1600 horas de audio grabadas durante siete semanas. Está compuesta por programas de televisión de muy diversos géneros. Podemos encontrar desde series hasta concursos, pasando por documentales o dibujos animados. Esta gran variedad supone un reto importante a la hora de llevar a cabo la tarea de alineamiento. En un documental, por ejemplo, no suele haber artefactos acústicos superpuestos al locutor y el ritmo del discurso suele ser constante y sin grandes cambios de amplitud. En cambio, en un programa de dibujos animados, el habla es mucho mas espontanea y resulta más probable encontrar fragmentos de música superpuestos al discurso.

Para la tarea de alineamiento se utilizan tres conjuntos o *sets* de datos que se describen en la Tabla 3.1.

Conjunto de datos	**Nº programas**	**Duración total (Horas)**
train.full	2193	1580
dev.full	47	28
eval.task1	16	11

Tabla 3.1: Base de datos del *MGB Challenge*

Cada uno de estos conjuntos ha sido diseñado para un objetivo concreto:

- ***train.full***: este conjunto se usa para entrenar los modelos acústicos y de lenguaje del sistema de RAH.
- ***dev.full***: usado durante la fase de desarrollo del sistema de alineamiento para ajustar los parámetros en las distintas estrategias seguidas.
- ***eval.task1***: conjunto de datos con los que se realiza la evaluación final del sistema de alineamiento optimizado con *dev.full*.

No obstante dado el gran volumen de datos del *set dev.full*, y ya que los datos del *eval.task1* son suficientemente representativos para llevar a cabo las tareas planificadas en este TFG, la fase de experimentación tan solo se realizará sobre el último conjunto de datos.

Además de los ficheros de audio, los participantes reciben un conjunto de metadatos que acompañan a dichos ficheros. Estos metadatos, tal y como son proporcionados por la BBC en un primer momento,contienen los subtítulos del programa correspondiente. Tras un proceso de normalización se generan los *scripts* que se usarán como punto de partida para la tarea de alineamiento.[9]

Capítulo 4

Alineamiento con el sistema base

En este capítulo se detalla el sistema de alineamiento desarrollado y evaluado durante este TFG. Se describen todos los bloques que lo componen, y se presentan los resultados obtenidos durante su evaluación y su posterior análisis.

4.1. Introducción

En términos generales, el sistema de alineamiento puede describirse como un módulo que, partiendo de un texto sin marcas temporales y su audio correspondiente, los alinea a nivel de palabra, generando un fichero de texto con marcas temporales, tal y como se ilustra en la Figura 4.1.

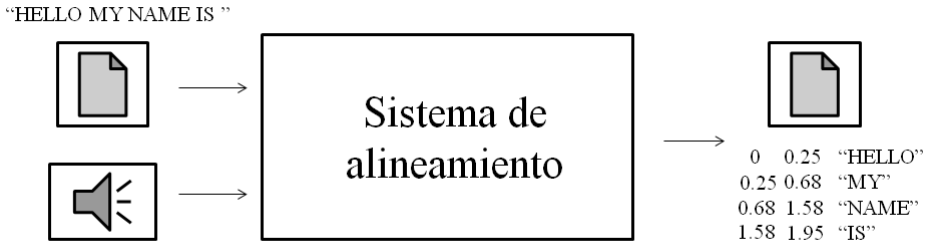

Figura 4.1: Entradas y salidas del sistema de alineamiento

Para llevar a cabo la tarea de alineamiento, se considera un diagrama de bloques como el detallado en la Figura 4.2, en el que destacan cuatro bloques principales: el segmentador, la inserción de modelo de relleno, el reconocedor automático del habla y un bloque de composición. Existe además un quinto bloque correspondiente a la evaluación de las prestaciones y de cálculo de las métricas de error.

4.2. Descripción de los componentes del sistema

4.2.1. Segmentador

Se trata del bloque de entrada del sistema de alineamiento. Realiza una primera adaptación de la información del *script* y el audio a la tarea de alineamiento. Recibe como entrada la

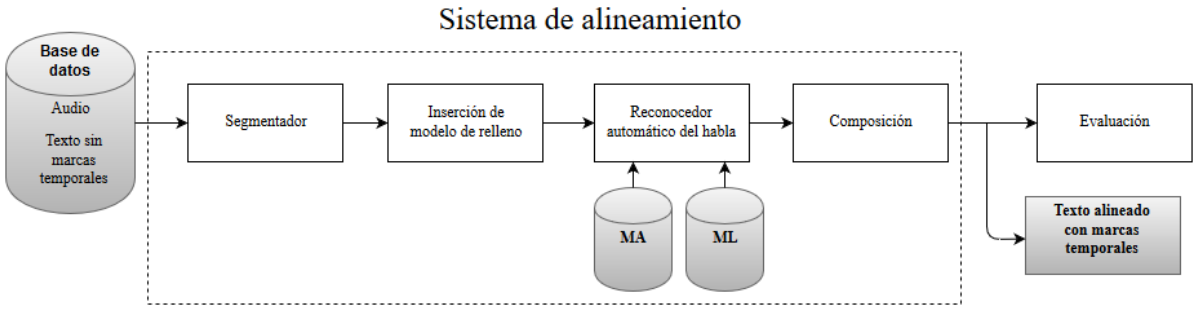

Figura 4.2: Diagrama de bloques del sistema de alineamiento

información del *script* del programa en forma de texto y el audio de dicho programa, previamente procesado para la extracción de características, obteniendo así sus coeficientes MFCC. A esta información de audio parametrizada se le denominará de ahora en adelante fichero `.prm`.

Como ya indica su nombre, el bloque de segmentación se encarga de realizar una división de la información en bloques más pequeños que denominaremos *chunks*. Es decir, dada la información de audio y texto de un programa, obtiene N subconjuntos de dicha información que representan a los *chunks* desde el 1 hasta el N. Adoptar este tipo de estrategias facilita la tarea del reconocedor automático del habla, ya que un fragmento demasiado grande de audio en el reconocedor puede hacer que se pierda en el algoritmo de decodificación y no encuentre el camino óptimo, anulando así su salida. Utilizando esta estrategia, el reconocedor recibirá solo la información para alinear un *chunk* cada vez. Los parámetros que rigen este bloque de segmentación son:

- **Tiempos de guarda**: se utilizan para paliar el problema de la ambigüedad en la localización del audio correspondiente a cada texto. A priori no se tiene información ninguna sobre qué fragmento de texto corresponde con cada fragmento de audio. El hecho de hacer los fragmentos de audio más grandes añadiendo tiempo de guarda por delante y por detrás hace que la probabilidad de encontrar el texto en el audio aumente. La herramienta es capaz de admitir diferentes tiempos de guarda al inicio y al final de cada fragmento de audio, tal y como se ilustra en la Figura 4.3.

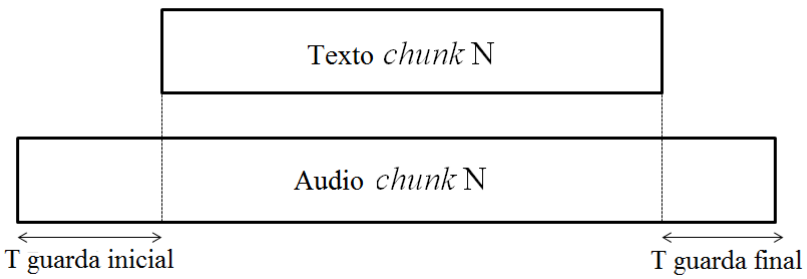

Figura 4.3: Tiempo de guarda en la herramienta de segmentación

- **Tamaño de los *chunks***: la longitud de los *chunks* es configurable. Dependiendo de como se reparta el texto del *script* entre los N *chunks* que componen el programa, podemos generar fragmentos más grandes o más pequeños.

4.2.2. Inserción de modelo de relleno

Para comprender el porqué del uso de este bloque en el sistema de alineamiento es necesario retomar los ML de gramáticas de reglas expuestos en la sección 2.6.2. Si nos centramos en el objetivo de este TFG, el contexto gramatical es bastante distinto del ejemplo de control de una interfaz de ventanas. La tarea de alineamiento requiere localizar una secuencia de texto conocida *a priori* dentro del audio. Por tanto, la gramática a definir se basará en el texto del guión del programa de televisión del que se dispone. Supongamos un fragmento de guión dado por la siguiente frase "*We do everything*". En la Tabla 4.1 se muestra la gramática JSGF creada para la tarea de alineamiento del fragmento de guión considerado.

