Universidad de Buenos Aires
Facultad de Ciencias Exactas y Naturales
Departamento de Computación

Desarrollo y evaluación de sistemas de calificación de la pronunciación basados en redes neuronales

Tesis presentada para optar al título de
Licenciada en Ciencias de la Computación

Cyntia Bonomi

Directora: Dra. Luciana Ferrer

Codirectora: Lic. Jazmín Vidal Domínguez

Buenos Aires, 2022

RESUMEN

Los sistemas de calificación de la pronunciación son una herramienta importante para el aprendizaje de idiomas. Estos sistemas evalúan la manera en que se producen los sonidos del lenguaje a lo largo de párrafos, oraciones, palabras o fonos (los sonidos individuales que componen las palabras). Los sistemas de calificación de la pronunciación permiten interactuar de manera incansable con los estudiantes, indicando errores en tiempo real y posibilitando a cada alumno avanzar a su propio ritmo. Actualmente, los sistemas que califican la pronunciación a nivel párrafo u oración presentan rendimientos similares a los de docentes humanos. Sin embargo, los sistemas que califican a nivel fono todavía están lejos de tener un funcionamiento que les permita ser utilizados en un escenario educativo real.

Una manera de implementar sistemas de calificación de la pronunciación es usando como base sistemas de Reconocimiento Automático del Habla (RAH). Los sistemas RAH resuelven la tarea de estimar la transcripción ortográfica de una señal de habla. Estos sistemas pueden ser usados para calificar pronunciación usando el método llamado Goodness of pronunciation (GOP). Este método consiste en utilizar un sistema RAH entrenado con habla nativa de la población de interés para estimar las probabilidades a posteriori de los sonidos que el estudiante debiera haber pronunciado. Se asume que estas probabilidades serán bajas cuando la pronunciación sea incorrecta, ya que las características de la señal no coincidirán con lo que el modelo entrenado con hablantes nativos espera encontrar.

En los últimos años, a partir del desarrollo de algoritmos de optimización eficientes y de mejoras en el hardware, las redes neuronales profundas (DNN) han mejorado la resolución de múltiples tareas. En particular, han mejorado notablemente el rendimiento de los sistemas RAH y, en consecuencia, el rendimiento de los sistemas de calificación de la pronunciación que los usan como base. Estos resultados, junto con la falta de sistemas que califiquen a nivel de fono con resultados comparables a los de docentes humanos, motivan la exploración de sistemas de calificación de la pronunciación a nivel fono basados en DNNs. En esta tesis estudiamos un sistema de calificación de la pronunciación a nivel fono basado en DNNs. Entrenamos diferentes modelos con el fin de encontrar la red con mejor rendimiento. Estos modelos utilizan distintas características para representar la señal, y diversas arquitecturas e hiperparámetros de entrenamiento. Comparamos la mejor red con un sistema basado en maquinas de soporte vectorial (SVM) desarrollado previamente en el grupo de investigación de la Dra. Ferrer. Cabe destacar que, el sistema basado en SVM, a diferencia del sistema desarrollado en esta tesis utiliza datos no nativos para su entrenamiento. Sin embargo, el sistema basado en DNN obtiene mejores resultados.

Actualmente, la mayoría de las redes utilizadas en sistemas de calificación de pronunciación son redes entrenadas para sistemas RAH. Estas redes requieren de un gran número de parámetros.

Los resultados obtenidos en esta tesis sugieren que es posible utilizar redes más pequeñas para resolver la tarea de calificación de la pronunciación. Por otro lado, si bien llegamos a mejores resultados que los dados por el sistema SVM, el rendimiento del sistema aún no es óptimo para ser utilizado en escenarios educativos reales. Hay mucho trabajo futuro por hacer en esta tarea.

Palabras claves: aprendizaje de idiomas asistido por computadora, calificación de la pronunciación, redes neuronales, recursos escasos

Resumen

Pronunciation scoring systems are an important tool for language learning. These systems grade the way in which sounds are produced within paragraphs, sentences, words or phones (the individual sounds that form words). Pronunciation scoring systems enable constant interaction with students, highlighting errors in real time and allowing the student to progress at his or her own rate. Currently, the systems that score pronunciation at the paragraph or sentence level produce similar results to human teachers. However, the systems that score at the phone level are still far from being able to be used in a real teaching scenario.

One way to implement pronunciation scoring systems is using Automatic Speech Recognition (ASR) systems as a base. The ASR systems fulfill the task of estimating orthographic transcriptions from speech utterances. These systems can be used to score pronunciation using the method called Goodness of pronunciation (GOP). This method consists of an ASR system trained with native utterances from the language to be taught to estimate posteriors for each phone in the target language. It is assumed that these posteriors would be low when the pronunciation is incorrect, given the fact that the characteristics of the speech signal would not match those in the native model of the ASR system.

In recent years, due to the development of efficient optimization algorithms and improvements in hardware, Deep Neural Networks (DNN) have improved a wide variety of tasks. In particular, they have notably improved the performance of ASR systems and, consequently, the performance of pronunciation scoring systems that use them as a base. These results, together with the absence of systems that score on a phone level with results similar to human teachers, encourage the research on phone level pronunciation scoring systems based on DNNs. In this dissertation we study a pronunciation scoring system at the phone level based on DNNs. We train different models in order to find the network with the best performance. These models use different acoustic features to represent the utterance and diverse training architectures and hyperparameters. The DNN with the best performance is compared with a system based on support vector machines (SVM) previously developed in the research group of Dr. Ferrer. It is worth noting that, the SVM based system, as opposed to the system developed in this dissertation, uses non-native data for its training. However, the DNN based system yields better results.

Nowadays, most networks used in pronunciation scoring systems are networks trained for ASR systems. These networks require a large number of parameters. The results obtained in this dissertation suggest that it is possible to use smaller networks to solve the task of scoring pronunciation. In this particular case, although the GOP-DNN system yields better results than the SVM system, its performance is not yet adequate enough to be used in real educational

scenarios. There is still much work to be done in this area.

Keywords: computer-assisted language learning, pronunciation scoring, neural networks, low resources

Índice general

1. INTRODUCCIÓN

La presente tesis de licenciatura se encuadra en el marco de un Proyecto de Investigación Científica y Tecnológica (PICT) de la Dra. Ferrer financiado por la Agencia Nacional de la Promoción Científica y Tecnológica que busca desarrollar un sistema de asistencia computarizada de aprendizaje de idiomas de acceso gratuito para niños y adultos argentinos en proceso de aprendizaje del idioma inglés. La principal contribución de la tesis al proyecto es la implementación de un sistema de calificación de la pronunciación basado en redes neuronales que mejora el rendimiento en comparación con el sistema previamente utilizado en el grupo.

1.0.1. Descripción del problema y motivación

En la actualidad, el dominio del idioma inglés está vinculado a perspectivas de acceso a la educación superior, de inserción laboral y de movilidad social [46, 6]. En Argentina, las escuelas primarias están obligadas a enseñar al menos un idioma extranjero. Sin embargo, la carga horaria dedicada a estos efectos es reducida y suele no ser suficiente para una práctica exhaustiva. Esto tiene un impacto directo en los sectores más vulnerables y de bajos ingresos, donde la población no tiene la posibilidad de asistir a escuelas privadas o contratar clases de apoyo para complementar la educación pública.

Hace décadas se estudia cómo desarrollar sistemas que brinden asistencia computarizada para el aprendizaje de idiomas (ACAI). Estos sistemas tienen la capacidad de interactuar de manera incansable con el estudiante, indicando errores en tiempo real y permitiendo a cada alumno avanzar a su propio ritmo. Cuando además, se involucran juegos o actividades interactivas, los sistemas ACAI tienen la posibilidad de generar interés y estimular la práctica de idiomas como complemento a las tareas que puedan ser indicadas en el aula por un docente [33, 23, 45]. Si bien existen y están en uso sistemas computarizados que brindan este tipo de apoyo en el campo del aprendizaje de idiomas, uno de los problemas para su distribución masiva es que suelen ser pagos o con acceso gratuito a un subconjunto acotado de herramientas, excluyendo de su uso a aquellos sectores que más podrían necesitarlos.

Los sistemas ACAI abarcan diferentes aspectos de la enseñanza de un segundo idioma, entre ellos, los orientados a la gramática, el vocabulario y la pronunciación. Los sistemas que se encargan de la gramática, están enfocados en la enseñanza de las estructuras oracionales, mientras que los que se encargan del vocabulario se enfocan en la enseñanza del conjunto de palabras de una lengua. Aquellos orientados a la pronunciación utilizan, en su mayoría, técnicas de reconocimiento automático del habla (RAH) para grabar lo dicho por un estudiante y evaluar la manera en que producen los sonidos del lenguaje a lo largo de párrafos, oraciones, palabras o fonos, los

sonidos individuales que componen las palabras.

La utilidad de los sistemas ACAI ha sido demostrada en diversos trabajos científicos [46, 32]. En particular, han probado ser útiles aquellos sistemas que califican la pronunciación a nivel palabra o fono en etapas iniciales del aprendizaje dónde los estudiantes presentan dificultades para identificar sus propios errores de pronunciación [7, 25].

Mientras que los sistemas que califican la pronunciación a nivel párrafo u oración presentan rendimientos similares a los de docentes humanos, los sistemas que califican a nivel fono todavía están lejos de tener un funcionamiento óptimo que les permita ser utilizados en un escenario educativo real [35]. Sin embargo, en los últimos años, a partir del desarrollo de algoritmos eficientes de optimización y de mejoras en el hardware, las redes neuronales profundas han mejorado la resolución de múltiples tareas, entre ellas la de los sistemas RAH en los que estos sistemas se basan [12], abriendo posibilidades alentadoras para el trabajo futuro en el área de calificación automática de la pronunciación.

1.0.2. Trabajos previos

Los sistemas ACAI encargados de calificar la pronunciación a nivel fono se pueden dividir en dos familias: la que solo utiliza para entrenar el sistema bases de datos de habla nativa del idioma que se desea calificar [34, 41, 20, 11, 50, 13, 16] y la que se entrena o adapta utilizando, además, bases de datos de habla no nativa producida por la población de estudiantes de dicho idioma [49, 22, 9, 14, 16, 17]. Mientras que los métodos de la primera familia son más genéricos y relativamente baratos de implementar dado que no requieren de la recolección de datos específicos para cada población de hablantes, los métodos de la segunda familia generalmente resultan en sistemas de mejor rendimiento. Sin embargo, tienen como desventaja que las bases de datos que utilizan para su entrenamiento suelen no ser públicas y, si lo son, suelen ser demasiado pequeñas como para entrenar estos sistemas sin riesgo de sobre ajuste.

A partir de diferentes medidas probabilísticas, los métodos de la primera familia calculan cuán similar resulta la emisión de un estudiante del idioma respecto de un determinado modelo de habla nativa. Estos métodos, suelen utilizar sistemas RAH entrenados con habla nativa de la población de interés para estimar las probabilidades a posteriori de los sonidos que el estudiante debería haber pronunciado. Asumen que estas probabilidades serán bajas cuando la pronunciación sea incorrecta, ya que las características de la señal no coincidirán con lo que el modelo entrenado con hablantes nativos espera encontrar. Estos sistemas están formados por dos etapas. En la primera etapa, el sistema RAH se utiliza para alinear la frase dicha por el estudiante a la transcripción de lo dicho. En la segunda etapa, para cada fono en los alineamientos se computa un puntaje dado por el logaritmo de la probabilidad del fono dadas las características de entrada en ese segmento, normalizado por la cantidad de bloques que componen al segmento. Estos bloques son llamados en inglés *frames*. En [20], se presenta uno de los primeros métodos de este grupo,

donde la probabilidad se calcula utilizando las verosimilitudes generadas por el modelo acústico del RAH, el cual, en los sistemas clásicos, está dado por una colección de modelos de mezclas de Gaussianas (GMMs). Para cada fono el puntaje $\hat{l}$ se define como:

$$\hat{l} = \frac{1}{t_e - t_s + 1} \sum_{t=t_s}^{t_e} \log p(y_t|q_i) \tag{1.1}$$

donde $p(y_t|q_i)$ es la probabilidad de la observación y_t dado el fono q_i en el tiempo t dividida por $t_e - t_s + 1$, su duración en *frames*. Un segundo método de este grupo también propuesto en [20], muestra que se obtiene un mejor rendimiento si se transforman las verosimilitudes originalmente obtenidas a partir del sistema RAH en probabilidades a posteriori de la pronunciación correcta de un fono dada una observación. Aplicando el teorema de Bayes sobre las salidas del sistema RAH se define:

$$P(q_i|y_t) = \frac{p(y_t|q_i)P(q_i)}{\sum_{j=1}^{M} p(y_t|q_j)P(q_j)} \tag{1.2}$$

donde M es el conjunto de modelos independientes de contexto de los fonos y $P(q_i)$ es la probabilidad a priori de la clase del fono q_i. De este modo el puntaje $\hat{\rho}$ a la calidad de la pronunciación para cada fono esta dado por:

$$\hat{\rho} = \frac{1}{t_e - t_s + 1} \sum_{t=t_s}^{t_e} \log P(q_i|y_t) \tag{1.3}$$

con $P(q_i|y_t)$ la probabilidad a posteriori de la observación q_i dado la observación y_t en el tiempo t. Finalmente, un tercer método de este grupo llamado Goodness of Pronunciation (GOP) [50], propone utilizar como puntaje el logaritmo de la probabilidad a posteriori del fono q_i dadas las características de entrada en ese segmento $Y_i = [y_{t_s}, ..., y_{t_e}]$, normalizado por su longitud. En este enfoque también se utiliza el teorema de Bayes para calcular la probabilidad a posteriori y se define como:

$$GOP(q_i) = |\log(P(q_i|Y_i))|/(t_e - t_s + 1) = \left| \log \left(\frac{p(Y_i|q_i)P(q_i)}{\sum_{j=1}^{J} p(Y_i|q_j)P(q_j)} \right) \right| \Big/ (t_e - t_s + 1) \tag{1.4}$$

donde J es la cantidad de fonos y $t_e - t_s + 1$ el número de *frames* en el segmento acústico Y_i. En este puntaje, tanto en el numerador como en el denominador las probabilidades se calculan a nivel de segmento y luego se aplica el logaritmo al cociente entre las probabilidades. En contraste, en el puntaje de la Ecuación 1.3 se calcula la suma del logaritmo del cociente de las probabilidades calculadas a nivel de frame. En ambos puntajes se divide por la duración del fono d. El método GOP es el elegido para el desarrollo de esta tesis y se explica en detalle en la sección 2.

Los métodos de la segunda familia que se entrenan o adaptan utilizando bases de datos de habla no nativa están entrenados directamente para distinguir pronunciaciones correctas e incorrectas utilizando ejemplos del habla de los estudiantes de la lengua que se desea enseñar. Un método de la segunda familia que ha probado ser superior a los métodos de la primera

familia recién descriptos, consiste en generar modelos de mezclas de Gaussianas para representar la distribución de ciertas características espectrales encontradas en cada fono del idioma de interés cuando el fono es pronunciado correcta e incorrectamente por un estudiante [10]. Al igual que en los métodos anteriores, en este enfoque se requiere de una etapa inicial de alineamiento forzado entre las señales de habla y su transcripción. Dos modelos de mezclas de Gaussianas son entrenados a partir de una base de datos no nativa etiquetada a nivel fono con calidad de pronunciación. Un modelo es entrenado con las pronunciaciones correctas, similares a las nativas del fono, mientras que el otro se entrena con las pronunciaciones incorrectas o pronunciaciones no nativas del mismo fono. Una vez entrenados los modelos para pronunciaciones correctas e incorrectas el puntaje se obtiene como el logaritmo del cociente de la verosimilitud entre los dos modelos λ_M y λ_C de la siguiente manera:

$$\mathrm{LLR}(q_i) = \frac{1}{t_e - t_s + 1} \sum_{t=t_s}^{t_e} \log \frac{p(y_t|q_i, \lambda_M)}{p(y_t|q_i, \lambda_C)} \tag{1.5}$$

donde $t_e - t_s + 1$ es la duración en *frames* del fono q_i, y_t es el vector de observación en el *frame* t, t_s es el *frame* donde comienza el fono y t_e donde termina. De forma similar, en [9] se utiliza un método que consiste en entrenar por cada fono un único modelo GMM con todas las instancias bien y mal pronunciadas para obtener modelos independientes de clase llamados en inglés *Universal Backround Models* (UBM). A continuación, dicho modelo general por fono es adaptado a los fonos bien y mal pronunciados para generar modelos correctos e incorrectos específicos en cada caso. Finalmente, ambos modelos se utilizan para calcular el LLR dado en la ecuación (1.5). Un último método propuesto en el mismo trabajo, consiste en desarrollar un sistema a partir del modelo UBM para cada fono. Por cada instancia de fono se obtiene un nuevo modelo GMM adaptando los parámetros del UBM al vector de características acústicas que representa al fono. Las medias y los pesos del GMM adaptado son apiladas para obtener un supervector. Los supervectores son usados como características de entrada para un sistema de clasificación basado en máquinas de soporte vectorial (SVM). Este sistema será utilizado para comparar con el sistema desarrollado en esta tesis basado en DNN. Ambos sistemas serán explicados en detalle en la sección métodos. Los dos últimos enfoques propuestos en [9] demostraron ser superiores al sistema de calificación de la pronunciación basado en LLR calculado a partir de GMMs entrenados independientemente propuesto en [10].