Linea	Contenido
1	# JSGF V1.0 ISO8859-1 en;
2	grammar fsg.rulegrammar
3	
4	public <rulegrammar> = <fon>* <1> <2> <3> <fon>*;
5	
6	<1> = (WE) {0};
7	<2> = (DO) {1};
8	<3> = (EVERYTHING) {2};
9	<fon> = ph1 | ph2 | ph3 | ... | phM;

Tabla 4.1: Descripción de gramáticas JSGF para la tarea de alineamiento del *MGB Challenge*

Vemos que la regla pública <rulegrammar> está definida por la secuencia de palabras que componen la frase extraída del guión. La regla <fon> constituye una red de fonemas. Se trata de una implementación de un modelo de relleno. Los modelos de relleno se utilizan en sistemas de RAH para modelar todo aquello que esté fuera de su rango de vocabulario, ayudando así al reconocedor a ignorar todo lo que no sea voz (música, artefactos acústicos) [10]. Estas redes se colocan al principio y al final de la secuencia de palabras para que el reconocedor sea capaz de engancharse con el comienzo del audio en el momento correcto. Las gramáticas de este tipo podrían llegar a ser excesivamente estrictas si la secuencia que intentamos alinear es demasiado larga. Resulta muy probable que encontremos música o algún otro tipo de artefactos acústicos en el audio que hagan que el sistema de RAH se pierda y su salida no sea correcta. Por ello, en los experimentos a realizar incluiremos redes de fonemas intermedias que sirvan de escape al reconocedor y le ayuden a volver a engancharse con el audio. El operador "*" indica que podemos encontrar esta regla ninguna, una o más veces. La máquina de estados finitos [6] que define esta gramática se muestra en la Figura 4.4.

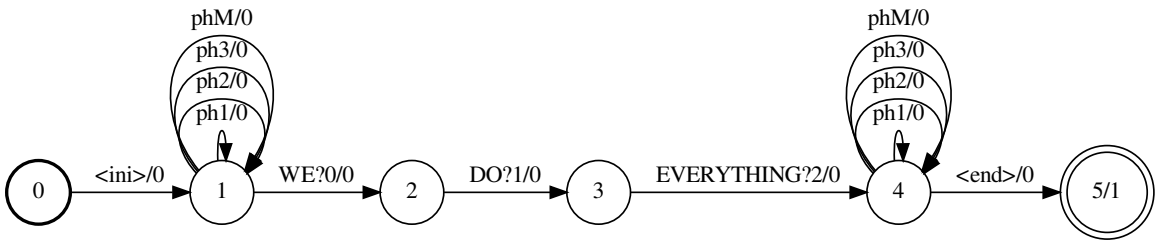

Figura 4.4: Máquina de estados finitos de la gramática *rulegrammar*

Así, para cada uno de los N *chunks* obtenidos a la salida del bloque de segmentación, se añaden redes de fonemas intermedias con el objetivo de suavizar las restricciones de una

gramática excesivamente estricta, facilitando la tarea de alineamiento en secuencias muy largas. El funcionamiento de este bloque es simple *a priori*. Inserta caracteres especiales ("+" o "." por ejemplo) entre las palabras del texto de cada *chunk*, según se le indique mediante un parámetro de entada. Así, el reconocedor en la siguiente etapa interpreta estos caracteres, los elimina y genera la gramática con redes de fonemas intermedias en las posiciones indicadas. Cabe destacar también que, por defecto, el sistema incluirá una red de fonemas al principio y al final de cada *chunk*.

Por tanto, en la fase de experimentación distinguiremos dos modos de operación de este bloque: uno en el cual se incorporan redes de fonemas intermedias al final de cada intervención, y otro en el cual se evalúa la efectividad de incorporar redes de fonemas intermedias cada N palabras.

4.2.3. Reconocedor automático del habla

Es el elemento principal del sistema de alineamiento. Su función es la de alinear el texto del *script* con el audio del programa. Este módulo ha sido desarrollado por ViVoLab y utilizado en otros proyectos de similares características, como el subtitulado de noticias en tiempo real [11]. Está basado en los Modelos Ocultos de Markov explicados en la sección 2.4. Para el entrenamiento de estos modelos se ha utilizado el conjunto *train.full* de la base de datos proporcionada por la organización del *MGB Challenge*. Respecto al modelado acústico utiliza unidades acústicas dependientes del contexto, donde cada unidad se modela con una GMM de 16 componentes. Si nos centramos en el ML utilizado, se trata de un modelo de gramática de reglas con formato como el expuesto en la sección 4.2.2. Antes de iniciar el reconocimiento se generan las gramáticas a partir del fragmento del *script* que recibe como entrada el reconocedor, e inserta las redes de fonemas donde se haya especificado haciendo uso de caracteres especiales en el bloque de inserción de modelo de relleno.

El funcionamiento del reconocedor es a nivel de *chunk*, es decir, recibe únicamente el *script* y el audio de un *chunk* y realiza el alineamiento. Por tanto será necesario realizar tantas llamadas al reconocedor como *chunks* compongan el programa para obtener el alineamiento del programa completo.

Un parámetro importante para el reconocedor es el **peso de la red de fonemas**. Este valor está directamente relacionado con la probabilidad de que el sistema de reconocimiento opte por uno o varios fonemas, en lugar de transitar directamente entre las palabras de la gramática. Será otro de los parámetros sobre los que se realizarán varios experimentos, variando su valor para comprobar cuál es la mejor configuración. Dado el amplio rango de valores que puede tomar este parámetro, de ahora en adelante nos referiremos a él en escala logarítmica con la letra P. Por tanto la relación entre P y el auténtico valor del peso de la red de fonemas, p, es la mostrada en (4.1).

$$P = \log_{10} p \tag{4.1}$$

La salida del reconocedor contiene la información del texto alineado a nivel de palabra con las correspondientes marcas temporales. Cabe destacar que existe la posibilidad de que el reconocedor obtenga una salida nula si no ha sido capaz de alinear el texto con el audio. La información del texto alineado es generada por el reconocedor en dos formatos diferentes:

- `.ctm`: formato exigido por la organización del *MGB Challenge* para la evaluación de los resultados en la herramienta de *scoring* proporcionada. En la Tabla 4.2 vemos un ejemplo de este formato. En el primer campo se encuentra el nombre del fichero con el que estamos trabajando. El segundo campo se corresponde con el canal de audio utilizado, no obstante

Nombre fichero	0	Tini	Duración	Palabra
FicheroA	0	0.00	0.50	HELLO
FicheroA	0	0.50	0.22	MY
FicheroA	0	0.72	0.5	NAME
FicheroA	0	1.22	0.5	IS

Tabla 4.2: Ejemplo de formato `.ctm`

aquí colocamos un 0 ya que no se utiliza esta información. Por último, en los tres campos restantes, tenemos la información del texto alineado: el tercer campo es el tiempo de inicio de la palabra y el cuarto la duración, ambos en segundos, y en el quinto campo encontramos la palabra alineada.

- `.lbl`: se trata de un formato de etiquetas de audio. Se utiliza para comprobar si el alineamiento es correcto de una forma mas gráfica en un *software* de manipulación de audio, como puede ser Audacity [12]. Permite escuchar el audio y visualizar las palabras alineadas al mismo tiempo. En la Tabla 4.3 vemos un ejemplo de este formato. En este

Tini	Tfin	Palabra
0.00	0.50	HELLO
0.50	0.72	MY
0.72	1.22	NAME
1.22	1.72	IS

Tabla 4.3: Ejemplo de formato `.lbl`

caso, el primer y segundo campo representan el tiempo de inicio y el tiempo de final de la palabra alineada, respectivamente, ambos en segundos. El tercer campo es la palabra alineada.

4.2.4. Composición

Este bloque recoge la información de salida de los diferentes *chunks* que forman un programa y se encarga de concatenar cada una de dichas salidas para formar la salida completa a evaluar en la herramienta de *scoring*. Es importante comentar la necesidad de utilizar un componente de sincronización con el reconocedor automático del habla, ya que el bloque de composición no debe actuar hasta que todos los *chunks* en los que se divide un programa hayan sido reconocidos, para evitar errores y pérdida de información.

4.2.5. Evaluación

Una vez obtenida la salida del sistema de alineamiento se realiza la evaluación de resultados mediante las métricas de error expuestas en las ecuaciones (3.1), (3.2) y (3.3). Para llevar a cabo esta tarea se utilizará la herramienta de *scoring* proporcionada por la organización del *MGB Challenge* y presentada en la sección 3.5.