En los últimos años, el uso de modelos basados en DNNs ha resultado en grandes mejoras en el rendimiento de los sistemas que resuelven diversas tareas. En particular, las redes neuronales han mejorado notablemente el rendimiento de los sistemas RAH [12]. Como consecuencia, la aplicación de estas herramientas se ha incorporado recientemente también a la tarea de puntuación de la pronunciación, mostrando gran potencial. Ya existen implementaciones de estas técnicas dentro de los métodos de la primera y de la segunda familia. Por ejemplo, en el caso de la primera familia, Hu [16] propone una reimplementación del GOP clásico usando DNNs. En este método,

el modelo acústico tradicional basado en GMMs es reemplazado por un DNN entrenado, como los GMMs, con habla nativa para la tarea de reconocimiento de habla. Los puntajes de calificación de la pronunciación son calculados a partir de las probabilidades a posteriori devueltas por la red neuronal para cada fono. Esta manera de calcular GOP usando un modelo DNN en lugar de GMMs resulta en mejoras en el rendimiento en la tarea de puntuación de la pronunciación. Este método es explicado en detalle en la sección 2 ya que es el método bajo estudio en este trabajo.

Otro método propuesto por Huang en [17], pero perteneciente a la segunda familia, consiste en agregar a la salida de la DNN un módulo entrenado con datos de hablantes no nativos anotados con calidad de la pronunciación para distinguir pronunciaciones correctas de incorrectas usando regresión logística lineal. Por otro lado, Hu [16] también utiliza un sistema RAH basado en DNNs y, similar al caso anterior, sobre la salida del modelo acústico implementa un clasificador binario para cada fono basado en redes neuronales usando como entrada las probabilidades a posteriori generadas por la red del RAH y una serie de valores calculados en base a ellas. A partir de estos, otros trabajos [51, 31, 26, 1] han utilizado distintas variantes de arquitecturas DNN seguidas por otro modelo de red neuronal no profunda para la tarea de puntuación de la pronunciación. En este trabajo no vamos a explorar esta dirección porque buscamos métodos de la primera familia que no requieran datos anotados para la tarea. Sin embargo, a pesar de que nos concentramos en métodos de la primera familia, vamos a comparar con un método de la segunda familia que fue implementado como primera solución al problema en el grupo de investigación de la Dra. Ferrer.

Finalmente, en el último tiempo ha surgido un enfoque en el área de reconocimiento automático del habla, en el que se utilizan arquitecturas llamadas de extremo a extremo (E2E) [5, 36, 30, 48, 4, 24, 8, 27]. Los sistemas E2E utilizan una red integrada que recibe la señal de habla y genera resultados de acuerdo a la tarea que se quiere resolver. En estos sistemas el problema de alineamiento está integrado en el proceso de entrenamiento. La ventaja de los modelos E2E reside en poder entrenar todos los módulos juntos optimizando la tarea objetivo. Este tipo de sistemas, quedan por fuera del alcance de esta tesis.

2. METODOS

En esta sección primero presentamos una breve explicación de la tarea de reconocimiento automático del habla (RAH) con foco en los sistemas RAH de los que hacen uso los sistemas de calificación de la pronunciación presentados en esta tesis. Una descripción más detallada del tema se puede encontrar en [39, 44]. Luego introducimos los dos sistemas de calificación de la pronunciación que componen este trabajo. El primero, un sistema de supervectores modelados con máquinas de soporte de vectores (SVM) basado en [9] que pertenece a la familia de sistemas que usa datos no nativos anotados con calidad de pronunciación para el entrenamiento. Este sistema es usado para comparar con el segundo sistema, el algoritmo de Goodness of Pronunciation basado en DNNs. Este sistema fue propuesto en [14], no requiere datos anotados con calidad de pronunciación para su entrenamiento y será nuestro sistema principal, sobre el que realizaremos experimentos. Por ultimo se describen las métricas utilizadas para evaluar el desempeño de los sistemas.

2.1. Reconocimiento automático del habla

El reconocimiento automático de habla (RAH) resuelve la tarea de estimar, dada una señal de habla, su transcripción ortográfica. En los sistemas RAH clásicos, la señal de habla es transformada por un extractor de características en una secuencia de vectores acústicos $Y = [y_1, y_2, ..., y_T]$. Cada vector es una representación compacta del espectro en una cierta región. Usualmente están dados por coeficientes en la escala de frecuencia Mel, llamados *Mel frequency cepstral coefficients* (MFCC). Estos coeficientes son descriptos en detalle en la siguiente sección. El texto, por su parte, es representado como una secuencia discreta de palabras $w = [w_1, w_2, ...w_N]$ pertenecientes a un vocabulario. Los sistemas RAH tienen como objetivo inferir cuál es la secuencia de palabras $w*$ más probable dado un vector de características acústicas Y, es decir:

$$w^* = \arg\max_w P(w|Y) \tag{2.1}$$

Como en un escenario clásico la probabilidad a posteriori $P(w|Y)$ es difícil de modelar directamente se aplica el teorema de Bayes para resolver el problema equivalente:

$$w^* = \arg\max_w \frac{p(Y|w)P(w)}{p(Y)} = \arg\max_w p(Y|w)P(w) \tag{2.2}$$

Donde $P(w)$ representa la probabilidad a priori de observar w independientemente de la señal de entrada y es determinada por el modelo de lenguaje y la verosimilitud $p(Y|w)$ de observar una determinada señal de entrada dada una secuencia de palabras está dada por el modelo acústico.

En las siguientes secciones, primero explicamos en más detalle las MFCCs, luego los componentes de los sistemas RAH: modelo acústico y modelo de lenguaje. Por último, explicamos los alineamientos forzados realizados a partir de sistemas RAH que serán utilizados en el sistema desarrollado en esta tesis.

2.1.1. Mel frequency cepstral coefficients (MFCC's)

Las características más usadas en procesamiento de habla son las *Mel frequency cepstral coefficients* (MFCC). La base del cálculo de estos coeficientes es la suposición de que, en periodos cortos de tiempo, la señal de habla permanece estacionaria, es decir, sus características espectrales se mantienen constantes. Esto permite segmentar a la señal en bloques de pocos milisegundos, llamados en inglés *frames*. Para cada frame se calculan coeficientes siguiendo los pasos detallados a continuación:

1. Pre-énfasis:
   Se aplica a la señal un filtro pasa alto para amplificar las frecuencias altas y disminuir las bajas. Este filtro permite equilibrar el espectro de frecuencias ya que en el habla las frecuencias altas tienen menor amplitud que las bajas.

2. Segmentación y ventaneo:
   En este paso se segmenta la señal en *frames*. En general, en los sistemas de reconocimiento de habla la señal se divide en *frames* utilizando una ventana de duración de 25 ms y un desplazamiento entre ventanas de 10 ms.

3. Transformada rápida de Fourier (FFT):
   En este paso, por cada *frame* se calcula la transformada de Fourier para pasar del dominio del tiempo al dominio de la frecuencia.

4. Banco de filtros en escala Mel:
   El espectro obtenido en el paso anterior se pasa a través de filtros triangulares en la escala Mel. La escala Mel tiene como objetivo imitar la percepción humana, siendo más discriminatoria en las frecuencias bajas y menos discriminatoria en las frecuencias altas. Usualmente se utilizan entre 20 y 40 filtros. Luego, debido a que los humanos no escuchamos la intensidad de los sonidos en una escala lineal, se aplica el logaritmo natural a los valores obtenidos después de pasar los filtros triangulares.

5. Transformada discreta de coseno (DCT):
   Los coeficientes del banco de filtros calculados en el paso anterior están correlacionados, esto puede traer problemas en el aprendizaje de los modelos. Para evitar esta correlación se aplica la transformada discreta de coseno que decorrelaciona los coeficientes del ban-

co de filtros Mel. Estos coeficientes decorrelacionados son las llamadas MFCCs. Para el reconocimiento automático del habla se suelen utilizar 13.

6. Deltas y doble deltas
   Las MFCCs describen adecuadamente las características de cada *frame* de la señal de habla. Sin embargo, es posible agregar información sobre la variación de la señal anexando a cada coeficiente su primera derivada (deltas) y su segunda derivada (doble delta).

2.1.2. Modelo acústico

El modelo acústico es usado para generar la probabilidad $p(Y|w)$ siendo Y una secuencia de vectores acústicos y w una secuencia de palabras. La unidad mínima de sonido representada por el modelo acústico es el fono. Cada palabra del vocabulario es mapeada a la cadena de fonos que la compone utilizando un diccionario, típicamente confeccionado a mano, llamado lexicón. Por ejemplo, la palabra del inglés "cat" está compuesta por la secuencia de tres fonos expresados en símbolos IPA (*International Phonetic Alphabet*) como /k//ae//t/. Cada fono es representado utilizando un modelo oculto de Markov (HMM). A su vez, los modelos de fonos pueden ser concatenados para formar palabras y los modelos de palabras pueden concatenarse para formar frases. En la siguientes secciones explicamos los modelos HMM y su uso en los sistemas RAH.

Modelos ocultos de Markov (HMM)

Los HMMs son modelos estadísticos compuestos por un conjunto de estados y transiciones que los conectan. Estos modelos son útiles para representar señales dependientes del tiempo. Partiendo de un estado inicial, se transiciona a otro estado con cierta probabilidad y, por cada estado visitado, se genera una observación con una cierta distribución de probabilidad. En procesamiento del habla cada fono del inventario de un idioma se modela como un HMM, en esos casos se usan HMMs de primer orden, en los que la probabilidad del estado actual depende únicamente del estado anterior. Además, los fonos son modelados con HMMs que solo tienen transiciones de izquierda a derecha o dentro del mismo estado. La figura 2.1 muestra un ejemplo de modelo HMM de primer orden con transiciones de izquierda a derecha para un fono. En cada tiempo t un modelo HMM está en cierto estado. En el siguiente tiempo, puede pasar al siguiente estado o quedarse en el mismo. En cada tiempo t se genera una observación y_t con distribución de probabilidad $b_j(y_t)$, donde b_j es la distribución de probabilidad del estado j, el estado en el que está el modelo en el tiempo t. Finalmente, las transiciones entre estados están dadas por distribuciones de probabilidad de transición a_{ij}.

A partir de la concatenación de HMMs de fonos es posible armar los HMMs de palabras, que siguen siendo HMMs de primer orden de izquierda a derecha. Luego, dada una transcripción, se puede armar el HMM de la frase concatenando los HMMs de las palabras en la transcripción.

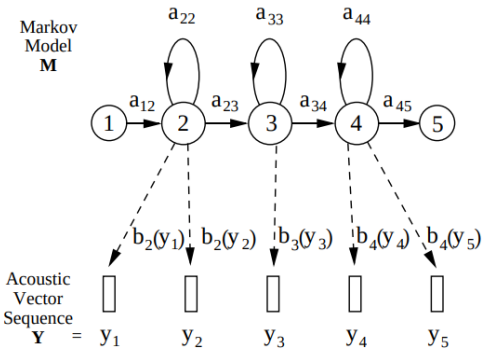

Fig. 2.1: Tomado de [44]. Ejemplo de modelo basado en HMM de primer orden de izquierda a derecha. Los estados del modelo se encuentran conectados por transiciones probabilísticas. Por cada período de tiempo se cambia a un nuevo estado generando un vector acústico de características de acuerdo a la distribución de emisión asociada al estado.

Los modelos HMM se entrenan para ajustar las probabilidades de transición entre estados y la distribución de emisión maximizando la probabilidad de las secuencias de observaciones para un corpus de entrenamiento con audios y sus transcripciones. Para esta tarea se suele utilizar el método de Baum-Welch (el algoritmo de *expectation–maximization* (EM) aplicado a los HMM) como se explica en [39].

Una vez obtenidos los parámetros del modelo dos problemas pueden ser abordados. El primero, conocido como el problema de evaluación, consiste en obtener la probabilidad de la secuencia de observaciones y puede ser resuelto por el algoritmo *forward-backward*. El segundo consiste en obtener la secuencia de estados ocultos más probable dada una secuencia de observaciones. Este problema es conocido como el problema de decodificación y es resuelto por el algoritmo de decodificación de Viterbi. Para una descripción detallada del uso de estos dos algoritmos en sistemas HMM ver [39].

Senones

Como explicamos anteriormente, en los HMM utilizados para reconocimiento de habla, las palabras del lenguaje son representadas a partir de la concatenación de los HMMs de los fonos que la componen. Cada fono puede ser modelado como si fuera estacionario en tres partes representadas por tres estados: inicio, medio y fin. En la figura 2.1 podemos ver un ejemplo de un fono modelado con tres estados 2, 3 y 4. El estado 1 y el estado 5 corresponden al estado de entrada y salida respectivamente. Sin embargo, los fonos cambian según el contexto en el que aparecen. Por ejemplo, la /g/ inicial de la palabra gato suena distinto que la /g/ entre vocales de la palabra agua. Para lograr una buena discriminación fonética es necesario entrenar un HMM por cada

fono con su contexto, llamados trifonos. Sin embargo la falta de datos de entrenamiento y la gran cantidad de posibles trifonos dificulta la obtención de modelos de trifonos bien entrenados. Una manera de sortear este problema es agrupar los estados de los trifonos por similitud acústica.

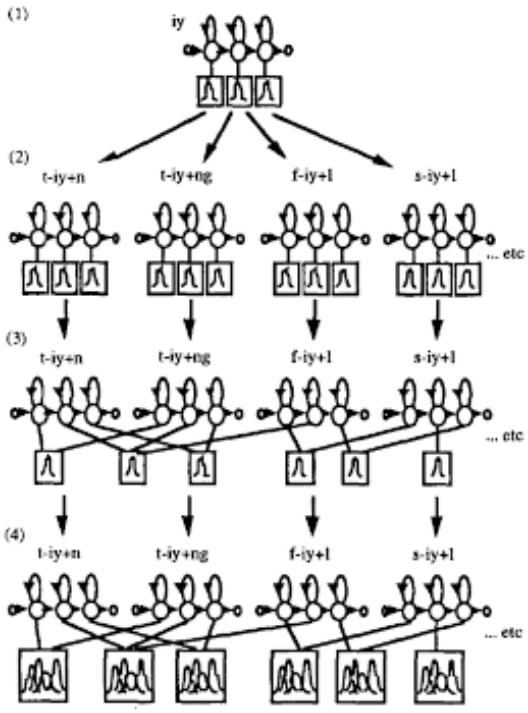

Fig. 2.2: Tomado de [52]. Construcción del HMM de estados vinculados.

Este enfoque es presentado por [18] y utilizado en [52] donde se construye un sistema de estados vinculados como se muestra en la figura 2.2. En el primer paso se crea y entrena un modelo HMM monofono de 3 estados de izquierda a derecha con una función de emisión Gaussiana simple por cada estado. En el segundo paso, por cada contexto, se clona el modelo monofono creando modelos de trifonos dependientes del contexto no vinculados. Estos modelos son entrenados usando la re-estimación de Baum-Welch. En el tercer paso, para cada conjunto de trifonos derivados del mismo monofono, se agrupan los estados de acuerdo a un árbol de decisiones fonéticas. Los árboles de decisiones fonéticas se usan para agrupar los estados de los trifonos cuyo fono central es el mismo en grupos, de manera de maximizar la verosimilitud de los datos de entrenamiento. Los estados agrupados son llamados *senones*. Más detalles sobre estos árboles de decisión se encuentran en [44]. En el cuarto y último paso de la vinculación de estados, el número de componentes de la mezcla de gaussianas correspondiente a cada estado se incrementa y los modelos se vuelven a estimar hasta que el rendimiento en un conjunto de pruebas de desarrollo alcanza su punto máximo o se alcanza el número deseado de componentes de la mezcla.

En este trabajo utilizaremos HMMs en donde los estados ocultos representan *senones* y los

observables están dados por características acústicas con contenido representativo del habla, dadas por las MFCCs. De esta manera los HMM permiten modelar las dependencias temporales de los *senones*, y por consiguiente de los fonos, dentro de una palabra.

Cada estado de los HMMs tiene asociado una distribución de emisión. Tradicionalmente estas distribuciones fueron dadas por modelos de mezcla de Gaussianas (GMM). En los últimos años gracias a los avances tanto en los algoritmos de aprendizaje automático como en el hardware se han podido desarrollar modelos DNNs para reemplazar a los modelos GMMs resultando en mejoras en el rendimiento de los sistemas RAH. A diferencia de los modelos HMM-GMM donde se entrena un modelo GMM por cada *senone*, en los modelos HMM-DNN se entrena un único modelo DNN que predice la probabilidad de todos los *senones*. Es importante notar que los modelos DNNs requieren, para su entrenamiento, alineamientos temporales dados por sistemas RAH entrenados previamente. En general estos alineamientos están dados por sistemas RAH basados en HMM-GMM. En las siguientes secciones explicamos a los modelos GMMs y DNNs.

Modelo de mezcla de Gaussianas (GMM)

Los GMMs son modelos probabilísticos que resultan de hacer combinaciones lineales convexas de más de una distribución Gaussiana. La función de densidad de una mezcla de K gaussianas está dada por:

$$p(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k, \Sigma_k) \tag{2.3}$$

donde π_k es un coeficiente que puede considerarse como la probabilidad a priori de elegir al componente k. Cada densidad Gaussiana $\mathcal{N}(x|\mu_k, \Sigma_k)$ se denomina componente de la mezcla y tiene su propia media μ_k y covarianza Σ_k. La distribución Gaussiana para un vector D-dimensional X se define como:

$$\mathcal{N}(x|\mu, \Sigma) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right) \tag{2.4}$$

donde μ es la media del vector D-dimensional, Σ es una matriz de covarianza de tamaño D x D y $|\Sigma|$ denota el determinante de Σ. En los sistemas HMM-GMM cada estado correspondiente a un senone tiene asociado un GMM que genera las probabilidades de cada observación. En esta tesis, las observaciones están dadas por las características extraídas de la señal, las MFCCs.

Redes neuronales profundas (DNN)

Las redes neuronales poseen "neuronas" representadas por nodos que componen la información de entrada generando un valor de salida. Las neuronas se organizan en capas y están conectadas a través de ejes con pesos asociados. En este trabajo nos concentraremos en la familia de las redes neuronales profundas. Una red neuronal profunda es una red neuronal con múltiples capas ocultas entre su entrada y salida (Figura 2.3).

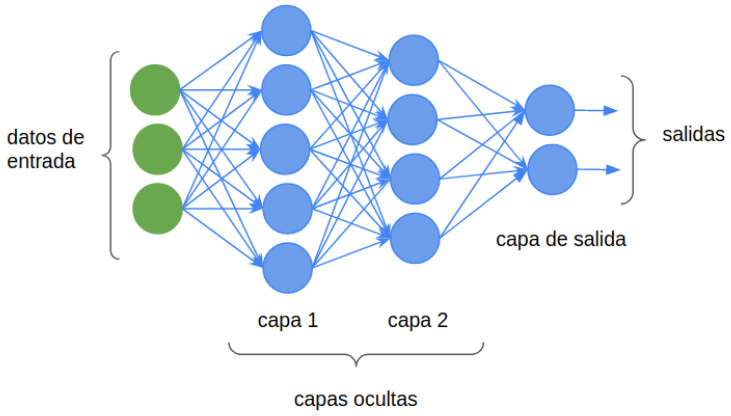

Fig. 2.3: Ejemplo de una red neuronal profunda con dos capas ocultas, la primera con 5 neuronas y la segunda con 4. La capa de entrada corresponde a las características de entrada y la capa de salida corresponde a las predicciones para dos 2 posibles clases. Todos los nodos entre neuronas se conectan a través de ejes, cuando todos los nodos de una capa están conectados a todos los nodos de la capa siguiente, se dice que la capa es densa.

Para cada nodo oculto j, se construye una combinación afín utilizando las salidas de la capa anterior $y_1, ..., y_D$ o la entrada en el caso de ser la primera capa oculta:

$$x_j = b_j + \sum_{i=1}^{D} w_{ij} y_i \tag{2.5}$$

donde $j = 1, ..., M$ siendo M el número de neuronas de la siguiente capa, b_j es el sesgo de la unidad j y w_{ij} es el peso de la conexión entre la unidad i y la unidad j de la capa siguiente (Figura 2.4). Las cantidades x_j se conocen como pre-activaciones y cada una de ellas se transforma usando una función de activación no lineal $h(\cdot)$:

$$y_j = h(x_j) \tag{2.6}$$

En general se utilizan como funciones no lineales $h(\cdot)$ a las funciones sigmoidea, tanh y relu.

La capa de salida es una capa especial donde la función de activación se selecciona de acuerdo a la naturaleza del problema. En este trabajo estamos interesados en la clasificación multiclase de clases mutuamente excluyentes. Para este problema en general se utiliza la función exponencial normalizada, también conocida como función softmax que mapea las salidas a valores entre 0 y 1 que suman 1 como se muestra a continuación:

$$p_j = \frac{\exp(x_j)}{\sum_k \exp(x_k)} \tag{2.7}$$

Los pesos y los sesgos del modelo se encuentran minimizando una función de costo que mide la discrepancia entre las etiquetas de cada muestra y los resultados producidos por la red para

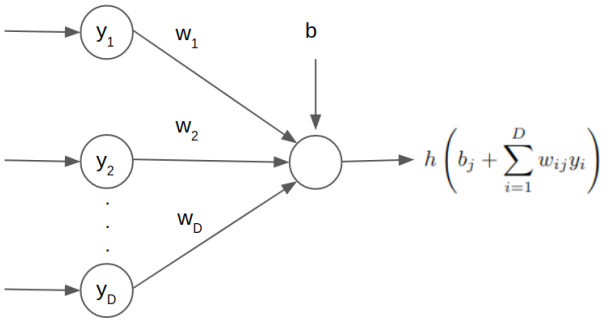

Fig. 2.4: Unidad oculta que recibe como entrada las salida de la capa anterior (o las entradas a la red en el caso de ser la primera capa oculta) $y_1, y_2 ... y_D$, cada entrada y_i se multiplica por el peso asociado al eje w_i y se suma el sesgo de la unidad, b_i. Al resultado obtenido se le aplica una función de activación no lineal diferenciable $h(\cdot)$.

cada instancia de entrenamiento. Cuando se usa como salida de la red la función softmax, una función de costo natural es la entropía cruzada entre los resultados esperados d_{nj} y la salida de la función softmax p_{nj}:

$$C(W) = -\sum_{n=1}^{N}\sum_{j=1}^{J} d_{nj} \log p_{nj} \tag{2.8}$$

donde p_{nj} es la salida de la red para la muestra n y la clase j y d_{nj} es la etiqueta con un esquema de codificación conocido como one hot encoding. En este esquema la variable se representa con un vector J-dimensional, siendo J la cantidad de clases. El elemento que se encuentra en la posición correspondiente a la clase verdadera de la muestra tiene el valor 1 y el resto es igual a 0.

El algoritmo más utilizado para entrenar redes neuronales es el descenso por gradiente. Este algoritmo minimiza la función de costo moviéndose iterativamente en la dirección de mayor decrecimiento dado por el negativo del gradiente de acuerdo a la siguiente ecuación:

$$W^{(\tau+1)} = W^{(\tau)} - \alpha \nabla \mathcal{C}(\mathcal{W}^{(\tau)}) \tag{2.9}$$

donde W es un vector que contiene todos los parámetros de la red y el parámetro $\alpha > 0$ se conoce como tasa de aprendizaje (TA). Después de cada actualización de este tipo, el gradiente se vuelve a evaluar para el nuevo vector de pesos y se repite el proceso. Los gradientes se calculan usando la técnica de retro-propagación. Detalles de esta técnica pueden encontrarse en [2]. En cada paso, el vector de peso se mueve en la dirección de la mayor tasa de disminución de la función de costo.

El descenso por gradiente puede variar en función de cómo se utilizan las instancias de entrenamiento que se utilizan para calcular el costo. Los tres tipos más importantes son:

- estocástico: calcula el costo y actualiza el modelo para cada ejemplo en el conjunto de datos

de entrenamiento. La actualización del modelo con tanta frecuencia puede ser computacionalmente costosa y puede causar gradientes ruidosos.

- lote: calcula el error para cada ejemplo en el conjunto completo de datos de entrenamiento, pero solo actualiza el modelo después de que se hayan evaluado todos las instancias de entrenamiento usando el gradiente promedio . Este enfoque es menos costoso que el método estocástico ya que solo se actualiza cada vez que se realiza una pasada sobre el conjunto de datos de entrenamiento, es decir, una época. Además desarrolla una convergencia y un gradiente de error estable, sin embargo puede converger a estados que no son óptimos debido a que cuando se entrena con todos los datos al mismo tiempo es mas fácil provocar un sobre ajuste que cuando se hace por lotes. El sobre ajuste ocurre cuando un sistema se sobre entrena quedando ajustado a características específicas del conjunto con el que se está entrenando.
- mini lote: se divide el conjunto de datos de entrenamiento en pequeños lotes que se utilizan para calcular el costo y actualizar los coeficientes del modelo. Este método es una solución de compromiso entre los métodos anteriores que, en general, da los mejores resultados.

La tasa de aprendizaje es un hiperparámetro importante debido a que controla el tamaño del paso en cada iteración mientras se mueve hacia un mínimo de la función de costo. Cuanto menor sea el valor, más lento se mueve a lo largo de la pendiente descendiente. Con una tasa de aprendizaje demasiado alta se corre el riesgo de no encontrar los mínimos, sin embargo una tasa de aprendizaje demasiado baja tardará demasiado en converger o se quedará en mínimos locales. Una manera de evitar este problema es disminuir exponencialmente la tasa de aprendizaje a lo largo del tiempo, es decir, se comienza a entrenar con un valor alto de tasa de aprendizaje y se va disminuyendo exponencialmente a medida que avanza el entrenamiento.

En esta tesis usamos entrenamiento por mini lotes para dividir el conjunto de datos de entrenamiento. Además usamos decaimiento exponencial para la tasa de aprendizaje. El decaimiento exponencial se realiza seleccionando una tasa t de decaimiento y una cantidad de mini lotes p. Cada p mini lotes la tasa de aprendizaje actual decaerá el valor que indique t.

Como ya hemos mencionado, los sistemas RAH basados en HMM-DNN utilizan una única DNN entrenada para la tarea de reconocimiento de senones. En estos modelos las probabilidades de emisión de los estados del HMM, que representan senones, son calculadas a partir de las posterior dadas por la DNN de la siguiente manera:

$$p(Y|s) = \frac{P(s|Y)P(Y)}{P(s)} \tag{2.10}$$

donde Y es la secuencia de características completa para la señal de audio, $P(Y)$ es desconocido y es aproximado por una constante, $P(s|Y)$ esta dada por el modelo DNN y $P(s)$ es la probabilidad a priori del senone s.

2.1.3. Modelo de lenguaje

Los modelos de lenguaje estiman la probabilidad de observar una palabra dentro de una oración. Estos modelos en general son representados por modelos de N-Gramas donde la probabilidad de una palabra w_k depende de las $n-1$ palabras anteriores $W_{k-n+1}^{k-1} = w_{k-n+1}, w_{k-n+2}, ..., w_{k-1}$ y se define como:

$$P(w_k|W_1^{k-1}) = P(w_k|W_{k-n+1}^{k-1}) \tag{2.11}$$

Los distribuciones de probabilidad de los N-gramas se pueden calcular a partir de datos de texto, estimándose a partir del recuento de frecuencias. Por ejemplo para un trigrama (N=3):

$$P(w_k|w_{k-1}, w_{k-2}) = \frac{t(w_{k-2}, wk-1, w_k)}{b(w_{k-2}, w_{k-1})} \tag{2.12}$$

donde $t(a, b, c)$ es el numero de veces que aparece el trigrama a, b, c en los datos de entrenamiento y $b(a, b)$ es el numero de veces que aparece el bigrama a, b. En los sistemas RAH, estos modelos son utilizados para determinar las probabilidades de transición entre las diferentes palabras.

2.1.4. Alineador forzado

En la alineación forzada, el habla y su transcripción ortográfica se alinean automáticamente a nivel de palabra, fono y *senone*, lo que proporciona una forma de determinar en que instante de tiempo fue pronunciada cada unidad. La alineación forzada puede realizarse a partir de un sistema RAH, donde el modelo de lenguaje se restringe a las secuencias de palabras dadas por la transcripción. En este trabajo usaremos un modelo RAH basado en HMM-GMM para obtener los alineamientos para cada *senone* a partir del habla de los estudiantes asumiendo que el texto es conocido.

2.2. Sistemas de puntuación de la pronunciación

En esta sección describimos dos sistemas de calificación de la pronunciación. El primero basado en SVM y el segundo en DNN. El sistema basado en SVM es usado para comparar con el sistema basado en DNN desarrollado en esta tesis.

2.2.1. Supervectores modelados con SVMs

Este sistema consiste en una Maquina de Vectores Soporte (SVM) entrenada con atributos llamados supervectores. Fue propuesto por Franco en [9] y al momento de su presentación tenía rendimientos de estado del arte.

En la figura 2.5 se presenta la arquitectura del sistema. Primero se extraen MFCCs a partir de la señal de habla y se realizan los alineamientos forzados entre dicha señal y su transcripción. A continuación se calculan supervectores utilizando un modelo llamado Gaussian Mixture Model

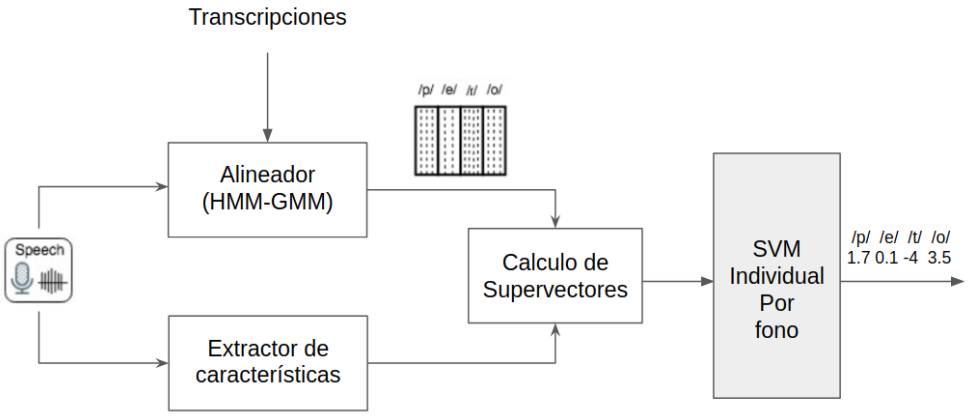

Fig. 2.5: Sistema de puntuación de la pronunciación: Supervectores modelados con SVMs. En este sistema, a partir del calculo de alineamientos forzados y extracción de características (MFCCs), se calcula, por cada instancia de fono un vector de características llamado supervector. Los supervectores son usados para entrenar un modelo SVM por fono. La salida del modelo es utilizada como la puntuación de la muestra.

– Universal Background Model (GMM-UBM). Un UBM es un GMM entrenado para representar la distribución de las características acústicas independientemente de la clase y del hablante.

Por cada fono se entrena un GMM-UBM con instancias bien y mal pronunciadas. Luego, por cada instancia de fono, se obtiene un nuevo GMM adaptando los parámetros del GMM-UBM al vector de instancias por medio de la estimación llamada Máximo a Posteriori (MAP) [40]. Las medias y los pesos de las componentes Gaussianas de cada GMM adaptado son apiladas para obtener un supervector como se ve en la Figura 2.6. Estos supervectores son utilizados como entrada para un SVM. En el sistema utilizado en este trabajo, para adaptar el GMM-UBM solo usamos los pesos y las medias debido al tamaño limitado del conjunto de datos.

Una vez calculados los supervectores, por cada fono f_k se entrena un modelo SVM. Cada uno de estos modelos es entrenado considerando un conjunto de m muestras $S = (x_i, z_i)$ con $i = 1, ..., m$ donde x_i es el supervector y z_i es la clase para la instancia i del fono f_k correspondiente en los datos de entrenamiento. Se utiliza 1 para una pronunciación correcta y -1 para una incorrecta. El objetivo del SVM es encontrar una función $f(x) = w^T x + b$, tal que el signo de $f(x)$ sirva para estimar la clase predicha para el vector de características x.

Dado el conjunto de datos S los parámetros w y b se obtienen resolviendo el problema de optimización:

$$\begin{aligned} &\underset{w,\,\epsilon}{\text{minimize}} \quad \left\{ \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n} \epsilon_i \right\} \\ &\text{subject to} \quad z_i * (w^T x_i + b) \geq 1 - \epsilon_i \quad \text{and} \quad \epsilon_i \geq 0 \quad \text{for all} \quad i \end{aligned} \tag{2.13}$$

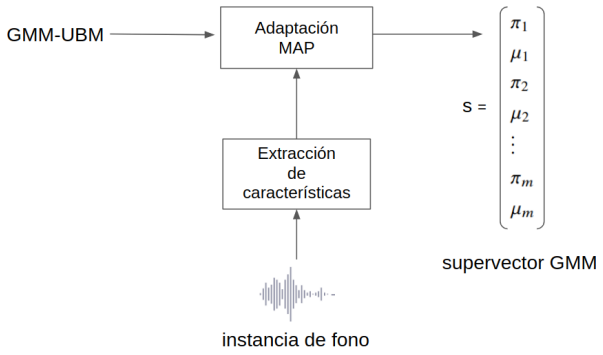

Fig. 2.6: Figura adaptada de Campbell [3]. Dada una instancia de fono, se obtiene un modelo GMM adaptando los parámetros del GMM-UBM al vector de instancia con una estimación de Máximo a Posteriori (MAP). Las medias μ_i y los pesos π_i del modelo GMM son apiladas para obtener un supervector.

Las variables ϵ_i miden el error cometido en cada muestra. La variable C es un parámetro de entrada que determina un relación de compromiso entre los errores y el tamaño del margen w.

Dada una instancia de fono de testeo o evaluación, primero se calcula su supervector x y $f(x)$ (la distancia de ese supervector al hiperplano SVM) se toma como la calificación de la muestra.

2.2.2. Goodness of Pronunciation (GOP) basado en DNNs

Este sistema se basa en el trabajo de Hu [16]. En la figura 2.7 podemos ver un esquema del sistema: el modelo acústico recibe como entradas características extraídas de las señales de habla y los alineamientos forzados entre las señales de audio y su transcripción ortográfica. El modelo acústico devuelve las probabilidades a posteriori de cada *senone* para cada frame, las cuales son luego usadas por la medida *Goodness of Pronunciation* (GOP) adaptada para modelos DNN.