4.3. Análisis de resultados

Como ya se explicó anteriormente, este TFG se centra en el conjunto de datos *eval.task1* de la base de datos proporcionada por la organización del *MGB Challenge*, compuesta por 16 programas de televisión con una duración aproximada de 11 horas. Los programas que

podemos encontrar en la base de datos son muy variados y cubren diferentes géneros, desde concursos hasta documentales.

Tal y como se expone en la sección 4.2.1, existe una ambigüedad en la localización del audio correspondiente a cada *chunk* de texto. En primera instancia, planteamos la obtención de marcas temporales aproximadas basadas en un reparto uniforme de la duración total del programa considerado entre los *chunks* que lo componen: de la información proporcionada por la organización del *MGB Challenge* en los ficheros `.xml` se extraen las distintas intervenciones y se realiza un reparto temporal uniforme entre la longitud total del audio. Esta aproximación, a pesar de tener sus limitaciones, sirve para reducir en cierta medida la incertidumbre en la localización temporal del texto. Algunos de los inconvenientes de este método se tratarán de solventar en la herramienta de segmentación incluyendo las guardas temporales. La información del reparto temporal uniforme se guarda en los ficheros `.lab`, que servirán de entrada a la herramienta de segmentación. Estos ficheros contienen la información del texto de una intervención y las marcas de tiempo aproximadas.

4.3.1. Sistema base

En un primer experimento se desea observar cómo influyen en el rendimiento del sistema de alineamiento, tanto el tiempo de guarda aplicado en la herramienta de segmentación, como el peso de la red de fonemas en el reconocedor. Para ello, y por simplicidad, consideramos 2 programas de los 16 que componen la base de datos. De experimentos previos a este TFG se identificaron 2 programas cuyos resultados de alineamiento asociados eran uno de los mejores y uno de los peores respectivamente, a los que a partir de ahora denominaremos "programa A" y "programa B". Realizamos un barrido desde un tiempo de guarda de 10 segundos hasta 140 segundos, considerando mismos tiempos de guarda inicial y final. Para cada tiempo de guarda considerado, se barrió también el valor del peso de la red de fonemas, considerando el rango entero desde -2 hasta -7. Respecto a las redes de fonemas intermedias, en este experimento se insertó una red tras cada linea del fichero `.lab`.

En la Figura 4.5 vemos la medida F obtenida para cada una de las diferentes configuraciones de parámetros evaluada en los programas A y B, respectivamente. Observamos que la tendencia es creciente con el tiempo de guarda, llegando a alcanzar medidas F cercanas a 0.2 en el programa A y cercanas a 0.8 en el programa B, para tiempos de guarda alrededor de 120 segundos. Aumentar el tiempo de guarda en la herramienta de segmentación hace que el reconocedor disponga de más audio, y por tanto, la probabilidad de encontrar el texto del *script* en el audio y alinearlo correctamente aumente. De esta forma, paliamos parcialmente el problema de los tiempos aproximados uniformes que se utilizan en los ficheros `.lab`. No obstante puede parecer que la solución para obtener mejores resultados sea aumentar el tiempo de guarda, pero esto no es así. Para tiempos de guarda mayores de 140 segundos el reconocedor no es capaz de obtener una salida si no disminuimos el valor del *beam search* en el algoritmo de decodificación. Esto se debe a que el sistema dispone de demasiado audio y el número de caminos a progresar en el algoritmo de Viterbi aumenta considerablemente, provocando que el reconocedor no pueda realizar la decodificacion correctamente.

En la Tabla 4.4 podemos observar los valores de F para las mejores configuraciones de parámetros obtenidas en el barrido. Vemos que los mejores resultados se obtienen para tiempos de guarda cercanos a 120 segundos y valor del peso de la red de fonemas en torno a -6. Respecto a la influencia del peso de la red de fonemas, se aprecia que la medida F disminuye a medida que aumenta el valor del peso de la red, incluso decayendo drásticamente para $P = -2$.

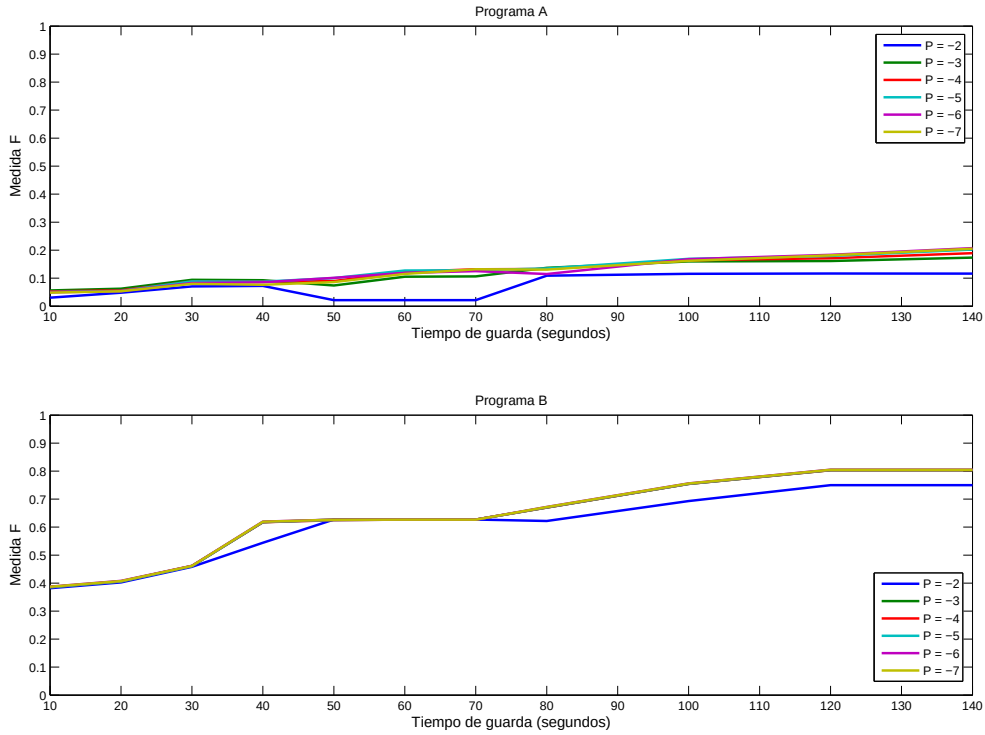

Figura 4.5: Evolución de la medida F en función del tiempo de guarda para distintos valores de peso de la red de fonemas

		programa A		programa B	
		Tiempo de guarda(s)		Tiempo de guarda(s)	
		120	140	120	140
Peso red de fonemas	-2	0.1167	0.1163	0.7491	0.7497
	-3	0.1614	0.1763	0.8034	0.8034
	-4	0.1711	0.1893	0.8034	0.8034
	-5	0.1792	0.2021	0.8040	0.8040
	-6	0.1832	**0.2072**	0.8052	**0.8052**
	-7	0.1816	0.2050	0.8049	0.8049

Tabla 4.4: Medida F en función de distintos tiempos de guarda y peso de la red de fonemas para programa A y programa B

Una vez obtenidos los resultados para los programas A y B llevamos a cabo los mismos experimentos utilizando los 16 programas que componen el *set eval.task1*, pero fijando los valores de los parámetros a los que mejores resultados de F han devuelto anteriormente (tiempo de guarda 120 y 140 segundos, peso de la red de fonemas -5, -6 y -7). Los resultados obtenidos son los mostrados en la Tabla 4.5, donde el mejor resultado corresponde a un tiempo de guarda 140 segundos con peso de la red de fonemas -7, lo cual es consistente con la tendencia observada en las Figura 4.5. El valor de F obtenido **0.5583**, será considerado como nuestro ***baseline***, es decir punto de partida de nuestra experimentación y valor con el que compararemos los resultados obtenidos de aquí en adelante, en el resto de experimentos llevados a cabo.