La medida *Goodness of Pronunciation* (GOP) fue desarrollada por Witt en [50] con el fin de evaluar la calidad de la pronunciación a partir de un modelo entrenado con habla nativa. Esta medida calcula la similitud entre el habla de los estudiantes y el modelo entrenado. El sistema original está dado por un modelo RAH basado en HMM-GMM que estima las probabilidades a posteriori de los sonidos que el estudiante debería haber pronunciado dada la secuencia de palabras de la frase leída. Se asume que estas probabilidades serán bajas cuando la pronunciación sea incorrecta, ya que las características de la señal no coincidirán con lo que el modelo entrenado con hablantes nativos espera encontrar.

La medida GOP se define como la duración normalizada del logaritmo de la probabilidad a posteriori de que un hablante haya pronunciado el fono q_i dado el segmento acústico correspon-

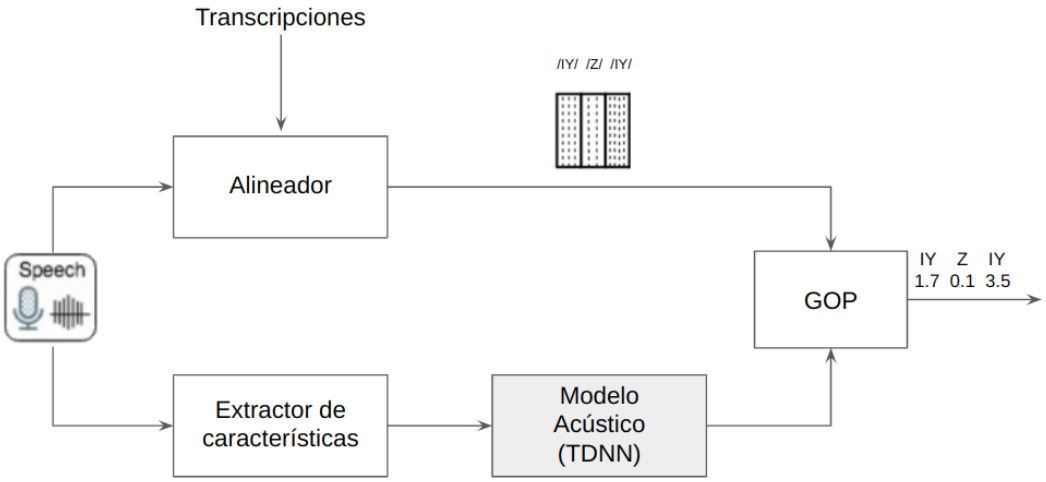

Fig. 2.7: Sistema de puntuación de la pronunciación: *Goodnes of Pronunciation* (GOP) basado en DNNs. En este sistema, a partir del calculo de alineamientos forzados y extracción de características, se utiliza un modelo acústico basado en TDNNs para calcular la probabilidad de todos los *senones* posibles para cada frame. Luego se calcula el puntaje GOP para cada fono a partir de las probabilidades de los *senones* que lo componen.

diente $Y_i = [y_{t_s}..., y_{t_e}]$, es decir:

$$GOP(q_i) = |\log P(q_i|Y_i)|/(t_e - t_s + 1) \tag{2.14}$$

siendo t_s y t_e el índice del frame donde inicia y termina Y_i respectivamente y $t_e - t_s + 1$ el numero de frames en el segmento acústico Y_i.

Debido a que el modelo HMM-GMM devuelve la verosimilitud, y no la probabilidad a posteriori requerida en 2.14, se calcula la probabilidad a posteriori a partir del Teorema de Bayes:

$$GOP(q_i) = |\log P(q_i|Y_i)|/(t_e - t_s + 1) = \left|\log\left(\frac{p(Y_i|q_i)P(q_i)}{\sum_{j=1}^{J} p(Y_i|q_j)P(q_j)}\right)\right| \Big/ (t_e - t_s + 1) \tag{2.15}$$

siendo J el conjunto de todos los fonos modelados y $P(Y_i|q_i)$ se estima como $\prod_{t=t_s}^{t_e} p(y_t|q_i)$. Para calcular GOP, se asume que todos los fonos son igualmente probables ($\mathrm{P}(q_i) = \mathrm{P}(q_j)$) para todo q_i y q_j.

En los últimos años el puntaje GOP basado en HMM-GMM fue extendido en diferentes trabajos [13, 16, 15] para ser utilizado con modelos HMM-DNN. La salida del modelo acústico basado en DNN genera probabilidades a posteriori de cada *senone* s_t dada la secuencia de observaciones Y en el tiempo t. A partir de la secuencia de senones (determinada por el alineador forzado) que pertenecen a un fono, una forma directa de evaluar la calidad de la pronunciación

a partir de un modelo basado en DNN es calcular GOP como:

$$GOP(q_i) = \log P(q_i|Y; t_s, t_e) \approx \frac{1}{t_e - t_s + 1} \sum_{t=t_s}^{t_e} \log P_t(s_t|Y) \tag{2.16}$$

donde Y es la secuencia de características completa para la señal de audio y $P_t(s_t|Y)$ es la estimación de la probabilidad a posteriori para el senone s_t en el frame t.

Es posible que el modelo DNN no tenga buenos resultados para algunos *senones* generando de esta manera una medida pobre para el conjunto de fonos que los utilizan. Es por esto que Hu en [16] propone una medida alternativa que toma en cuenta al conjunto de *senones* correspondientes a cada fono:

$$GOP(q_i) = \log P(q_i|Y; t_s, t_e) \approx \frac{1}{t_e - t_s + 1} \sum_{t=t_s}^{t_e} \log P_t(q_i|Y) \tag{2.17}$$

donde:

$$P_t(q_i|Y) = \sum_{s \in Q_i} P_t(s|Y) \tag{2.18}$$

donde Y es la secuencia de características completa para la señal de audio y $P_t(q_i|Y)$ es la estimación de la probabilidad a posteriori para el fono q_i en el frame t. Q_i es el conjunto de todos los *senones* correspondientes al fono q_i. Es importante tener en cuenta que cada fono está formado por tres senones que pertenecen a un conjunto de senones posibles para ese fono. A su vez cada senone pertenece a un único fono.

Time Delay Neural Network (TDNN)

Como ya hemos mencionado, en las DNNs por cada unidad básica se calcula la suma ponderada de las entradas (ecuación 2.5) y luego se utiliza una función no lineal (ecuación 2.6).

En las TDNN por cada unidad básica se introducen contextos $D1, ..., DN$ que serán recibidos por las neuronas de la capa siguiente (Figura 2.8). De esta manera, una unidad TDNN tiene la capacidad de relacionar y comparar la entrada actual con observaciones dentro de su contexto cercano dado por los retardos de tiempo. En cada capa de la red, es posible elegir qué vectores presentes en el contexto del frame central se combinarán para generar la entrada a la próxima capa.

Las TDNN son muy usadas en los sistemas RAH, fueron el estándar durante muchos años y por eso las hemos usado en este trabajo.

2.3. Métricas

A continuación se describen diferentes métricas que permiten evaluar los resultados de los métodos de puntuación de pronunciación. Estas métricas son calculadas a partir de los puntajes obtenidos para una base de datos de hablantes no nativos. En este trabajo, diremos que una instancia es positiva cuando fue bien pronunciada y negativa en caso contrario.

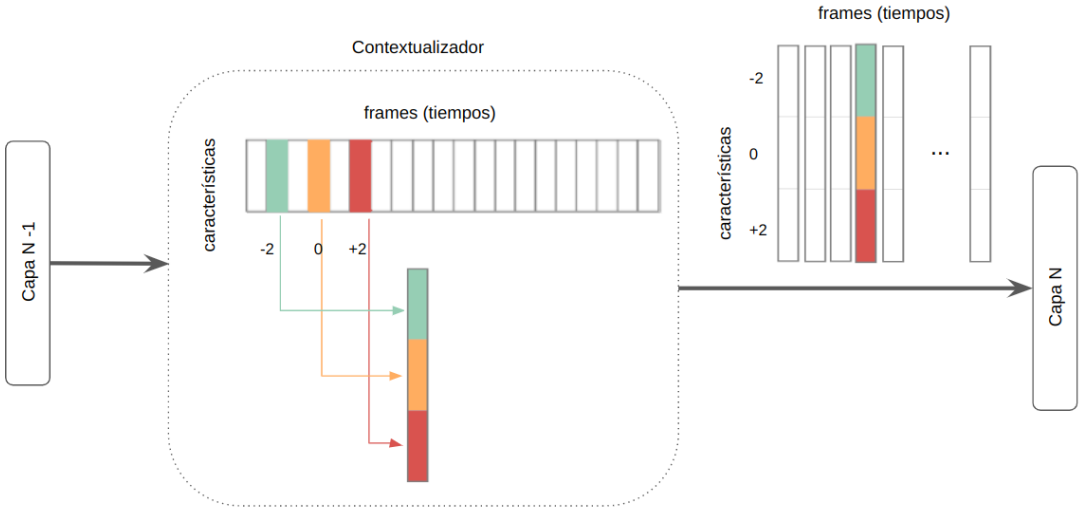

Fig. 2.8: Ejemplo TDNN con contexto de -2, 0, 2. El contextualizador por cada *frame* concatena el frame central con los dos frames que se encuentran a distancia dos a izquierda y derecha.

2.3.1. Histograma

Usaremos histogramas para representar la distribución de las clases de los diferentes fonos. Las instancias de cada fono cuentan con una calificación de pronunciación dada por el sistema y son divididas en las clases de fonos “bien” y “mal” pronunciados de acuerdo a las anotaciones realizadas por un lingüista experto.

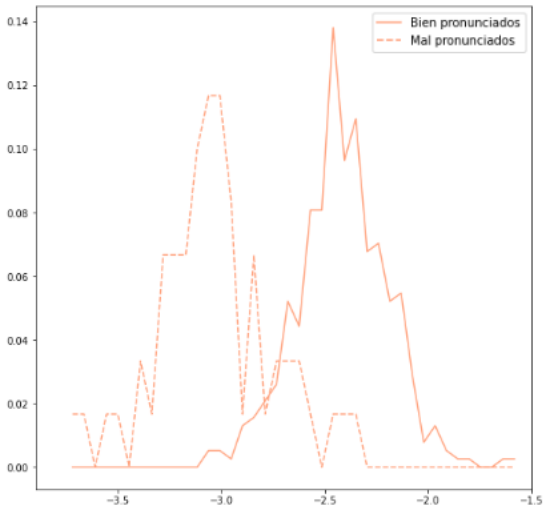

Fig. 2.9: Ejemplo de histograma para dos clases de fonos, bien y mal pronunciados

Los histogramas son gráficos bidimensionales en cuyo eje horizontal se distribuyen los datos divididos en rangos de valores numéricos, llamados bins, y en cuyo eje vertical se representa la frecuencia de los datos. Se puede ver un ejemplo de histograma en la figura 2.9

2.3.2. Curva ROC

La curva ROC (*receiver operating characteristic*) ilustra para diferentes valores de un umbral de decisión (valor del puntaje a partir del cual un caso se etiqueta como positivo) el *False Positive Rate* (FPR) frente al *True Positive Rate* (TPR) definidos como:

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}}$$
$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

donde:

False Positive (FP): cantidad de instancias clasificadas por el sistema erroneamente como positivas.

False Negative (FN): cantidad de instancias clasificadas por el sistema erróneamente como negativas.

True Negative (TN): cantidad de instancias clasificadas por el sistema correctamente como negativas.

True Positive (TP): cantidad de instancias clasificadas por el sistema correctamente como positivas.

En la figura 2.10 podemos ver un histograma de las distribuciones de las clases de fonos bien y mal pronunciados para un fono a partir de las puntuaciones dadas por el sistema. La barra vertical azul indica el umbral de decisión. La curva ROC (Figura 2.11) es construida variando el umbral a lo largo de todo el rango de valores de la entrada calculando los FPR y TPR correspondientes. Una vez obtenida la curva ROC se selecciona un umbral de decisión adecuado de acuerdo a la tarea que se quiera resolver. En muchos trabajos se utiliza como umbral de decisión al umbral correspondiente al *Equal Error Rate* (EER). Siendo el EER definido como el punto donde el *False Negative Rate* coincide con el *False Positive Rate*.

Además de la curva ROC nos interesa calcular la curva de *False Postive Rate* frente a *False Negativa Rate*. En la figura 2.11 se puede ver un ejemplo de ambas curvas.

Área bajo la curva ROC (AUC)

Una medida relacionada a la curva ROC es AUC (*Area Under The Curve* en inglés) que, como indica su nombre, corresponde el área debajo de la curva ROC. El AUC es una medida

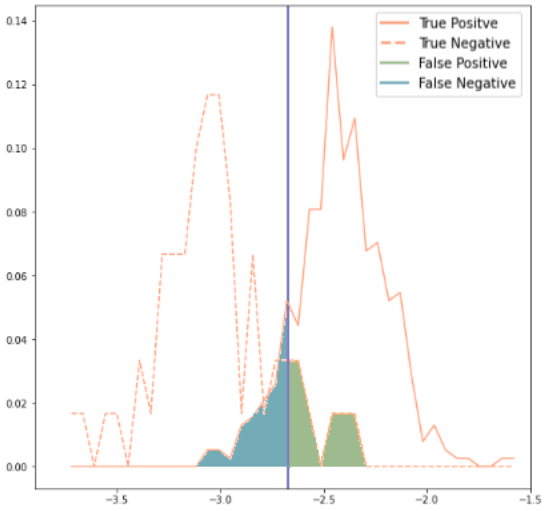

Fig. 2.10: Ejemplo de histograma para las clases de fonos bien y mal pronunciados con umbral de decisión (barra azul) separando TP, TN, FP, FN.

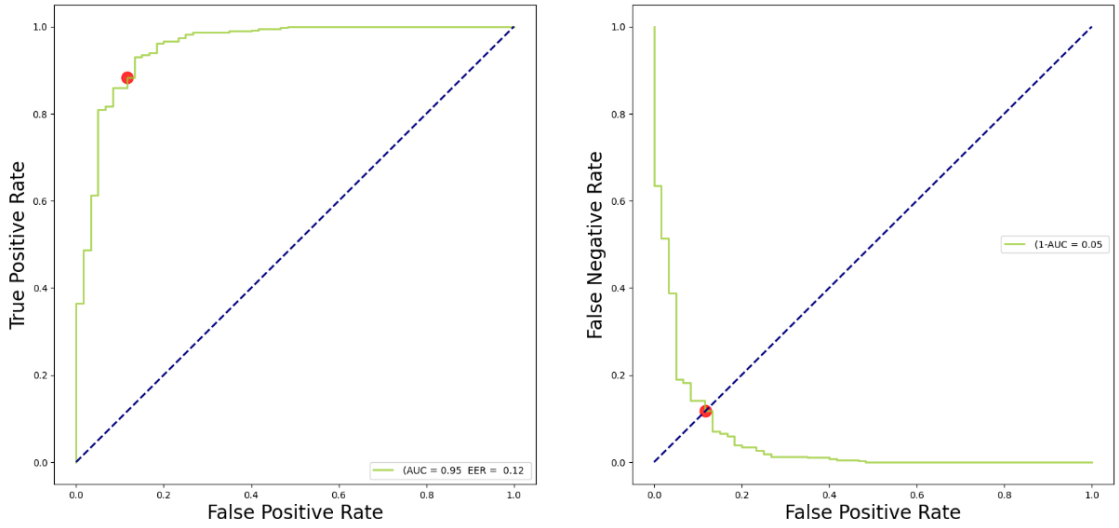

Fig. 2.11: A la izquierda una Curva ROC que ilustra el *False Positive Rate* frente al *True Positive Rate*, a la derecha una Curva de *False Negative Rate* frente a *False Positive Rate*. Los puntos rojos indican el punto correspondiente al EER.

estándar utilizada en aprendizaje automático. Un AUC cercano a 1 indica un sistema que separa las clases casi perfecto. Para la curva de FPR frente a FNR se utiliza la medida 1-AUC que indica el área debajo de la curva. En este caso, valores más chicos son mejores, como es el caso para el EER y el MinCost que lo definimos a continuación. Para simplificar el análisis visual, en

el siguiente capitulo, en lugar de reportar AUC reportamos 1-AUC junto con EER y MinCost.

2.3.3. Función de costo

Además de la medida AUC utilizamos una medida que consideramos más apropiada para la tarea de calificación de la pronunciación a la que llamamos costo. En esta métrica los FP son menos penalizados que los FN con el fin de evitar frustrar a los estudiantes indicando que un fono fue mal pronunciado cuando, en realidad, fue pronunciado correctamente. Definimos al costo como:

$$\text{Cost} = 0.5\ \text{FPR} + \text{FNR} \tag{2.19}$$

Para calcular el costo es necesario seleccionar un umbral de decisión. En este trabajo elegimos el umbral que minimiza el costo de cada fono en el conjunto de datos donde se está evaluando y lo llamamos MinCost. Este costo es optimista debido a que el umbral de decisión es seleccionado en el mismo conjunto donde se evalua. Por esta razón, también calculamos el costo que se obtiene cuando el umbral es elegido en un conjunto de datos diferente al que se está evaluando y lo llamamos ActCost.