Tiempo de guarda(s)	Peso red de fonemas	Precision	Recall	F
120	-5	0.6425	0.3877	0.4835
	-6	0.6255	0.4441	0.5194
	-7	0.6124	0.4736	0.5341
140	-5	0.6563	0.4186	0.5112
	-6	0.6365	0.4711	0.5415
	-7	**0.6227**	**0.5060**	**0.5583**

Tabla 4.5: Medida F, *precision* y *recall* en función de distintos tiempos de guarda y peso de la red de fonemas para la base de datos completa

Nº Palabras tras red de fonemas	Tiempo de guarda(s)	Peso red de fonemas	Precision	Recall	F
1	120	-6	0.5997	0.4261	0.4982
		-7	0.5884	0.3979	0.4747
	140	-6	0.6144	0.4290	0.5052
		-7	0.5984	0.3984	0.4783
5	120	-6	0.6077	0.5063	0.5524
		-7	0.6055	0.5194	0.5592
	140	-6	0.6151	0.5431	0.5769
		-7	**0.6163**	**0.5483**	**0.5803**
10	120	-6	0.5981	0.4989	0.5440
		-7	0.5952	0.5059	0.5469
	140	-6	0.6049	0.5330	0.5667
		-7	0.6051	0.5425	0.5721
25	120	-6	0.6004	0.4415	0.5089
		-7	0.5875	0.4814	0.5292
	140	-6	0.6119	0.4758	0.5353
		-7	0.5962	0.5155	0.5530
50	120	-6	0.6148	0.4138	0.4946
		-7	0.5976	0.4520	0.5147
	140	-6	0.6225	0.4424	0.5172
		-7	0.6069	0.4829	0.5378

Tabla 4.6: Medida F, *precision* y *recall* en función de distintos tiempos de guarda y peso de la red de fonemas introduciendo redes de fonemas intermedias para la base de datos completa

4.3.2. Inserción de modelos de relleno

La primera mejora que se plantea es la de introducir un mayor número de redes de fonemas intermedias. Hasta ahora las redes de fonemas se han colocado solo al principio y al final de cada linea del `.lab`, y aun así la longitud de los fragmentos sin redes de fonemas intermedias puede llegar a ser excesivamente larga. Por tanto, en este segundo experimento se evaluará la posibilidad de introducir redes de fonemas intermedias cada 1, 5, 10, 25 y 50 palabras, para las mejores configuraciones obtenidas en el experimento anterior (tiempo de guarda 120 y 140 segundos, y peso de la red de fonemas -6 y -7), y para todos los programas del *eval.task1*.

En la Tabla 4.6 se detallan los resultados obtenidos para esta nueva estrategia. En el caso

extremo de introducir una red de fonemas después de cada palabra vemos que las prestaciones del sistema caen. En cambio, para el caso de introducirlas cada 5 palabras se produce el máximo valor de medida F con un 0.5803 para tiempo de guarda de 140 segundos y un peso de la red de fonemas -7. Observamos que desde el caso de redes de fonemas cada 10 palabras hasta el caso de redes de fonemas cada 50 palabras, las prestaciones del sistema de alineamiento disminuyen gradualmente, quedando incluso por debajo de la medida F obtenida para el sistema base (0.5583) con redes introducidas cada 50 palabras. Por lo tanto, la estrategia de introducir de redes de fonemas intermedias permite aumentar la medida F desde 0.5583 hasta **0.5803**, lo cual se traduce en una mejora relativa del **3.94 %**.

En el siguiente capítulo se aplicarán nuevas estrategias para lograr aumentar la medida F respecto a los valores aquí obtenidos. El concepto fundamental en el que nos apoyaremos será la reducción de la incertidumbre al tratar de ubicar el texto con el audio correspondiente. Ello se llevará a cabo mediante el uso de marcas temporales parciales generadas automáticamente en lugar del reparto temporal uniforme realizado hasta ahora.

Capítulo 5

Alineamiento con anclas temporales

En este capítulo se presenta el uso de una nueva estrategia para mejorar las prestaciones del sistema de alineamiento, basada en la reducción de la incertidumbre al ubicar texto con audio.

5.1. Introducción

Con el término ancla nos referimos a una palabra o conjunto de palabras del *script* cuya marca temporal asociada (tiempo inicial y tiempo final en la secuencia de palabras pronunciada) es conocida *a priori* para utilizar como información adicional en el sistema de alineamiento. Respecto al diagrama de bloques de la Figura 4.2, dichas anclas se encuentran al principio de la cadena, tal y como se observa en la Figura 5.1, como un módulo auxiliar que proporciona dichas marcas temporales parciales que el reconocedor utilizará como guía. Con esta estrategia se busca disminuir la incertidumbre respecto a la localización temporal del texto del *script* en el audio, y así, mejorar las prestaciones con respecto a las estrategias adoptadas en el capítulo anterior, donde se llevaba a cabo un reparto temporal uniforme.

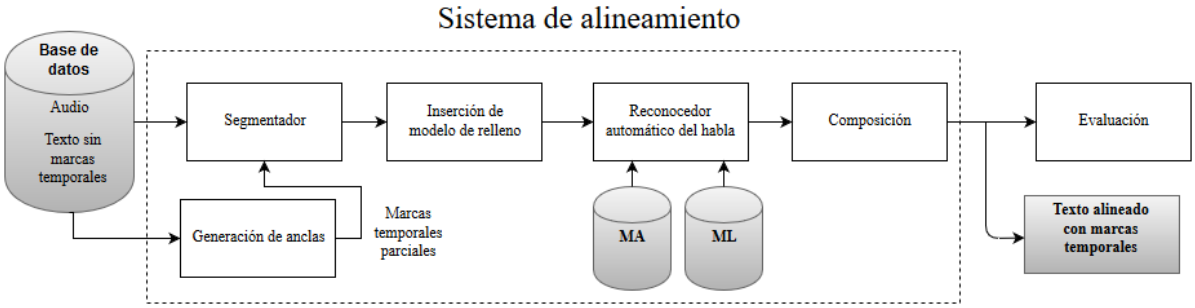

Figura 5.1: Diagrama de bloques del sistema de alineamiento con anclas temporales

En relación con el resto de bloques que aparecen en la Figura 5.1, no han sufrido ningún cambio respecto del sistema *baseline* y su funcionamiento es el mismo que el explicado en la sección 4.2 de esta memoria.

5.2. Generación de anclas temporales

Como se observa en la Figura 5.1, la generación de las anclas precisa de un nuevo bloque cuyo funcionamiento se detalla en esta sección. El proceso de generación de anclas se puede

dividir en tres tareas bien diferenciadas:

1. **Reconocimiento mediante MLE del audio del programa**: sobre el audio del programa proporcionado se realiza un reconocimiento de la secuencia de palabras locutada basado en un MLE de trigrama, que ha sido entrenado utilizando el *script* correspondiente al programa. Respecto a los MA, se basan en HMM cuya salida se modela con GMM de 16 componentes y 3 estados por unidad acústica, que en este caso son trifonemas, y han sido entrenados con un subconjunto del *train.full* de la base de datos del *MGB Challenge*. Además de obtener la transcripción, se obtienen unas marcas temporales proporcionadas por la herramienta de decodificación utilizada, que en esta ocasión ha sido HTK [4].

2. **Alineamiento de secuencias**: Partiendo de la transcripción obtenida en el reconocimiento estocástico y del texto del *script* del programa, se realiza un alineamiento de ambas secuencias. Este alineamiento tiene como objetivo encontrar la mejor correspondencia entre ambos textos, de forma similar al alineamiento que se lleva a cabo antes de calcular la WER en un sistema de RAH, y se suele implementar mediante algoritmos de programación dinámica. En este caso se utilizó el *toolkit* de *scoring* SCLITE [7], y el alineamiento obtenido fue similar al mostrado en la Figura 5.2.

```
95.23 95.66 C i i
x.x x.x I *** WAS
x.x x.x I ** ON
x.x x.x I *** TOP
x.x x.x I ** OF
x.x x.x I ****** THINGS
95.66 98.11 S DON'T I
98.11 102.61 S KNOW THOUGHT
102.61 106.58 C i i
x.x x.x I *** WAS
106.58 108.46 S DIDN'T COPING
108.46 110.23 S ABDO NO
110.23 133.19 S SCAR COME
133.19 134.13 S THE ON
134.13 134.40 C it's it's
134.40 149.54 S FINE NOT
149.54 158.69 S OH LIKE
158.69 159.88 S I'M YOU
159.88 160.64 S GOING HAVE
160.64 162.16 C to to
162.16 170.96 C go go
170.96 171.36 D TO **
171.36 171.81 D MUST ****
171.81 177.47 S NO THROUGH
```

Figura 5.2: Fichero de anclas tras el alineamiento de secuencias

   En la primera y segunda columna observamos el tiempo de inicio y final de la palabra. En la tercera columna podemos encontrar una letra bien sea C, I, D o S, que simbolizan una palabra correcta, una inserción, un borrado o una sustitución, respectivamente. Finalmente, en la cuarta y quinta columnas encontramos la palabra obtenida en el reconocimiento libre y la palabra procedente del *script*, respectivamente.

3. **Procesado del fichero de anclas**: de la información contenida en el fichero mostrado en la Figura 5.2 solo nos interesa quedarnos con aquellas palabras etiquetadas con una C, es decir, aquellas palabras reconocidas que tras el alineamiento de secuencias se han alineado correctamente con una palabra del *script*, y que identificaremos y utilizaremos como ancla. En primer lugar se eliminan todas las palabras etiquetadas con una D, es decir, aquellas palabras de la transcripción que no se corresponden con ninguna palabra del *script* tras el alineamiento. Seguidamente, se procesa el fichero sin borrados para obtener las anclas. Aquí entran en juego varios aspectos:

- **Número de palabras correctas consecutivas**: para asegurar un mínimo de precisión en las marcas temporales de las anclas, descartamos aquellas palabras correctas que estén aisladas y consideraremos un ancla como una secuencia de dos o más palabras consecutivas correctas. El número de palabras consecutivas que exijamos para considerar un ancla y su influencia en las prestaciones del sistema se evaluará en la fase experimental.
- **Localización de las anclas**: el proceso de generación de anclas termina con la generación de nuevos ficheros `.lab` con las marcas temporales dadas por las anclas. No obstante, existen dos opciones diferentes a la hora de trabajar con las anclas: considerarlas como el principio o como el final de un fragmento de texto. Sendas variantes se evaluarán en la fase experimental.

Tanto el reconocimiento estocástico como el alineamiento de secuencias han sido llevados a cabo por investigadores de ViVoLab, disponiendo para la realización de este TFG de los ficheros de anclas necesarios.

5.3. Análisis de resultados

Para los nuevos experimentos se han generado unos nuevos ficheros `.lab` apoyados en las marcas temporales fiables proporcionadas por las anclas. *A priori*, esto supone una aproximación más precisa que la estrategia adoptada en los experimentos desarrollados en la sección 4.3, lo cual se comprobará a lo largo de los nuevos experimentos llevados a cabo, en términos de medida F.

5.3.1. Sistema con anclas temporales

En el primer experimento realizado se barrieron todos los parámetros que caracterizan el sistema con anclas, es decir, el número de palabras correctas consecutivas para considerar un ancla (desde 2 palabras hasta 5), la posición de las anclas (al principio o al final de un fragmento de texto), y los dos parámetros ya considerados en el sistema *baseline* (el tiempo de guarda y el peso de la red de fonemas). No obstante, para el tiempo de guarda no se consideran valores tan altos como los de experimentos anteriores, ya que al introducir las anclas la localización temporal del texto en el audio debería ser más precisa, y por tanto no sería necesario contar con guardas temporales tan amplias. Así, se consideran valores de 10, 25, 50 y 75 segundos. Respecto al peso de la red de fonemas, teniendo en cuenta los resultados del sistema de base, experimentaremos con $P = -5, -6$ y -7, pues demostraron ser los que mayor medida F devolvían.

Todos los resultados se muestran en las Tablas del apéndice A, donde se presentan los valores de *precision*, *recall* y medida F para cada una de las configuraciones evaluadas. En este capítulo tan solo se mostrarán y comentarán los resultados más relevantes.

En la Figura 5.3 se representa la mejora relativa de F respecto al *baseline* (0.5583) obtenida con diferentes configuraciones de las anclas para los tres valores del peso de la red de fonemas evaluados. En linea continua observamos las configuraciones con anclas colocadas al principio de cada fragmento de texto y en linea discontinua aquéllas con anclas al final. Si nos centramos en el valor del peso de la red de fonemas, los resultados son similares a los del sistema base, donde un menor peso redunda en una mayor medida F.

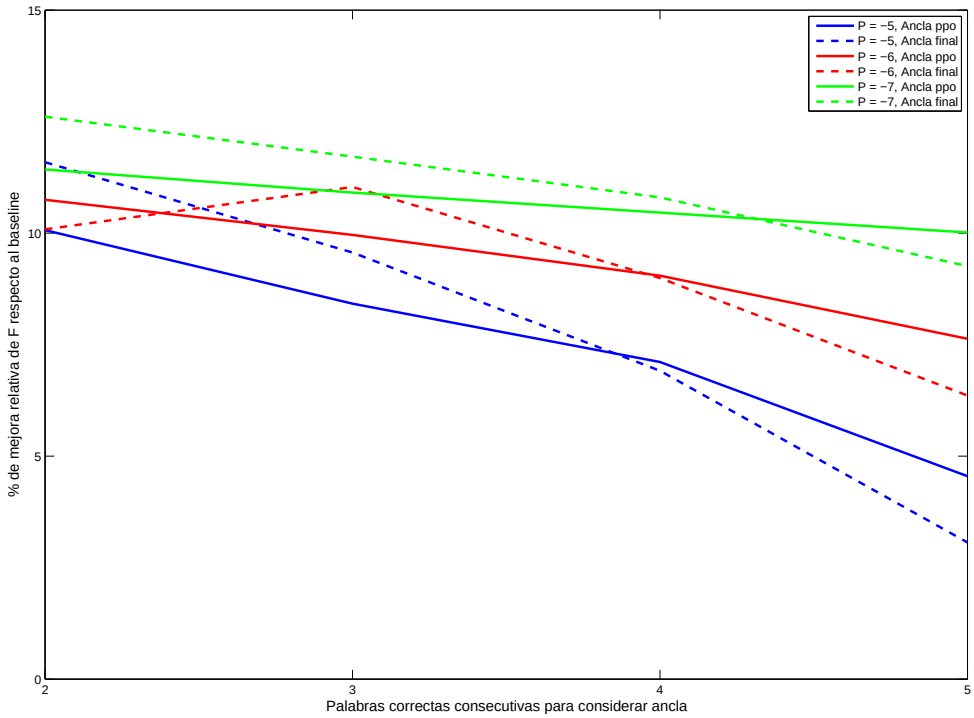

Figura 5.3: Mejora relativa de la medida F respecto al *baseline* para diferentes P y configuración de anclas

Respecto a las diferentes configuraciones de anclas consideradas, se observa que, conforme vamos incrementando el número de palabras correctas para considerar un ancla, la medida F decae. Esto se debe a que al aumentar el número de palabras hacemos que la cantidad de anclas encontradas disminuya, y por tanto nuestro sistema dispondrá de un menor número de marcas temporales a usar como guía en el alineamiento. Finalmente, respecto a la posición de las anclas en el texto, se observa que en términos generales las anclas colocadas al final del texto tienen unas prestaciones ligeramente superiores, aunque esta leve diferencia puede ser debida a una dependencia con los datos, de modo que al no ser estadisticamente significativa, no nos permite afirmar que la colocación de las anclas al principio o al final del texto sea mejor o peor.

Considerando la configuración de anclas que tiene asociada una mayor medida F, la representamos frente a los distintos valores del tiempo de guarda en la Figura 5.4. En linea continua vemos los resultados asociados a anclas colocadas al principio y en linea discontinua las anclas colocadas al final. Aquí observamos como, a diferencia de los experimentos realizados en la sección 4.3, la influencia del tiempo de guarda en la medida F ha disminuido notablemente. Esto es consistente con la nueva estrategia adoptada: una mayor precisión en la localización temporal del texto en el audio hace que aumente la probabilidad de que el texto a alinear se corresponda con el audio que se proporciona al reconocedor.

A la vista de estos resultados podemos afirmar que la estrategia de las anclas es adecuada, ya que supone una mejora sustancial en la medida F respecto del *baseline*, en todas y cada una de las configuraciones evaluadas. La mejor medida F se consigue para un tiempo de guarda