3. DISEÑO EXPERIMENTAL

3.1. Diseño experimental

En esta sección describimos las características de los sistemas utilizados y las bases de datos seleccionadas para su desarrollo y evaluación.

3.1.1. Bases de datos

Utilizamos las bases de datos Librispeech [37] y EpaDB [47] para entrenar y evaluar los sistemas basados en DNN y SVM. A continuación describimos ambas bases de datos y el uso que se les ha dado en cada uno de los sistemas.

Librispeech

Librispeech es una base de datos de habla leída en inglés tomados del proyecto LibriVox de grabaciones de lectura de libros de dominio público. Cuenta con una duración aproximada de 1000 horas de habla en 16KHz. Para este trabajo se ha seleccionado un subconjunto de habla limpia de 316 hs de 842 hablantes.

EpaDB

EpaDB fue creada especialmente para la tarea de calificación de la pronunciación. Consiste en 3200 audios de argentinos mayoritariamente de la región del Río de la Plata hablando en inglés. Cada participante, 25 hombres y 25 mujeres, grabó 64 frases cortas especialmente diseñadas para contener todos los sonidos del inglés y, en especial, aquellos difíciles de pronunciar para un hablante de español de Argentina. Los datos recolectados fueron transcritos y etiquetados con alineamientos temporales a nivel alófono (variantes dependientes del contexto) y a nivel frase por una lingüista y una profesora de inglés con conocimientos de fonética. Por cada frase se utilizó el alineador forzado Montreal Forced Aligner para determinar que fonos fueron pronunciados en cada instante de tiempo. Más detalles sobre el alineamiento forzado se pueden encontrar en la sección 3.1.2. Luego se corrigieron manualmente los limites de los sonidos y las etiquetas dadas por el alineador. Los limites y las etiquetas deben ser corregidos debido a que si bien las personas no nativas hablan inglés y el alineador transcribe inglés, los fonos no nativos, por provenir de un estudiante de idiomas, pueden ser mas largos o mas cortos o introducir errores de supresión, sustitución o adición. Los errores de supresión suceden cuando un fono es eliminado. Los de

sustitución, cuando un sonido del inglés es cambiado por otro sonido ya sea del ingles o del español. Por último, las adiciones ocurren cuando se agregan sonidos.

Definición de conjuntos de entrenamiento, desarrollo y evaluación

El objetivo de los modelos de aprendizaje automático es encontrar reglas que rigen la relación entre la entrada y la salida del modelo. La idea subyacente es que las reglas encontradas en el entrenamiento generalizarán a otros conjuntos de datos permitiendo que el modelo realice predicciones. Esto motiva la división de los datos en tres subconjuntos: entrenamiento, desarrollo y evaluación. El conjunto de entrenamiento se utiliza para entrenar al modelo y ajustar sus parámetro (pesos y sesgos), mientras que el conjunto de desarrollo, también conocido como conjunto de validación, es usado para ajustar los hiperparámetros. Más detalles sobre los hiperparámetros ajustados en 3.1.2 y 4.1. El tercer y último conjunto, llamado conjunto de evaluación, se utiliza para evaluar el rendimiento del modelo final, es decir, si el modelo generaliza a conjuntos de datos que tienen la misma distribución que los datos utilizados para entrenamiento y desarrollo pero que nunca antes se habían visto. Tanto el conjunto de desarrollo como el de evaluación permiten detectar el sobre ajuste. Este fenómeno ocurre cuando un sistema se sobre entrena quedando ajustado a características específicas del conjunto con el que se está entrenando.

El conjunto de entrenamiento de Librispeech cuenta con 759 hablantes y es utilizado para el entrenamiento de la DNN, mientras que el conjunto de validación cuenta con 83 hablantes y se utiliza para ajustar sus hiperparámetros. EpaDB se divide en dos subconjuntos, 30 hablantes en el conjunto de desarrollo y 20 en el conjunto de evaluación. En ambos sistemas, el conjunto de evaluación de EpaDB es utilizado para determinar el rendimiento del sistema final para la tarea de calificación de la pronunciación. El conjunto de datos de desarrollo de EpaDB es usado ligeramente distinto para cada uno de los modelos. En el sistema basado en DNN el conjunto de desarrollo se utiliza para ajustar los hiperparámetros de la red y para determinar un umbral de decisión que permita separar los fonos bien pronunciados de los mal pronunciados, mientras que en el sistema supervector-SVM además de ser utilizado para determinar un umbral, el conjunto de desarrollo es usado para entrenar al modelo con validación cruzada de k-pliegues. La validación cruzada de k-pliegues es un método estadístico que se utiliza para estimar la eficacia de los modelos de aprendizaje automático cuando se posee un conjunto acotado de datos como en el caso de EpaDB. En este método las instancias se dividen aleatoriamente en k submuestras de igual tamaño. Cada submuestra se separa como conjunto de validación y se entrena con el resto de los datos. El resultado de todas las muestras de validación es combinado para obtener el resultado final. En nuestro caso, este método se utiliza para obtener puntuaciones que permitan determinar el umbral en el sistema supervector-SVM.

3.1.2. Sistema GOP DNN

El sistema GOP-DNN que desarrollamos en este trabajo replica el sistema propuesto por [16]. Este sistema está formado por un modelo acústico DNN entrenado con los datos nativos tomados de la base de datos LibriSpeech. Recibe como entradas MFCCs y alineamientos forzados.

La salida del modelo es utilizada para calcular los puntajes a la calidad de la pronunciación dados por el algoritmo de GOP. A cada fono calificado se le asigna una etiqueta que indica si fue bien o mal pronunciado. Estas etiquetas junto con las calificaciones de pronunciación permiten la evaluación del rendimiento del sistema y la selección de un umbral de discriminación que será usado para tomar las decisiones.

A continuación describimos el tipo de MFCCs y alineamientos forzados utilizados para el entrenamiento del modelo DNN. Luego se describe el calculo de las etiquetas y por último las propiedades de los modelos que han sido entrenados.

MFCCs

Las características seleccionadas para el entrenamiento del modelo consisten en 13 MFCCs con sus correspondientes deltas y doble deltas. El tamaño de frame elegido tiene una duración de 25 ms. con un corrimiento de 10 ms. Entrenamos modelos con cuatro combinaciones distintas: (1) normalizar cada dimensión del vector de características por su media y desviación estándar dentro de cada muestra, (2) normalizar cada dimensión solo por su media, (3) normalizar cada dimensión solo por su desviación estándar y (4) no normalizarlas.

Alineamiento forzado

Los alineamientos forzados son calculados por la herramienta Montreal Forced Aligner [29]. MFA está construido sobre Kaldi, un conjunto de herramientas de reconocimiento automático de voz de código abierto [38]. El RAH implementado por MFA usa una arquitectura estándar de HMM-GMM.

Los alineamientos determinan el senone que pronunció el hablante en cada instante de tiempo. A su vez, cada senone pertenece a un único fono. Esto permite recuperar qué fono fue pronunciado en cada frame. Este alineador es entrenado con datos de hablantes nativos del idioma objetivo. Se asume que tal alineador puede generar alineamientos razonablemente buenos para hablantes nativos. Sin embargo, es claro que estos modelos pueden no ser óptimos para la población de hablantes no nativos dado que durante el entrenamiento nunca vieron pronunciaciones de este tipo. Dentro del grupo de investigación de la Dra. Ferrer ya se encuentran personas trabajando para mitigar este problema.

Obtención y alineamiento de las etiquetas

En esta sección revisamos como realizar el cálculo de las etiquetas que indican si un fono manual fue bien o mal pronunciado. Luego como realizar el alineamiento de cada una de estas etiquetas con los fonos automáticos y el puntaje GOP correspondiente. En la figura 3.1 podemos ver un ejemplo de la tarea para una instancia de la frase **"I knew a very tall man"**. Esta tarea se encuentra dividida en dos pasos:

Paso 1: Obtención de las etiquetas

En este paso se asigna una etiqueta a cada una de las anotaciones manuales donde se indica qué fono fue pronunciado. Para esta tarea se utiliza un conjunto de posibles transcripciones de referencia correspondientes a las posibles formas en que un hablante nativo podría pronunciar cada una de las frases que se encuentran en EpaDB. Las anotaciones manuales contemplan los errores de sustitución, adición y supresión marcando a las supresiones con un cero y las adiciones como la suma de los fonos pronunciados.

Para determinar las etiquetas de los fonos manuales, primero se obtiene la mejor transcripción de referencia para cada anotación manual, es decir, la que mejor se alinea con la transcripción manual. Luego se comparan posicionalmente los fonos manuales con los fonos de la mejor transcripción de referencia. Si los fonos coinciden, la etiqueta indicará que fue bien pronunciado (+), de lo contrario que fue mal pronunciado (-).

Paso 2: alineamiento de las etiquetas

En este paso se busca alinear las etiquetas obtenidas en el paso 1 con las calificaciones GOP dadas por el algoritmo. Para esta tarea es necesario tomar en cuenta que los fonos utilizados en las anotaciones manuales contemplan al habla no nativa, esto genera una discrepancia entre los fonos dados por el alineamiento forzado y los dados por las anotaciones manuales. Por otro lado la manera en que se anotan los fonos en el alineamiento forzado no coincide exactamente con la manera en que se anotan manualmente. En el caso de los alineamientos automáticos las supresiones no serán marcadas y las adiciones se encontrarán como fonos distintos, a diferencia de las anotaciones manuales donde las supresiones se marcan con cero y las adiciones como sumas de fonos. Para subsanar estas discrepancias se utilizan nuevamente las transcripciones de referencia. Primero se busca la mejor transcripción de referencia para la anotación automática dada por el alineador. En este caso se buscar la mejor transcripción de referencia omitiendo las supresiones debido a que la transcripción automática no contempla este tipo de errores. Dada la mejor transcripción de referencia sin supresiones se toma como transcripción de referencia a la transcripción seleccionada agregando las supresiones correspondientes a la frase. La mejor transcripción con supresiones para la anotación automática (y su correspondiente GOP) tendrá la

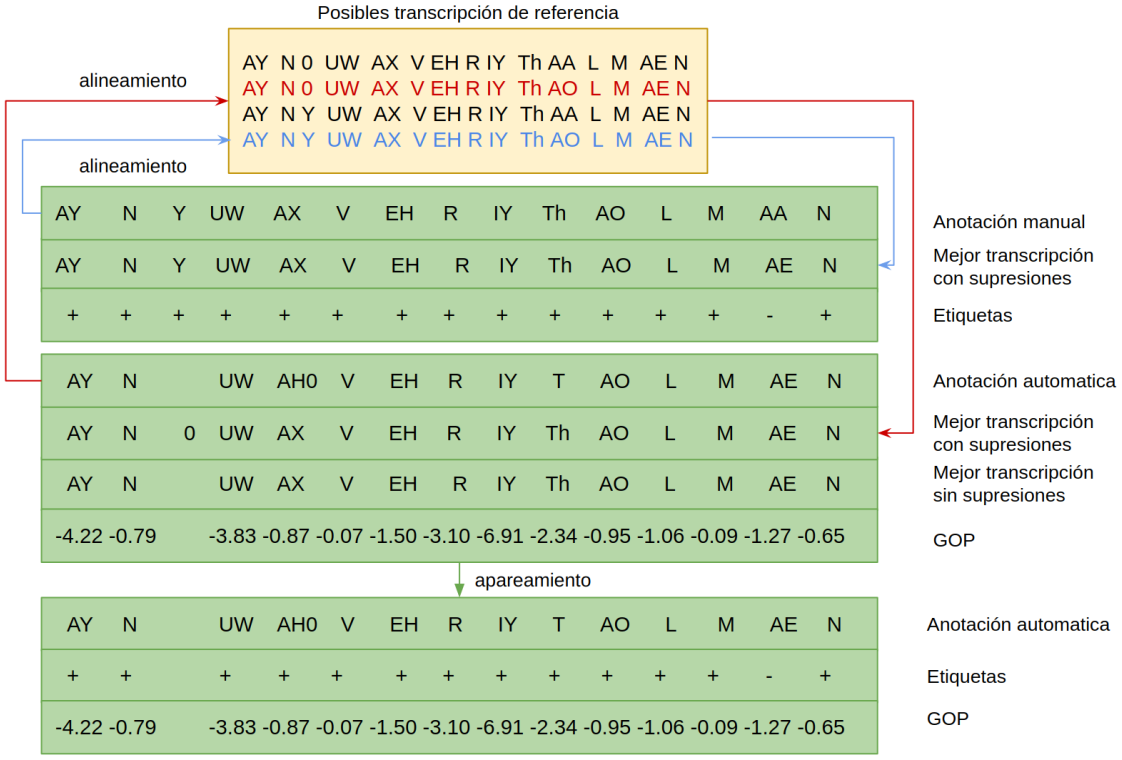

Fig. 3.1: Asignación de etiquetas para cada fono puntuado por el algoritmo para la frase **I knew a very tall man**. Paso 1) se busca la mejor transcripción de referencia para la anotación manual y se calculan las etiquetas comparando cada par de fonos, si coinciden se marca como bien pronunciada, de lo contrario como mal pronunciada. Paso 2) se busca la mejor transcripción de referencia para la anotación automática, luego se aparean la mejor transcripción de referencia para la anotación manual con su etiqueta y la mejor transcripción de referencia con supresiones para la anotación automática con su correspondiente GoP eliminando las supresiones.

misma longitud que la anotación manual (y sus correspondientes etiquetas) posibilitando de esta manera el apareamiento entre ellas. En este apareamiento se descartarán todas las supresiones, obteniendo de ese modo una etiqueta para cada fono automático.

Modelo TDNN

Construimos diferentes modelos para seleccionar las características de entrada y los hiperparámetros que generan los mejores resultados en la tarea de calificación de pronunciación. En esta sección describimos los hiperparámetros generales usados en los experimentos y en la sección 4.1 presentamos el rendimiento al variar algunos de estos hiperparámetros.

En todos los sistemas entrenados por cada señal de habla se calculan las MFCCs y se organizan en mini lotes de entrenamiento. Tanto para EpaDB como para Librispeech las muestras son de tamaño variable. Con el fin de asegurar que todos las muestras de EpaDB sean calificadas se ha utilizado como tamaño máximo de muestra al tamaño de la muestra de mayor longitud de EpaDB (500 frames). Sin embargo, muchas de las señales de Librispeech superan ese tamaño, para garantizar un mejor aprovechamiento de dichas señales se han cortado en cada silencio. Aún así algunas de las señales resultantes siguen superando al tamaño máximo, pero esto no supone un problema dado que simplemente se descartarán los frames que lo superen. Además de las frases que superan el tamaño máximo, existen frases que no llegan a cubrirlo. En estas frases se completa el espacio sobrante con ceros y luego al momento de calcular el costo no son tomados en cuenta.

Para actualizar iterativamente los pesos de la red, en lugar de utilizar el algoritmo del descenso del gradiente estocástico tradicional se utiliza el algoritmo Adam [21]. Adam mantiene una tasa de aprendizaje para cada peso de la red que se adaptan por separado a lo largo del aprendizaje. Estas tasas son calculadas a partir de estimaciones del primer y segundo momento de los gradientes. Además de la tasa de aprendizaje α utilizada en el descenso del gradiente, este algoritmo utiliza tres parámetros β_1, β_2 y ϵ. β_1 y β_2 representan tasas de caída exponencial para las estimaciones del primer y el segundo momento respectivamente. Mientras que ϵ es un número pequeño que permite evitar divisiones por cero. Los valores para estos parámetros han sido seleccionados de acuerdo a la sugerencia dada por sus creadores en [21] (β_1=0.9, β_2=0.999 y ϵ=10^{-8}).

También usamos batch normalization y dropout. Batch normalization, introducido en [19], es un método utilizado para hacer que el entrenamientos de las redes neuronales sean más rápidas y más estables mediante la normalización de cada mini-lote de entrenamiento. Dropout es una técnica introducida en [43]. La idea clave de esta técnica es descartar aleatoriamente unidades de la red, junto con sus conexiones, durante el entrenamiento, evitando de esta manera el sobre ajuste que puede ocurrir en algunas redes neuronales. La utilización de dropout requiere el uso de una tasa de dropout que indica el porcentaje de neuronas que serán descartadas, en esta tesis usaremos un porcentaje que varía entre 10-50 %.

Además en la sección de experimentos exploramos diferentes tipos y cantidades de capas, tasas de aprendizaje, cantidad de épocas de entrenamiento y tipos de normalización para las características de entrada.

3.1.3. Sistema supervector-SVM

El sistema supervector-SVM es una replica del sistema presentado en [9] implementado en la tesis de Licenciatura dirigida por la Dra. Ferrer [28]. Este sistema a diferencia del sistema DNN, es entrenado con datos no nativos, en este caso se usan los datos de entrenamiento de EpaDB con validación cruzada de k-pliegues como se describió en 3.1.1. El sistema recibe como entradas las

MFCCs descriptas en la sección 3.1.2 y alineamientos forzados con su correspondiente etiqueta. Las etiquetas para los alineamientos forzados son las mismas que utiliza el sistema DNN descripto en 3.1.2.