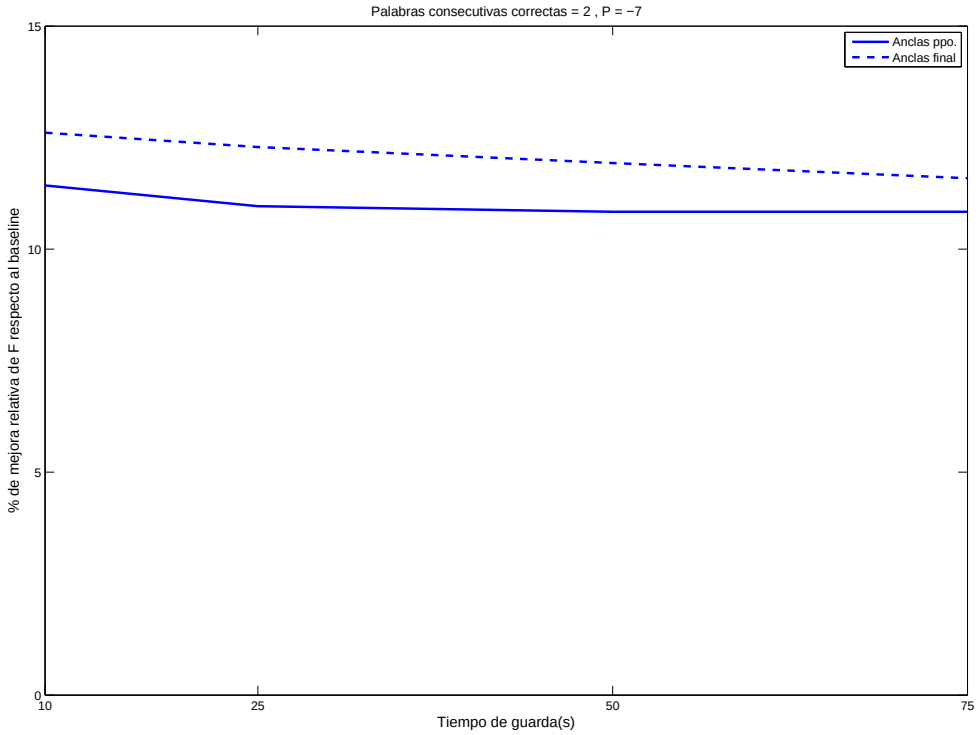

Figura 5.4: Mejora relativa de la medida F respecto al *baseline* para diferentes tiempos de guarda y mejor configuración de anclas

de 10 segundos, peso de la red de fonemas -7, y 2 palabras correctas consecutivas por ancla. En el caso de anclas al final, se obtiene una mejora relativa del 12.61 % respecto al *baseline*, alcanzando una medida F de **0.6287**, en cambio para el caso de anclas al principio del texto la mejora relativa se reduce ligeramente alcanzando un valor de 11.43 % y una medida F de 0.6221.

5.3.2. Inserción de modelos de relleno

Una vez evaluadas las distintas configuraciones de anclas en nuestro sistema y localizadas aquéllas cuya combinación de parámetros redundan en mejores prestaciones, planteamos un nuevo experimento similar al expuesto en la sección 4.3, incluyendo en el texto redes de fonemas intermedias cada N palabras.

En las Tablas 5.1 y 5.2 se muestran los resultados obtenidos para la mejor configuración de anclas al principio y al final introduciendo redes de fonemas cada 10, 25 y 50 palabras. Los mejores valores de F se obtienen para la configuración de redes de fonemas intermedias cada 10 palabras, consiguiendo aumentar el valor de F desde 0.6287 hasta **0.6332** (+0.72 %), en el caso de las anclas al final, y desde 0.6221 hasta 0.6253 (+0.51 %), en el caso de las anclas al principio. Tal y como se demostró en la sección 4.3, el hecho de añadir redes de fonemas intermedias permite aumentar la medida F del sistema, y en este caso junto con la estrategia de las anclas temporales, permite aumentarla aun más.

Peso red de fonemas	Nº Palabras tras red de fonemas	Precision	Recall	F
-8	10	0.6175	0.6399	0.6285
	25	0.6114	0.6479	0.6291
	50	0.6124	0.6470	0.6293
-7	10	**0.6169**	**0.6504**	**0.6332**
	25	0.6133	0.6471	0.6298
	50	0.6132	0.6453	0.6288
-6	10	0.6191	0.6478	0.6331
	25	0.6126	0.6436	0.6277
	50	0.6139	0.6410	0.6272
-5	10	0.6201	0.6431	0.6314
	25	0.6134	0.6353	0.6242
	50	0.6158	0.6301	0.6229

Tabla 5.1: Medida F, *precision* y *recall* con anclas al final y redes de fonemas intermedias

Peso red de fonemas	Nº Palabras tras red de fonemas	Precision	Recall	F
-8	10	**0.6137**	**0.6373**	**0.6253**
	25	0.6070	0.6418	0.6239
	50	0.6068	0.6419	0.6239
-7	10	0.6137	0.6371	0.6252
	25	0.6056	0.6391	0.6219
	50	0.6063	0.6390	0.6222
-6	10	0.6120	0.6330	0.6223
	25	0.6021	0.6332	0.6173
	50	0.6044	0.6334	0.6186
-5	10	0.6088	0.6295	0.6190
	25	0.6032	0.6257	0.6142
	50	0.6050	0.6235	0.6141

Tabla 5.2: Medida F, *precision* y *recall* con anclas al principio y redes de fonemas intermedias

5.3.3. Herramienta de corrección de finales

Llegados este punto de la experimentación se ha conseguido elevar la medida F desde la obtenida en el *baseline*, 0.5583, hasta 0.6332 considerando la información aportada por las anclas y la inclusión de redes de fonemas intermedias, lo cual se traduce en una mejora relativa del 13.53 %. No obstante hasta ahora no se ha tenido en cuenta una de las limitaciones del sistema de reconocimiento automático del habla: el reconocedor considera el final de una palabra como el principio de la palabra siguiente. Esto implica que no es capaz de distinguir silencios o fragmentos de audio sin voz entre dos palabras, tal y como se ilustra en la Figura 5.5. Es decir, palabras correctamente reconocidas por nuestro sistema de alineamiento son consideradas incorrectas en la herramienta de *scoring* debido a un tiempo de final marcado por el inicio de la siguiente palabra, lo que provoca una disminución en la medida F.

Como solución a este problema se dispone de una herramienta de corrección de finales para ser aplicada a la salida del sistema de alineamiento, tratando de paliar los efectos de la

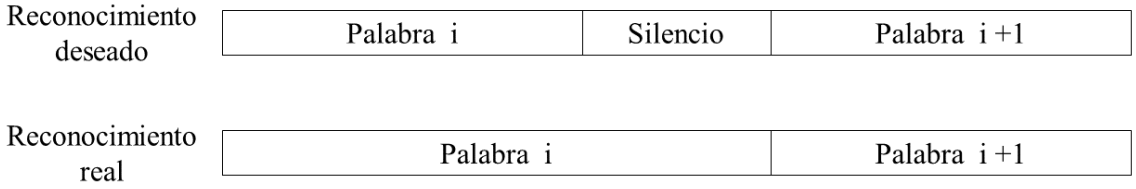

Figura 5.5: Limitación en el reconocimiento de los finales de palabra

limitación en el reconocedor. Para su funcionamiento, requiere del vocabulario del programa a reconocer, donde cada palabra de salida tenga asociada su transcripción fonética. Dada una palabra, cuenta el número de fonemas que la forman y, a partir de una duración media del fonema que recibe como parámetro de entrada, calcula una duración estimada para cada palabra. Si esta duración estimada es mayor que la duración que ha calculado el sistema de alineamiento la herramienta de corrección deja la palabra como estaba, pero si es menor la duración de la palabra se fija a la duración estimada por la herramienta. Esto es equivalente a la ecuación (5.1),

$$D_{out} = \begin{cases} D_{ali} & \text{si } WN_{phon} > D_{ali} \\ WN_{phon} & \text{si } WN_{phon} \leq D_{ali} \end{cases} \tag{5.1}$$

donde W representa la duración media en segundos por fonema, N_{phon} el número de fonemas que componen una palabra, D_{ali} la duración en segundos de la palabra determinada por el sistema de alineamiento y D_{out} la duración en segundos de la palabra tras la herramienta de corrección de finales.

Para la mejor configuración de parámetros obtenida hasta el momento (tiempo de guarda 10 segundos, peso de la red de fonemas -7, 2 palabras correctas consecutivas y redes de fonemas intermedias cada 10 palabras), se realizó un experimento con esta herramienta con el objetivo de comprobar cuál es la duración media por fonema más adecuada para lograr las mejores prestaciones. En la Tabla 5.3 se observan los resultados obtenidos al barrer el parámetro de la duración media de los fonemas desde 0.01 segundos hasta 0.1 segundos en pasos de 0.01 segundos.

Se observa que los mejores resultados se obtienen para una duración media por fonema de 0.04 segundos, alcanzando una medida F de **0.6747**. En este caso concreto, la herramienta de corrección de finales ha sido capaz de corregir unas 2500 palabras, pasando de 37384 palabras correctamente alineadas a 39902. Vemos también que si tomamos una duración media por fonema demasiado grande (de 0.05 segundos en adelante) la medida F apenas sufre cambios ya que la herramienta de corrección no tiene efecto alguno. Justo al contrario ocurre si asignamos un valor muy pequeño (0.01, 0.02 y 0.03), ya que en este caso la herramienta limita la duración de palabras que no deberían ser corregidas, haciendo así que las prestaciones del sistema de alineamiento caigan drásticamente.

Por tanto, mediante el uso de esta herramienta se ha conseguido aumentar considerablemente el número de palabras correctamente alineadas, lo cual conlleva un aumento en la medida F hasta 0.6747, lo que supone una mejora relativa del 20.85 % respecto de los resultados obtenidos en el *baseline* (0.5583).

Duración media por fonema(s)	Precision	Recall	F
0.01	0.3131	0.3317	0.3221
0.02	0.5455	0.5776	0.5611
0.03	0.6496	0.6873	0.6680
0.04	**0.6562**	**0.6942**	**0.6747**
0.05	0.6418	0.6784	0.6596
0.06	0.6296	0.6650	0.6468
0.07	0.6227	0.6575	0.6396
0.08	0.6202	0.6546	0.6369
0.09	0.6183	0.6525	0.6349
0.1	0.6173	0.6514	0.6339

Tabla 5.3: Medida F, *precision* y *recall* para la mejor configuración de parametros y anclas al final, corrigiendo los finales de palabra

5.3.4. Incorporación de la información de intervenciones

En este último experimento se fusiona la información temporal de las anclas con la información de las intervenciones para conseguir aumentar las prestaciones del sistema de alineamiento. Esta información de intervención se traducirá en forma de una red de fonemas intermedia al final de cada intervención, de forma similar a los experimentos de la sección 4.3.

Para la mejor combinación de anclas obtenida hasta ahora (tiempo de guarda 10 segundos, palabras correctas consecutivas para considerar ancla 2 y peso de la red de fonemas -7) e incluyendo las marcas de intervención, se obtiene una medida F de 0.6443, por encima de los resultados de introducir redes de fonemas cada 10 palabras (0.6332). No obstante, quedando demostrado en el experimento anterior la posibilidad de mejora que ofrece la herramienta de corrección de finales, se aplica también en este experimento, obteniendo así un resultado final de medida F de **0.6873**, después de corregir el final de aproximadamente 3000 palabras. Este resultado es el mejor obtenido en todos los experimentos realizados durante este TFG y supone una mejora relativa de medida F de un 23.11 % respecto al *baseline* (0.5583).

Capítulo 6

Conclusiones y líneas futuras de trabajo

Este capítulo presenta una breve descripción resumida y una valoración crítica del conjunto del trabajo realizado, además de proponer posibles líneas futuras.

6.1. Conclusiones

El aumento de la demanda de contenidos multimedia ha hecho que suceda lo mismo con el contenido subtitulado. Un sistema como el propuesto en este TFG simplificaría de forma notable la tarea a las empresas del sector audiovisual, convirtiendo este proceso en algo en gran medida automatizado. Basándose en motores de RAH y tomando como contexto principal la tarea de alineamiento del *MGB Challenge*, el objetivo principal de este TFG es el desarrollo y evaluación de diferentes estrategias que permitan aumentar la precisión en dicha tarea.

Para llevar a cabo este objetivo se partió de un sistema de alineamiento base (*baseline*) cuyos parámetros de funcionamiento (tiempo de guarda y peso de la red de fonemas) se optimizaron mediante una primera serie de experimentos, obteniendose así una medida F de partida de 0.5583 a comparar con las diferentes estrategias evaluadas. A continuación, se propuso incluir un módulo auxiliar (anclas temporales) para disminuir la incertidumbre en la localización del texto en el audio. A lo largo de una nueva serie de experimentos se demostró la eficacia de esta nueva estrategia, que proporciona unas tasa de mejora cercana al 13 % (F = 0.6332) respecto del *baseline*. Seguidamente, con el fin de paliar los problemas del reconocedor en la detección de los silencios entre palabras, se utilizó una herramienta de corrección de finales. Los resultados conseguidos fueron satisfactorios, alcanzando una medida F de 0.6747. Finalmente, se incorporó la información de las intervenciones, llegando a una medida F de 0.6873, lo que se traduce en una mejora relativa del 23.11 % respecto del *baseline*. En la Tabla 6.1 se muestra un resumen de los resultados obtenidos a lo largo de este trabajo fin de grado para las diferentes estrategias evaluadas.

Ha quedado probado que el uso de estrategias que permitan reducir la ambigüedad en la localización del texto en el audio, como la incorporación de marcas temporales parciales, redundan en una mejora de las prestaciones en la tarea de alineamiento. También se ha demostrado la mejora en las prestaciones que supone el incluir modelos de relleno en la gramática del reconocedor, implementados en forma de redes de fonemas en nuestro caso, bien sea integrándolos cada N palabras o teniendo en cuenta la información de las intervenciones.

Por lo tanto, **el objetivo que se pretendía alcanzar con la realización de este**

Estrategia	Medida F	Mejora relativa respecto al *baseline*
baseline	0.5583	-
baseline + redes de fonemas intermedias	0.5803	+3.94 %
anclas temporales	0.6287	+12.61 %
anclas temporales + redes de fonemas intermedias	0.6332	+13.42 %
anclas temporales + redes de fonemas intermedias + corrección de finales	0.6747	+20.85 %
anclas temporales + intervención + corrección de finales	0.6873	+23.11 %

Tabla 6.1: Medida F obtenida para cada estrategia considerada y mejora relativa respecto del *baseline*

trabajo fin de grado ha sido plenamente satisfecho. Se ha conseguido aumentar la precisión en la tarea de alineamiento, quedando caracterizado el sistema *baseline* y las posteriores estrategias propuestas.

6.2. Líneas futuras de trabajo

A la vista de los resultados del sistema de alineamiento expuestos en esta memoria, cabe la posibilidad de aprovechar parte del trabajo realizado en este TFG como punto de partida en futuros proyectos siguiendo nuevas líneas de actuación.

En primer lugar se propondría la mejora de los modelos acústicos del sistema de RAH. Respecto de los utilizados durante el desarrollo de este trabajo fin de grado, se ampliaría el número de estados utilizados para modelar una unidad acústica, obteniendo modelos basados en 3 estados por unidad acústica.

Otro aspecto que constituye un posible punto de mejora es el tratamiento de las siglas y acrónimos del texto de los *scripts*. El sistema de RAH actualmente filtra todos los caracteres no alfanuméricos, perdiendo así la información respectiva a las siglas. Se propondría introducir algún carácter de escape o secuencias especiales de caracteres para poder recuperar la información de las siglas y acrónimos a la salida del reconocedor.

Finalmente se plantearía la aplicación de nuevas estrategias en la generación y tratamiento de la información contenida en cada uno de los *chunks* que se obtienen tras la herramienta de segmentación. Se propondría generar diferentes gramáticas para un mismo *chunk*, incorporando otra vez la información de las intervenciones, de tal forma que el reconocedor fuera capaz de llevar en paralelo varias gramáticas y alinear fragmentos aún más pequeños que los empleados hasta el momento.

Apéndice A

Resultados del alineamiento con anclas temporales

En este apéndice se detallan de forma íntegra los resultados obtenidos en la experimentación llevada a cabo con el sistema con anclas temporales, complementando así los datos aportados en la sección 5.3 de la memoria.

A.1. Introducción

Este experimento tiene como objetivo comprobar si la incorporación de las anclas temporales en nuestro sistema de alineamiento es adecuada, llevando a cabo un barrido de los valores de parámetros que rigen el sistema. Concretamente se han evaluado diferentes configuraciones en función del tiempo de guarda, el peso de la red de fonemas, las palabras correctas consecutivas necesarias para considerar un ancla y la localización de las anclas en el texto.

A.2. Resultados

En las siguientes Tablas se recogen los resultados del experimento en terminos de *precision*, *recall* y medida F, resaltando en negrita la mejor configuración evaluada, es decir, aquella con mayor medida F asociada. Por simplicidad, los resultados se muestran por separado según la localización de las anclas en el texto.

A.2.1. Anclas localizadas al final del texto

En las Tablas A.1, A.2 y A.3 presentamos los resultados del experimento para las configuraciones de anclas localizadas al final del texto, con pesos de la red de fonemas -5, -6 y -7 respectivamente. El valor máximo de medida F obtenido es de 0.6287 y se consigue para un tiempo de guarda de 10 segundos, peso de la red de fonemas -7 y un número de palabras consecutivas correctas para considerar ancla igual a 2.

A.2.2. Anclas localizadas al principio del texto

De forma análoga a la sección anterior, en las Tablas A.4, A.5 y A.6 se presentan los resultados del experimento, esta vez considerando las anclas localizadas al principio del texto, para los pesos de la red de fonemas -5, -6 y -7 respectivamente. En esta ocasión el máximo valor de F es de 0.6221 y se da para un tiempo de guarda de 10 segundos, peso de la red de fonemas -7 y un número de palabras correctas consecutivas para considerar ancla igual a 2.