Como ya hemos visto en la sección 2.2.1 en el sistema supervector-SVM primero se obtienen los GMM por cada instancia de fono a través de la adaptación de un UBM-GMM. Luego se entrena un modelo SVM usando como entrada los supervectores obtenidos al concatenar las medias y pesos de los GMMs adaptados. Para el GMM de cada fono, el número de mezclas de gaussianas entrenado es proporcional al número de instancias disponibles de cada fono. Cada 15 instancias de un fono se entrena una componente de la mezcla. El parámetro C para el clasificador SVM es definido como $C = \frac{1}{avg(x*x)}$, que es el promedio de la norma al cuadrado de los vectores de características de entrenamiento. Finalmente debido a que el número de instancias de cada clase está desequilibrado para la mayoría de los fonos, se utiliza un parámetro de peso de clase que permite ajustar el peso del parámetro C para cada clase. Los pesos son ajustados de forma inversamente proporcional a las frecuencias de clase en los datos de entrada.

4. EXPERIMENTOS Y RESULTADOS

4.1. Resultados

En esta sección describimos diferentes variantes entrenadas de modelos DNN y contrastamos los resultados con el sistema supervector-SVM. Primero se analizan diferentes modelos para encontrar la mejor configuración de la DNN en la tarea de calificación de la pronunciación. Luego se comparan los resultados con el sistema supervector-SVM.

En las siguientes secciones, donde se busca la mejor configuración de la DNN, utilizamos cuatro métricas para medir el desempeño de los sistemas. Senone Cross Entropy, que es calculada sobre el conjunto de datos de validación de Librispeech, para determinar el desempeño de la red en la tarea de clasificación de senones. El promedio del EER, MinCost y 1-AUC sobre todos los fonos que tienen mas de 50 instancias de fonos bien y mal pronunciados sobre el conjunto de datos de desarrollo de EpaDB.

Capas

En la primera exploración de hiperparámetros buscamos la mejor arquitectura para la red DNN optimizando la cantidad de capas, el contexto usado en cada capa y su tamaño. Realizamos mucha experimentación previa que permitió determinar el rendimiento del uso de dropout, batch normalization, la tasa de aprendizaje, los diferentes tipos de características y el tamaño de mini lote. A partir de estos resultados seleccionamos los mejores valores preliminares para la experimentación en esta sección: batch normalization sin dropout, una tasa de aprendizaje con decaimiento exponencial con una tasa de decaimiento de 0.9 y un paso de decaimiento de 750 como se describe en 4.1, características normalizadas por la media y la desviación estándar y tamaño de mini lote de 16 instancias. De esta manera nos aseguramos que la mejor arquitectura es elegida con parámetros de entrenamiento y características cercanos a los óptimos. Todas las redes entrenadas tienen capas de 512 nodos a excepción de la red $C5\_TD3\_53\_1024$ que tiene capas de 1024 nodos.

Los resultados experimentales en todos los sistemas entrenados para optimizar las capas se muestran en la figura 4.2 y la tabla 4.2. Las redes con mas de una capa y mayor contexto en mayor cantidad de capas muestran mejores resultados tanto en la tarea de clasificar senones como en la de calificar fonos. La red con mayor cantidad de capas y contexto $C5\_TD3\_53$ obtiene una ganancia relativa del 7.93 % para $1 - AUC$ y del 62.93 % para Senone Cross Entropy sobre la red con la menor cantidad capas y contexto $C1\_TD\_0$. Lo primero que podemos notar

	tamaño de capas	contexto	cantidad parámetros
C1_TD_0	[512]		1.611.804
C1_TD1_5	[512]	0:[-2, 2]	1.691.676
C2_TD2_13	[512,512]	0:[-2, 2],1:[-4, 4]	4.052.508
C3_TD3_13	[512,512,512]	0:[-2, 2], 1:[-1, 1], 2:{-3, 0, 3}	3.267.612
C3_TD3_21	[512,512,512]	0:[-4, 4], 1:{-2, 0, 2}, 2:{-4, 0, 4}	3.347.484
C3_TD3_37	[512,512,512]	0:[-7, 7], 1:{-5, 0, 5}, 2:{-6, 0, 6}	3.467.292
C3_TD3_53	[512,512,512]	0:[-10, 10], 1:{-7, 0, 7}, 2:{-9, 0, 9}	3.587.100
C4_TD3_53	[512,512,512,512]	0:[-10, 10], 1:{-7, 0, 7}, 2:{-9, 0, 9}	3.850.780
C5_TD3_53	[512,512,512,512,512]	0:[-10, 10],1:{-7, 0, 7},2:{-9, 0, 9}	4.114.460
C5_TD3_53_1024	[1024,1024,1024,1024, 1024]	0:[-10, 10],1:{-7, 0, 7},2:{-9, 0, 9}	12.420.124

Tab. 4.1: Configuraciones utilizadas para encontrar la mejor arquitectura para la red TDNN optimizando la cantidad de capas, el contexto usado en cada capa y su tamaño. “Tamaño de capas” indica la cantidad y el tamaño de cada una de las capas utilizadas, “contexto” expresa el contexto utilizado en cada capa (las capas sin contexto son capas densas). Los contextos expresados entre corchetes indican que el contexto es completo, por ejemplo, [-2, 2] indica que se utilizan los contextos -2, -1, 0, 1, 2, mientras que los expresados entre llaves solo contendrán los contextos indicados, por ejemplo, {-2, 0, 2} expresa que en la capa se utilizan los contextos -2, 0 y 2. La “cantidad de parámetros” expresa la cantidad de parámetros entrenables que posee cada red. El nombre del sistema hace referencia al parámetro que se va a optimizar, en este caso capas (C) y los diferentes parámetros asociados: el numero que acompaña a la letra C indica la cantidad de capas, luego se indica cuantas capas con time delay tiene la red, por ejemplo, TD3 indica 3 capas con contexto, el número siguiente indica cuanto contexto tiene la red. Por ultimo si la red fue entrenada con capas de 1024 nodos se agrega ese número, si fue entrenada con capas de 512 nodos no se agrega nada.

es que la ganancia relativa para la clasificación de senones no se traduce directamente en una ganancia relativa comparable en la calificación de fonos. Este mismo efecto podemos verlo al agregar contexto, la red $C3\_TD3\_53$ mejora un 23,36 % el Senone Cross Entropy y un 0,65 % $1 - AUC$ con respecto a la red $C3\_TD3\_13$. Del mismo modo sucede con el aumento de capas. La red $C3\_TD3\_13$ mejora un 0.97 % $1 - AUC$ con respecto a la red $C2\_TD2\_13$ donde se agrega una capa con contexto y mejora un 4.32 % el Senone Cross Entropy. La red $C3\_TD3\_53$ mejora en un 0.98 % $1 - AUC$ con respecto a la red $C5\_TD3\_53$ donde se agregan dos capas densas y mejora un 2.07 % el Senone Cross Entropy. Por otro lado, podemos ver que agregar parámetros no da garantías de mejoras en termino de calificación de fonos, por ejemplo, la red $C5\_TD3\_53$ mejora en la tarea de clasificación de senones cuando se aumenta el tamaño de

capas en $C5\_TD3\_53\_1024$ en un 4,61 % sin embargo empeora en un 0.66 % en la tarea de calificar fonos. En el scaterplot 4.1 se muestran los resultados para las métricas senone cross entropy y 1-AUC en todos los sistemas entrenados. En esta figura podemos observar, para algunos sistemas como $C3\_TD3\_13$ y $C5\_TD3\_53\_1024$ el fenómeno en el que se producen mejoras en la clasificación de senones pero no en la calificación de fonos. Creemos que este fenómeno sucede por las particularidades de la clasificación de senones, donde el modelo debe considerar muchos detalles para separar 3100 senones distintos y eso requiere el entrenamiento de modelos complejos, mientras que para la tarea objetivo solo se debe determinar si el fono fue bien o mal pronunciado, siendo 40 la cantidad de fonos totales. Podemos concluir que para nuestra tarea objetivo no es tan importante la cantidad de parámetros utilizados sino como son organizados, es decir, la cantidad de contexto, la cantidad de capas y el tipo de combinaciones elegidas para estos parámetros. Por otro lado es importante tener en cuenta que mejoras en la clasificación de senones no garantizan mejoras en la calificación de fonos aunque la correlación entre ambos rendimientos es alta (figura 4.2).

	EER	MinCost	1-AUC	Senone Cross Entropy
C1_TD0_0	0.374	0.866	0.328	3.450
C1_TD1_5	0.362	0.849	0.316	2.581
C2_TD2_13	0.360	0.838	0.310	1.781
C3_TD3_13	0.355	0.838	0.307	1.704
C3_TD3_21	0.358	0.834	0.306	1.501
C3_TD3_37	0.357	0.834	0.305	1.362
C3_TD3_53	0.353	0.827	0.305	1.306
C4_TD3_53	0.355	0.831	0.303	1.275
C5_TD3_53	0.350	0.827	0.302	1.279
C5_TD3_53_1024	0.356	0.829	0.304	1.220

Tab. 4.2: Rendimiento de las redes entrenadas para optimización de capas.

Tasa de aprendizaje

Para ajustar la tasa de aprendizaje utilizamos la configuración de la red $C5\_TD3\_53$ que fue la mejor configuración encontrada en la optimización de capas. En este apartado llamamos a esta red $TA\_D\_0{,}01$ para ser consistentes con la nomenclatura asignada a las redes entrenadas para la optimización de la tasa de aprendizaje, las cuales serán descriptas más adelante. Se han seleccionado diferentes valores para la tasa de aprendizaje en los diferentes entrenamientos, valores constantes y con decaimiento exponencial. Como se ha mencionado en la sección 2.1.2

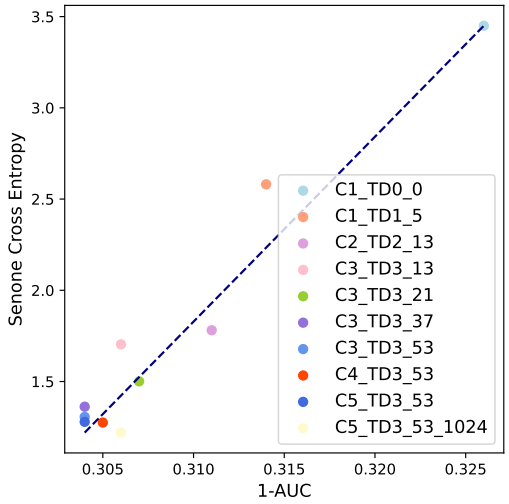

Fig. 4.1: Rendimiento en la tarea de clasificación de senones (Senone Cross Entropy) frente a rendimiento en la tarea de calificación de fonos ($1-AUC$) para todas las redes entrenadas en la optimización de capas.

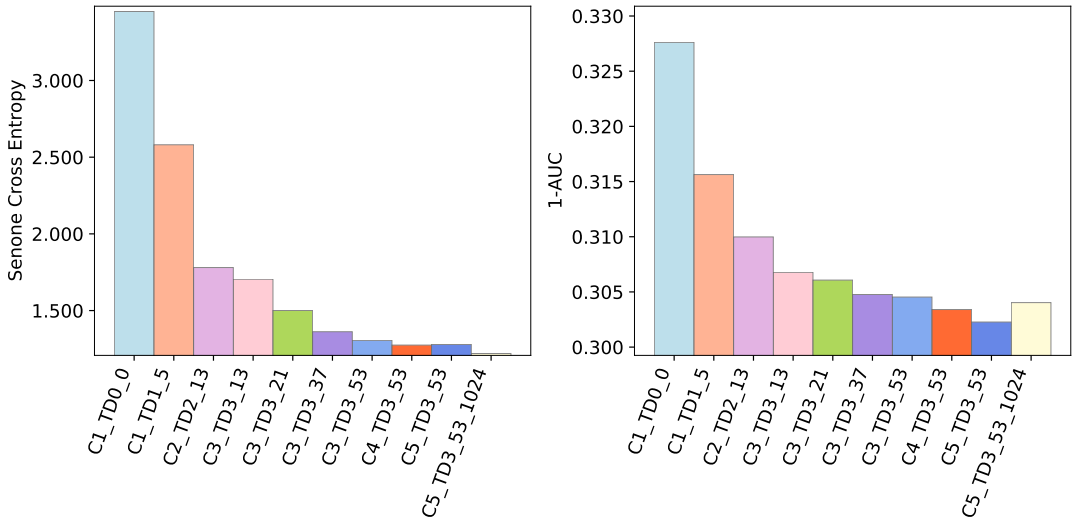

Fig. 4.2: Gráficos de barra para los resultados en la tabla 4.2. En el gráfico de la izquierda se mide el rendimiento de los sistemas entrenados para la optimización de capas utilizando Senone Cross-Entropy en el conjunto de datos de validación de Librispeech. En el gráfico de la derecha se mide el rendimiento de los mismos sistemas para la tarea de calificación de fonos utilizando la medida $1 - AUC$ sobre el conjunto de datos de desarrollo de EpaDB.

para el decaimiento exponencial deben ser seleccionados dos parámetros, la tasa t y el decaimiento p. Elegimos como tasa de decaimiento al valor 0.9. El decaimiento fue seleccionado con el fin de

reducir su valor 100 veces a lo largo de una época. Cada época cuenta con aproximadamente 500000 instancias repartidas en aproximadamente 31000 mini lotes de 16 instancias cada uno. Dado el valor de decaimiento 0.9 el decaimiento debe ser aproximadamente 750 para cumplir con el objetivo de reducirlo 100 veces a lo largo de una época. Podemos ver en la tabla 4.3 los valores iniciales y finales de la tasa de aprendizaje en cada red entrenada.

	Tasa de aprendizaje inicial	Tasa de aprendizaje final
TA_F_0.1	0.1	0.1
TA_F_0.01	0.01	0.01
TA_F_0.001	0.001	0.001
TA_D_0.1	0.1	0.001
TA_D_0.01	0.01	0.0001
TA_D_0.001	0.001	0.00001

Tab. 4.3: Configuraciones utilizadas para encontrar la mejor arquitectura para la red TDNN optimizando la tasa de aprendizaje. "Tasa de aprendizaje inicial" es el valor con el que comienza la tasa de aprendizaje y "Tasa de aprendizaje final" el valor con el que termina, para el caso donde estos valores coinciden la tasa de aprendizaje es constante. El nombre del sistema hace referencia al parámetro que se va a optimizar, en este caso tasa de aprendizaje y los diferentes parámetros asociados: F o D es el tipo de tasa utilizada, Fija o con Decaimiento exponencial, el número hace referencia al valor con el que comienza la TA.

Los resultados experimentales en todos los sistemas entrenados para optimizar la tasa de aprendizaje se muestran en la figura 4.3 y la tabla 4.4. Las redes entrenadas con tasa de aprendizaje inicial 0.1 (fija y con decaimiento exponencial) tienen un valor de senone cross entropy muy alto, si bien en el conjunto de datos de entrenamiento converge, debido a un sobre ajuste, los resultados no generalizaron en el conjunto de validación. Sin embargo podemos notar que los resultados en la calificación de fonos no están muy alejados del resto de las redes que no han sufrido sobre ajuste. El mejor resultado corresponde a $TA\_D\_0{,}01$, que coincide con la configuración usada para optimizar la arquitectura.

	EER	MinCost	1-AUC	Senone Cross Entropy
TA_F_0.1	0.418	0.937	0.391	72.215
TA_F_0.01	0.360	0.838	0.309	1.358
TA_F_0.001	0.357	0.833	0.307	1.271
TA_D_0.1	0.356	0.835	0.307	40542.504
TA_D_0.01	0.350	0.827	0.302	1.279
TA_D_0.001	0.354	0.828	0.304	1.285

Tab. 4.4: Rendimiento de las redes entrenadas para optimización de tasa de aprendizaje.

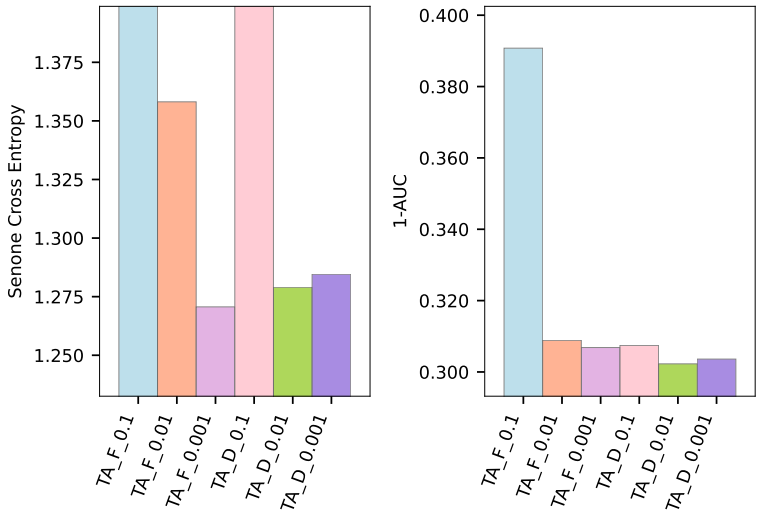

Fig. 4.3: Gráficos de barra para los resultados en la tabla 4.4. En el gráfico de la izquierda se mide el rendimiento de los sistemas entrenados para la optimización de tasa de aprendizaje utilizando Senone Cross-Entropy en el conjunto de datos de validación de Librispeech. Para la tasa de aprendizaje 0.1 fija y con decaimiento exponencial el sistema produce sobre ajuste y el valor de senone cross entropy queda por fuera del gráfico. En el gráfico de la derecha se mide el rendimiento de los mismos sistemas para la tarea de calificación de fonos utilizando la medida $1 - AUC$ sobre el conjunto de datos de desarrollo de EpaDB.

Dropout y Batchnorm

Para ajustar los hiperparámetros de dropout y batchnorm utilizamos la configuración de la red $C5\_TD3\_53$ que fue la mejor configuración encontrada en la optimización de capas. En este apartado llamamos a esta red BN para ser consistentes con la nomenclatura asignada a las redes entrenadas para la optimización de batchnorm y dropout. Se han probado combinaciones posibles de diferentes tasas de dropout y Batchnorm como se muestra en la tabla 4.5.

En la tabla 4.4 podemos ver que batch normalization produce mejoras. Este resultado es el esperado, dado que batch normalization es una técnica que da mejoras en diversas tareas. Por otro lado en los casos en que hay dropout los resultados empeoran. Suponemos que al ser arquitecturas relativamente chicas no necesitan la regularización dada por dropout.

Características

En esta sección se entrenan cuatro modelos para determinar que tipo de normalización de las características de entrada producen mejores resultados. Se han probado dos tipos de norma-

	Batchnorm	Tasa de dropout
DO_0.5	No	50 %
DO_0.2	No	20 %
DO_0.1	No	10 %
Sin-BN-DO	No	-
BN-DO_0.5	Si	50 %
BN-DO_0.1	Si	10 %
BN	Si	-

Tab. 4.5: Configuraciones utilizadas para encontrar la mejor arquitectura para la red TDNN optimizando batchnorm y dropout. "Bachnorm" indica si se utiliza la normalización por batches, "tasa de dropout" indica que porcentaje de nodos se descartan, el guión medio indica que no se utiliza dropout. El nombre del sistema hace referencia a los parámetros que se van a optimizar, en este caso dropout (DO) y batchnorm (BN) y, en caso de utilizar dropout, el porcentaje de nodos descartados. Por ejemplo, un porcentaje de dropout de 0.2 indica que se descartan aleatoreamente el 20 % de los nodos.

	EER	MinCost	1-AUC	Senone Cross Entropy
DO_0.5	0.497	0.977	0.501	6.810
DO_0.8	0.426	0.912	0.398	4.851
DO_0.9	0.397	0.895	0.356	3.811
Sin-BN-DO	0.381	0.860	0.336	2.556
BN-DO_0.5	0.367	0.851	0.317	2.313
BN-DO_0.9	0.354	0.828	0.303	1.480
BN	0.350	0.827	0.302	1.279

Tab. 4.6: Rendimiento de las redes entrenadas para optimización de batchnorm y dropout.

lización y sus combinaciones: (1) normalizar cada dimensión del vector de características por su media y desviación estándar dentro de cada muestra, (2) normalizar cada dimensión solo por su media, (3) normalizar cada dimensión solo por su desviación estándar y (4) no normalizar. Se pueden ver las combinaciones probadas en la tabla 4.7. La selección del tipo de normalización se realiza utilizando la red $C5\_TD3\_53$, la mejor configuración encontrada en la optimización de capas. En este apartado llamamos a esta red Norm-M para ser consistentes con la nomenclatura asignada a las redes entrenadas para la selección de las características.

En la tabla 4.8 y la figura 4.5 podemos ver que tanto la normalización por la media como por la desviación estándar producen mejoras en todas las métricas para la tarea de calificación de la pronunciación, los mejores resultados se obtienen cuando se normaliza solo por la media

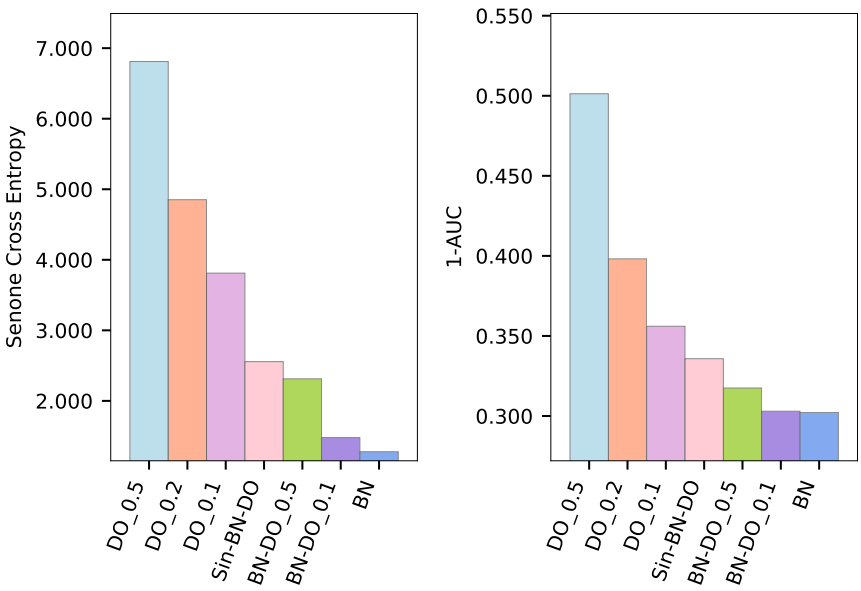

Fig. 4.4: Gráficos de barra para los resultados en la tabla 4.6. En el gráfico de la izquierda se mide el rendimiento de los sistemas entrenados para la optimización de batchnorm y dropout utilizando Senone Cross-Entropy en el conjunto de datos de validación de Librispeech. En el gráfico de la derecha se mide el rendimiento de los mismos sistemas para la tarea de calificación de fonos utilizando la medida $1 - AUC$ sobre el conjunto de datos de desarrollo de EpaDB.

	normalizar por la media	normalizar por la desviación estándar
Sin-Norm	No	No
Norm-DE	No	Si
Norm-MDE	Si	Si
Norm-M	Si	No

Tab. 4.7: Configuraciones utilizadas para encontrar la mejor arquitectura para la red TDNN seleccionando el tipo de normalización que se les aplicará a las características. Como su nombre lo indica, "normalizar por la media" indica si las características son normalizadas por su media (M) y "normalizar por la desviación estándar" si son normalizadas por su desviación estándar (DE).

o cuando se combina la normalización por la media y la desviación estándar. En este trabajo utilizaremos características normalizadas por media y desviación estándar.

	EER	MinCost	1-AUC	Senone Cross Entropy
Sin-Norm	0.364	0.852	0.314	1.345
Norm-DE	0.359	0.834	0.306	1.294
Norm-MDE	0.350	0.827	0.302	1.279
Norm-M	0.350	0.829	0.302	1.262

Tab. 4.8: Rendimiento de las redes entrenadas para la selección de características.

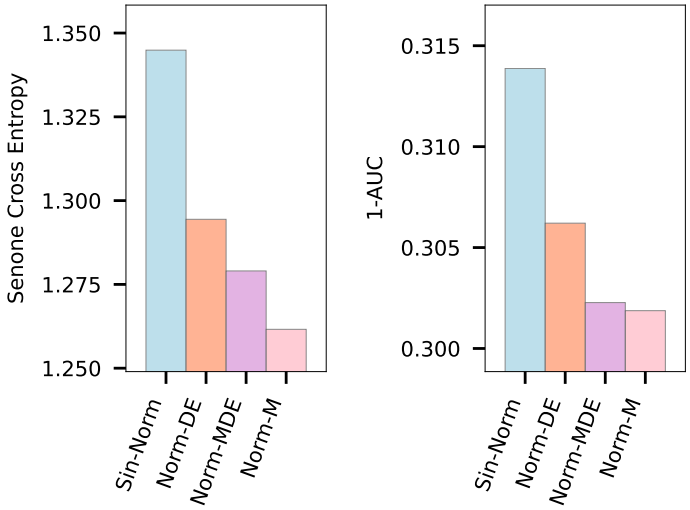

Fig. 4.5: Gráficos de barra para los resultados en la tabla 4.8. En el gráfico de la izquierda se mide el rendimiento de los sistemas entrenados para la selección de características, utilizando Senone Cross-Entropy en el conjunto de datos de validación de Librispeech. En el gráfico de la derecha se mide el rendimiento de los mismos sistemas para la tarea de calificación de fonos utilizando la medida $1 - AUC$ sobre el conjunto de datos de desarrollo de EpaDB.

Épocas

Con el fin de evitar un costo computacional excesivo, las redes utilizadas para la selección de los mejores hiperparámetros han sido entrenadas por una única época. En esta sección mostramos el rendimiento de la red con mejor rendimiento en la calificación de fonos ($C5\_TD3\_53$) entrenada por 4 épocas más. Los resultados experimentales para cada época se muestran en la figura 4.6 y la tabla 4.9. Se puede observar una clara mejora en la tarea de clasificar senones en la segunda época. Sin embargo esa mejora no se traduce en una mejora en el rendimiento para la tarea de calificar fonos. Podemos concluir que no es necesario entrenar más de una época dado que no mejora el rendimiento del sistema y lo vuelve muy costoso computacionalmente.

	EER	MinCost	1-AUC	Senone Cross Entropy
E1	0.350	0.827	0.302	1.279
E2	0.350	0.827	0.302	1.267
E3	0.351	0.827	0.302	1.268
E4	0.351	0.827	0.302	1.267
E5	0.351	0.828	0.302	1.267

Tab. 4.9: Rendimiento de la red con la mejor configuración encontrada en la optimización de hiperparametros y la selección de características en cada una de las épocas entrenada.

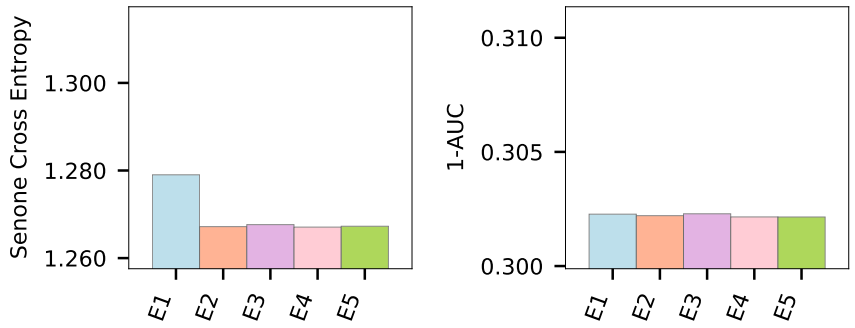

Fig. 4.6: Gráficos de barra para los resultados en la tabla 4.9. En el gráfico de la izquierda se mide el rendimiento de los sistemas entrenados a través de 5 épocas, utilizando Senone Cross-Entropy en el conjunto de datos de validación de Librispeech. En el gráfico de la derecha se mide el rendimiento de los mismos sistemas para la tarea de calificación de fonos utilizando la medida $1 - AUC$ sobre el conjunto de datos de desarrollo de EpaDB. El nombre de los sistemas indica la época que fue entrenada, por ejemplo, E3 indica que es el entrenamiento para la época 3.

DNN Vs. SVM

En esta sección comparamos el sistema GOP-DNN con la mejor configuración encontrada para la DNN a lo largo de los experimentos de optimización de hiperparámetros y selección de características con el sistema supervector-SVM.

En la tabla 4.10 reportamos el rendimiento de los sistemas para el promedio de las métricas EER, MinCost y 1-AUC calculadas sobre el conjunto de datos de evaluación utilizando el conjunto de fonos que tienen mas de 50 instancias positivas y negativas en el conjunto de datos de desarrollo. Como vimos en la sección 2.3.3 el MinCost es el costo cuando el umbral por cada fono es elegido para optimizar el costo en el mismo conjunto en el que se está evaluando. Nos gustaría poder estimar el umbral en los datos de evaluación, sin embargo, en la práctica, uno no cuenta con todos los datos en los que el sistema va a ser usado antes de usarlo. En su lugar usamos

los umbrales estimados para los datos de desarrollo. Como vimos en la sección 2.3.3 llamamos ActCost al costo en los datos de evaluación utilizando el umbral seleccionado para desarrollo, que no necesariamente es óptimo en los datos de evaluación. El ActCost es un mejor reflejo del rendimiento que el sistema tendrá en la práctica al usar umbrales predefinidos en datos de desarrollo.

En la tabla 4.10 podemos ver que, en promedio, el sistema DNN tiene mejores resultados que el sistema supervector-SVM en todas las métricas utilizadas. El sistema DNN tiene una mejora del 12,5 % para EER, 13,63 % para MinCost, 20,21 % para $1 - AUC$ y 11,13 % para ActCost. Esta ganancia ocurre a pesar de que el sistema supervector-SVM usa los datos de desarrollo para entrenar el modelo, es decir, está entrenado para la tarea de calificación de la pronunciación, mientras que el sistema DNN se entrena para la tarea de clasificar senones. Esta desventaja del sistema DNN frente al sistema supervector-SVM podría haber llevado a un rendimiento menor del sistema DNN. A pesar de esta desventaja, la mejora obtenida sugiere que un posible paso a seguir sería utilizar un enfoque basado en el aprendizaje por transferencia, adaptando el modelo DNN a la tarea de calificación de la pronunciación utilizando los datos de hablantes no nativos. El grupo de la Dra. Ferrer ya ha comenzado a explorar esta dirección, obteniendo muy buenos resultados [42].

	EER	1-AUC	MinCost	ActCost
SVM	0.409	0.381	0.946	0.988
DNN	0.358	0.304	0.817	0.878

Tab. 4.10: EER, MinCost y 1-AUC son las métricas para el conjunto de datos de evaluación, ActCost es la métrica para el conjunto de datos de evaluación utilizando el umbral seleccionado en el conjunto de datos de desarrollo.

En la figura 4.7 reportamos el comportamiento de cada sistema por fono, donde encontramos que los fonos AE, AH, AH0, AO, P y Z para las métricas EER y $1 - AUC$ en el sistema DNN tienen peores resultados que el sistema supervector-SVM. Para la métrica MinCost (marcada como una linea solida dentro de cada barra) vemos que el sistema DNN tiene peores resultados solo en los fonos AH, AH0 y P. En la figura 4.7 también reportamos el ActCost (tope de la barra) que nos permite ver el efecto de la selección de los umbrales en los datos de desarrollo. Para la mayoría de los fonos en los dos sistemas el ActCost se encuentra dentro del 10 % del MinCost, lo que indica que el umbral seleccionado en el conjunto de desarrollo generaliza bien para datos nunca vistos. El ActCost para el sistema DNN reporta peores resultados en los fonos AE, AH, AH0, D, IH, y Z. Sin embargo podemos ver que la perdida relativa para estos fonos es pequeña en comparación de la ganancia obtenida para otros fonos como AA, EH, EY, G, HH, IY, JH, K, NG, OW, R, Y y ZH. Además, cabe señalar que, para los fonos cuyo costo es cercano a 1.0, el sistema clasifica siempre como “bien pronunciado”, perdiendo la oportunidad de detectar errores.

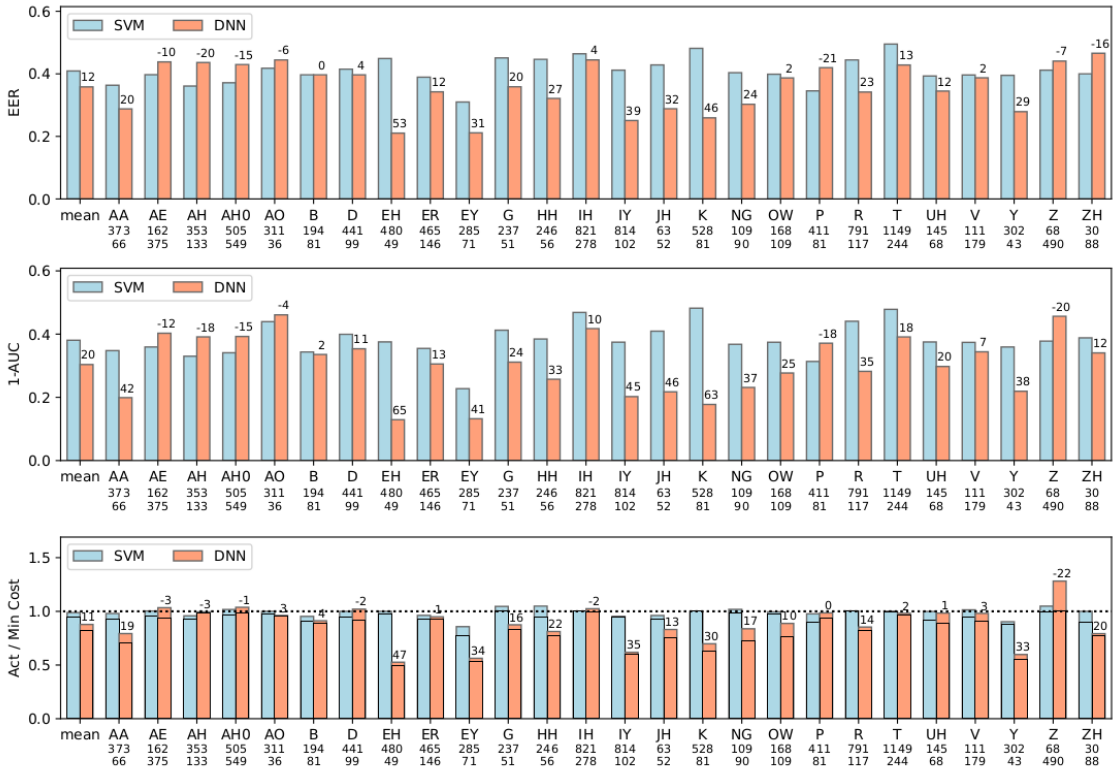

Fig. 4.7: EER, 1-AUC por fono en el conjunto de datos de evaluación. MinCost (línea negra) y ActCost (altura de la barra) por fono en el conjunto de datos de evaluación, ActCost utiliza el umbral seleccionado en el conjunto de datos de validación mientras que MinCosto utiliza el umbral seleccionado en el conjunto de datos de evaluación. Sobre el eje x se muestra el número de instancias pronunciadas correcta e incorrectamente de cada fono. El número sobre cada barra expresa la ganancia relativa.