Tiempo de guarda(s)	Palabras correctas consecutivas para considerar ancla	Precision	Recall	F
10	2	**0.6167**	**0.6295**	**0.6230**
	3	0.6200	0.6037	0.6117
	4	0.6296	0.5674	0.5969
	5	0.6423	0.5210	0.5754
25	2	0.6172	0.6179	0.6176
	3	0.6208	0.5972	0.6088
	4	0.6302	0.5641	0.5953
	5	0.6412	0.5204	0.5745
50	2	0.6213	0.6065	0.6138
	3	0.6217	0.5897	0.6053
	4	0.6307	0.5599	0.5932
	5	0.6415	0.5181	0.5732
75	2	0.6277	0.6014	0.6143
	3	0.6239	0.5839	0.6033
	4	0.6306	0.5544	0.5900
	5	0.6417	0.5170	0.5727

Tabla A.1: Medida F, *precision* y *recall* para diferentes configuraciones de anclas al final del texto con peso de la red de fonemas -5

Tiempo de guarda(s)	Palabras correctas consecutivas para considerar ancla	Precision	Recall	F
10	2	0.6177	0.6116	0.6146
	3	0.6155	0.6244	0.6199
	4	0.6223	0.5952	0.6085
	5	0.6271	0.5639	0.5938
25	2	**0.6186**	**0.6312**	**0.6249**
	3	0.6167	0.6183	0.6175
	4	0.6236	0.5942	0.6085
	5	0.6274	0.5629	0.5934
50	2	0.6227	0.6205	0.6216
	3	0.6200	0.6131	0.6165
	4	0.6234	0.5906	0.6066
	5	0.6287	0.5608	0.5928
75	2	0.6244	0.6126	0.6184
	3	0.6222	0.6096	0.6158
	4	0.6250	0.5880	0.6059
	5	0.6290	0.5606	0.5928

Tabla A.2: Medida F, *precision* y *recall* para diferentes configuraciones de anclas al final del texto con peso de la red de fonemas -6

Como se observa se encuentra ligeramente por debajo del valor obtenido para la configuración con anclas al final del texto, lo cual es estadisticamente insignificante y no nos permite afirmar el mejor o peor funcionamiento de unas u otras anclas.

Tiempo de guarda(s)	Palabras correctas consecutivas para considerar ancla	Precision	Recall	F
10	2	**0.6137**	**0.6445**	**0.6287**
	3	0.6104	0.6375	0.6237
	4	0.6117	0.6257	0.6186
	5	0.6139	0.6061	0.6100
25	2	0.6156	0.6385	0.6269
	3	0.6117	0.6329	0.6221
	4	0.6117	0.6237	0.6176
	5	0.6137	0.6056	0.6096
50	2	0.6203	0.6295	0.6249
	3	0.6142	0.6272	0.6206
	4	0.6133	0.6209	0.6171
	5	0.6138	0.6048	0.6093
75	2	0.6239	0.6222	0.6230
	3	0.6172	0.6235	0.6204
	4	0.6134	0.6168	0.6151
	5	0.6147	0.6026	0.6086

Tabla A.3: Medida F, *precision* y *recall* para diferentes configuraciones de anclas al final del texto con peso de la red de fonemas -7

Tiempo de guarda(s)	Palabras correctas consecutivas para considerar ancla	Precision	Recall	F
10	2	**0.6072**	**0.6220**	**0.6145**
	3	0.6104	0.6004	0.6053
	4	0.6220	0.5757	0.5980
	5	0.6355	0.5397	0.5837
25	2	0.6093	0.6110	0.6101
	3	0.6116	0.5950	0.6031
	4	0.6220	0.5728	0.5964
	5	0.6353	0.5386	0.5830
50	2	0.6139	0.6019	0.6079
	3	0.6116	0.5858	0.5984
	4	0.6230	0.5673	0.5939
	5	0.6363	0.5347	0.5811
75	2	0.6166	0.5941	0.6051
	3	0.6146	0.5828	0.5983
	4	0.6239	0.5619	0.5913
	5	0.6364	0.5331	0.5802

Tabla A.4: Medida F, *precision* y *recall* para diferentes configuraciones de anclas al principio del texto con peso de la red de fonemas -5

Tiempo de guarda(s)	Palabras correctas consecutivas para considerar ancla	Precision	Recall	F
10	2	**0.6048**	**0.6325**	**0.6183**
	3	0.6061	0.6219	0.6139
	4	0.6141	0.6035	0.6088
	5	0.6218	0.5813	0.6009
25	2	0.6064	0.6230	0.6146
	3	0.6072	0.6168	0.6120
	4	0.6146	0.6013	0.6079
	5	0.6223	0.5806	0.6007
50	2	0.6107	0.6154	0.6130
	3	0.6116	0.6109	0.6112
	4	0.6154	0.5982	0.6067
	5	0.6237	0.5791	0.6006
75	2	0.6159	0.6112	0.6135
	3	0.6137	0.6082	0.6109
	4	0.6161	0.5954	0.6056
	5	0.6253	0.5732	0.5981

Tabla A.5: Medida F, *precision* y *recall* para diferentes configuraciones de anclas al principio del texto con peso de la red de fonemas -6

Tiempo de guarda(s)	Palabras correctas consecutivas para considerar ancla	Precision	Recall	F
10	2	**0.6067**	**0.6382**	**0.6221**
	3	0.6056	0.6335	0.6192
	4	0.6079	0.6258	0.6167
	5	0.6119	0.6166	0.6142
25	2	0.6076	0.6318	0.6195
	3	0.6063	0.6292	0.6175
	4	0.6082	0.6244	0.6162
	5	0.6123	0.6163	0.6143
50	2	0.6118	0.6259	0.6188
	3	0.6095	0.6255	0.6174
	4	0.6110	0.6218	0.6164
	5	0.6127	0.6157	0.6142
75	2	0.6151	0.6226	0.6188
	3	0.6113	0.6237	0.6175
	4	0.6113	0.6200	0.6156
	5	0.6142	0.6118	0.6130

Tabla A.6: Medida F, *precision* y *recall* para diferentes configuraciones de anclas al principio del texto con peso de la red de fonemas -7

Bibliografía

[1] Vivolab, sitio web: Pagina principal. `http://vivolab.unizar.es/`, 2015 (Acceso: 20 Abril 2016).

[2] Mgb challenge, sitio web: Pagina principal. `http://www.mgb-challenge.org/`, 2016 (Acceso: 20 Abril 2016).

[3] Xuedong Huang, Alejandro Acero, and Hsiao-Wuen Hon. *Spoken language processing: a guide to theory, algorithm, and system development*. Prentice Hall PTR, 2001.

[4] Steve J. Young, D. Kershaw, J. Odell, D. Ollason, V. Valtchev, and P. Woodland. *The HTK Book Version 3.4*. Cambridge University Press, 2006.

[5] Sun Developer Network (SDN). Java grammar format specification. `https://www.w3.org/TR/jsgf/`.

[6] Mehryar Mohri, Fernando Pereira, and Michael Riley. Weighted finite-state transducers in speech recognition. *Computer Speech & Language*, 16(1):69–88, 2002.

[7] Nist sclite scoring package version 1.5 - icsi. `http://www1.icsi.berkeley.edu/Speech/docs/sctk-1.2/sclite.htm`, 2000 (Acceso: 20 Mayo 2016).

[8] Thomas Hain, NST MGB Organisers. Mgb challenge 2015 task 2 alignment of broadcast audio to a subtitle file. `http://data.cstr.ed.ac.uk/asru/scoring/eval_task2.pdf`, 2015.

[9] P Bell, MJF Gales, T Hain, J Kilgour, P Lanchantin, X Liu, A McParland, S Renals, O Saz, M Wester, et al. The MGB challenge: Evaluating multi-genre broadcast media recognition. *Proc. of ASRU, Arizona, USA*, 2015.

[10] Richard C Rose and Douglas B Paul. A hidden markov model based keyword recognition system. In *Acoustics, Speech, and Signal Processing, 1990. ICASSP-90., 1990 International Conference on*, pages 129–132. IEEE, 1990.

[11] Jose Enrique Garcia, Alfonso Ortega, Eduardo Lleida, Tomas Lozano, Emiliano Bernues, Daniel Sanchez. Audio and text synchronization for tv news subtitling based on automatic speech recognition. 2008.

[12] Audacity, sitio web: pagina principal. `http://www.audacityteam.org/`, 2016 (Acceso: 1 Junio 2016).