Como se puede observar en la figura 4.7, el sistema supervector-SVM muestra un mayor número de fonos con un costo cercano a uno en comparación con el sistema GOP-DNN.

Por último, en la figura 4.8 se presentan las curvas de falsos negativos frente a falsos positivos y las distribuciones de puntuaciones para seis fonos que muestran ganancias en términos de ActCost en el sistema DNN. En la figura 4.9, se muestran seis fonos que tambien presentan ganancias en ActCost utilizando el sistema supervector-SVM.

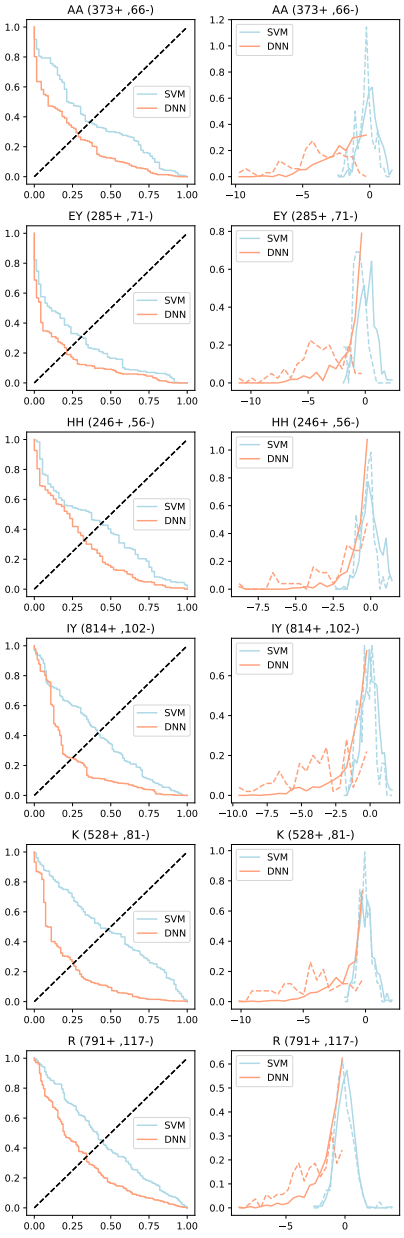

Fig. 4.8: Curvas de falsos negativos frente a falsos positivos para seis fonos con mejoras en términos de ActCost en el sistema DNN.

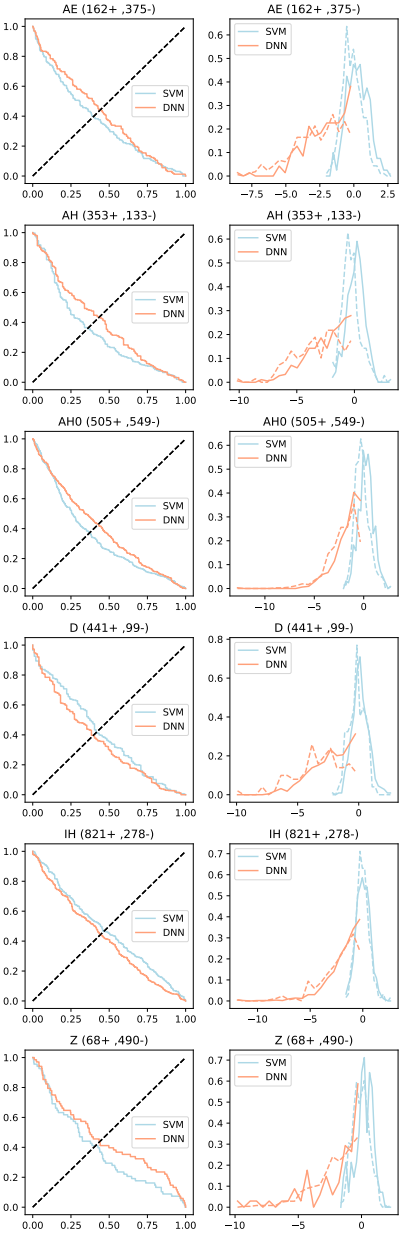

Fig. 4.9: Curvas de falsos negativos frente a falsos positivos para seis fonos con mejoras en términos de ActCost en el sistema supervector-SVM.

5. CONCLUSIONES

En el presente trabajo desarrollamos un sistema de calificación de la pronunciación basado en la medida de calidad de la pronunciación llamada Goodness of Pronunciation (GOP). Este método consiste en utilizar un sistema RAH entrenado con habla nativa de la población de interés para estimar las probabilidades a posteriori de los sonidos que el estudiante debiera haber pronunciado. Se asume que estas probabilidades serán bajas cuando la pronunciación es incorrecta, ya que las características de la señal no coincidirán con lo que el modelo entrenado con hablantes nativos espera encontrar. Originalmente los sistemas de calificación de la pronunciación que utilizaban GOP estaban dados por modelos RAH basados en HMM-GMM. En la actualidad, muchos de los sistemas de calificación de la pronunciación utilizan modelos RAH donde se reemplazan los modelos GMMs por DNNs con grandes arquitecturas (miles de parámetros).

En este trabajo desarrollamos un sistema de calificación de la pronunciación sencillo utilizando una DNN entrenada para clasificar senones (trifonos agrupados por similitud acústica) y la medida de calidad de la pronunciación GOP adaptada a DNNs. Medimos el rendimiento de pequeñas arquitecturas DNN tanto para la tarea de clasificación de senones como para la tarea de calificación de fonos. Las diferentes arquitecturas se utilizaron para seleccionar características, cantidad de épocas y los hiperparámetros: capas, tasa de aprendizaje, dropout y batch normalization. En la selección de hiperparámetros encontramos que la ganancia relativa para la clasificación de senones no se traduce directamente en una ganancia relativa comparable en la calificación de fonos. Por ejemplo, el aumento de parámetros siempre produce mejoras en la clasificación de senones, pero no sucede lo mismo con la calificación de la pronunciación de fonos, encontrando algunos casos donde el aumento de parámetros empeora la tarea de calificar fonos. Sin embargo, la organización de las capas (cantidad y contexto) produce mejoras significativas en las dos tareas. Los resultados en esta tesis sugieren que en la tarea de calificación de fonos no es necesario contar con redes de gran tamaño como las utilizadas en los sistemas RAH. Por otro lado, encontramos que con una buena selección de capas, el resto de los parámetros toma una importancia secundaria en nuestra tarea objetivo de calificación de la pronunciación de fonos, aunque no sucede lo mismo con la tarea de clasificación de senones. Para la tarea de calificación de fonos, la tasa de aprendizaje muestra un rendimiento parejo tanto para las tasas con decaimiento exponencial como para las fijas, sin embargo para la tarea de clasificación de senones tiene un resultado significativamente peor cuando es fija y comienza con un valor muy alto. Batch normalization produce mejoras en todos los casos y dropout siempre empeora los resultados para las dos tareas. Los diferentes tipos de normalización de características de entrada utilizados no producen mejoras significativas, aunque normalizar ya sea por la media, la desviación estándar o ambas produce leves mejoras en ambas tareas. Aumentar la cantidad de épocas de entrenamiento

de la red tampoco produce mejoras en la tarea de calificación de fonos.

Por último comparamos los resultados obtenidos por la DNN con mejor rendimiento para la calificación de fonos con el sistema supervector-SVM. El sistema supervector-SVM es un sistema discriminativo que recibe como entradas supervectores y determina si una instancia de fono fue bien o mal pronunciada. Los supervectores son calculados para cada instancia de fono apilando los pesos y las medias de un modelo GMM obtenido a partir de la adaptación de un GMM-UBM entrenado con instancias bien y mal pronunciadas para ese fono. Los resultados obtenidos por la DNN son comparables con el método discriminativo basado en SVM. Sin embargo, es importante notar que esta comparación no es del todo justa debido a que, a diferencia del sistema DNN, el sistema supervector-SVM aprende de datos anotados para la tarea especifica que quiere resolver. Sin embargo, a pesar de que el SVM tiene más información disponible para su entrenamiento, en nuestros experimentos no presenta mejores resultados en la tarea de calificar fonos. Esto motiva el siguiente paso en esta línea de investigación: utilizar el modelo desarrollado en este trabajo y adaptarlo a la tarea de calificación de la pronunciación usando datos no nativos. Esta es una dirección que ya se empezó a explorar en el grupo con muy buenos resultados [42]. En ese trabajo no se usó la DNN entrenada en esta tesis debido que al momento de la implementación la red aún no estaba terminada. La red usada en ese trabajo es una red entrenada para RAH, con una gran cantidad de hiperparámetros. Otro trabajo en la misma línea de investigación es reemplazar esa red por una red con menor cantidad de parámetros como la implementada en esta tesis. Por otro lado en los últimos tiempos surgieron muchas arquitecturas distintas para sistemas RAH que podrían ser investigadas para la tarea de calificación de la pronunciación. Debido a que el trabajo desarrollado en esta tesis sugiere que el gran tamaño de las arquitecturas RAH no es necesario para la tarea objetivo, podrían probarse las nuevas arquitecturas utilizadas en RAH reduciendo la cantidad de parámetros. La ventaja de las redes pequeñas es que permiten una fácil adaptación a pocos datos.

BIBLIOGRAFíA

[1] Vipul Arora, Aditi Lahiri y Henning Reetz. «Phonological feature based mispronunciation detection and diagnosis using multi-task DNNs and active learning». En: (2017).

[2] Christopher M Bishop y Nasser M Nasrabadi. *Pattern recognition and machine learning*. Vol. 4. 4. Springer, 2006.

[3] William M Campbell et al. «SVM based speaker verification using a GMM supervector kernel and NAP variability compensation». En: *2006 IEEE International conference on acoustics speech and signal processing proceedings*. Vol. 1. IEEE. 2006, págs. I-I.

[4] Lei Chen et al. «End-to-end neural network based automated speech scoring». En: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2018, págs. 6234-6238.

[5] Jan K Chorowski et al. «Attention-based models for speech recognition». En: *Advances in neural information processing systems*. 2015, págs. 577-585.

[6] Graeme Couper. «The short and long-term effects of pronunciation instruction.» En: *Prospect* 21.1 (2006), págs. 46-66.

[7] Andrea Dlaska y Christian Krekeler. «Self-assessment of pronunciation». En: *System* 36.4 (2008), págs. 506-516.

[8] Yiqing Feng et al. «SED-MDD: Towards sentence dependent end-to-end mispronunciation detection and diagnosis». En: *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2020, págs. 3492-3496.

[9] Horacio Franco, Luciana Ferrer y Harry Bratt. «Adaptive and discriminative modeling for improved mispronunciation detection». En: *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2014, págs. 7709-7713.

[10] Horacio Franco et al. «Automatic detection of phone-level mispronunciation for language learning». En: *Sixth European Conference on Speech Communication and Technology*. 1999.

[11] Horacio Franco et al. «Automatic pronunciation scoring for language instruction». En: *1997 IEEE international conference on acoustics, speech, and signal processing*. Vol. 2. IEEE. 1997, págs. 1471-1474.

[12] Geoffrey Hinton et al. «Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups». En: *IEEE Signal processing magazine* 29.6 (2012), págs. 82-97.

[13] Wenping Hu, Yao Qian y Frank K Soong. «A new DNN-based high quality pronunciation evaluation for computer-aided language learning (CALL).» En: *Interspeech*. 2013, págs. 1886-1890.

[14] Wenping Hu, Yao Qian y Frank K Soong. «A new neural network based logistic regression classifier for improving mispronunciation detection of L2 language learners». En: *The 9th International Symposium on Chinese Spoken Language Processing*. IEEE. 2014, págs. 245-249.

[15] Wenping Hu, Yao Qian y Frank K Soong. «An improved DNN-based approach to mispronunciation detection and diagnosis of L2 learners' speech.» En: *SLaTE*. 2015, págs. 71-76.

[16] Wenping Hu et al. «Improved mispronunciation detection with deep neural network trained acoustic models and transfer learning based logistic regression classifiers». En: *Speech Communication* 67 (2015), págs. 154-166.

[17] Hao Huang et al. «A transfer learning approach to goodness of pronunciation based automatic mispronunciation detection». En: *The Journal of the Acoustical Society of America* 142.5 (2017), págs. 3165-3177.

[18] Mei-Yuh Hwang y Xuedong Huang. «Subphonetic modeling with Markov states-Senone». En: *[Proceedings] ICASSP-92: 1992 IEEE International Conference on Acoustics, Speech, and Signal Processing*. Vol. 1. IEEE. 1992, págs. 33-36.

[19] Sergey Ioffe y Christian Szegedy. «Batch normalization: Accelerating deep network training by reducing internal covariate shift». En: *International conference on machine learning*. PMLR. 2015, págs. 448-456.

[20] Yoon Kim, Horacio Franco y Leonardo Neumeyer. «Automatic pronunciation scoring of specific phone segments for language instruction». En: *Fifth European Conference on Speech Communication and Technology*. 1997.

[21] Diederik P Kingma y Jimmy Ba. «Adam: A method for stochastic optimization». En: *arXiv preprint arXiv:1412.6980* (2014).

[22] Ann Lee y James Glass. «Pronunciation assessment via a comparison-based system». En: *Speech and Language Technology in Education*. 2013.

[23] Junkyu Lee, Juhyun Jang y Luke Plonsky. «The effectiveness of second language pronunciation instruction: A meta-analysis». En: *Applied Linguistics* 36.3 (2015), págs. 345-366.

[24] Wai-Kim Leung, Xunying Liu y Helen Meng. «CNN-RNN-CTC based end-to-end mispronunciation detection and diagnosis». En: *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2019, págs. 8132-8136.

[25] John Levis. «Computer technology in teaching and researching pronunciation». En: *Annual Review of Applied Linguistics* 27 (2007), págs. 184-202.

[26] Wei Li et al. «Improving non-native mispronunciation detection and enriching diagnostic feedback with DNN-based speech attribute modeling». En: *2016 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE. 2016, págs. 6135-6139.

[27] Tien-Hong Lo et al. «An Effective End-to-End Modeling Approach for Mispronunciation Detection». En: *arXiv preprint arXiv:2005.08440* (2020).

[28] Leandro Ariel Matayoshi. *Pronunciation Assessment at Phone Level for Second Language Learning*. "`http://gestion.dc.uba.ar/media/academic/grade/thesis/Tesis_Licenciatura_Leandro_Matayoshi.pdf`". Sep. de 2018.

[29] Michael McAuliffe et al. «Montreal Forced Aligner: Trainable Text-Speech Alignment Using Kaldi.» En: *Interspeech*. Vol. 2017. 2017, págs. 498-502.

[30] Seyedmahdad Mirsamadi, Emad Barsoum y Cha Zhang. «Automatic speech emotion recognition using recurrent neural networks with local attention». En: *2017 IEEE International conference on acoustics, speech and signal processing (ICASSP)*. IEEE. 2017, págs. 2227-2231.

[31] Faria Nazir et al. «Mispronunciation detection using deep convolutional neural network features and transfer learning-based model for Arabic phonemes». En: *IEEE Access* 7 (2019), págs. 52589-52608.

[32] Ambra Neri et al. «The effectiveness of computer assisted pronunciation training for foreign language learning by children». En: *Computer Assisted Language Learning* 21.5 (2008), págs. 393-408.

[33] Ambra Neri et al. «The pedagogy-technology interface in computer assisted pronunciation training». En: *Computer assisted language learning* 15.5 (2002), págs. 441-467.

[34] Leonardo Neumeyer et al. «Automatic text-independent pronunciation scoring of foreign language student speech». En: *Proceeding of Fourth International Conference on Spoken Language Processing. ICSLP'96*. Vol. 3. IEEE. 1996, págs. 1457-1460.

[35] Mary Grantham O'Brien et al. «Directions for the future of technology in pronunciation research and teaching». En: *Journal of Second Language Pronunciation* 4.2 (2018), págs. 182-207.

[36] Koji Okabe, Takafumi Koshinaka y Koichi Shinoda. «Attentive statistics pooling for deep speaker embedding». En: *arXiv preprint arXiv:1803.10963* (2018).

[37] Vassil Panayotov et al. «Librispeech: an asr corpus based on public domain audio books». En: *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE. 2015, págs. 5206-5210.

[38] Daniel Povey et al. «The Kaldi speech recognition toolkit». En: *IEEE 2011 workshop on automatic speech recognition and understanding*. CONF. IEEE Signal Processing Society. 2011.

[39] Lawrence R Rabiner. «A tutorial on hidden Markov models and selected applications in speech recognition». En: *Proceedings of the IEEE* 77.2 (1989), págs. 257-286.

[40] Douglas A Reynolds, Thomas F Quatieri y Robert B Dunn. «Speaker verification using adapted Gaussian mixture models». En: *Digital signal processing* 10.1-3 (2000), págs. 19-41.

[41] Orith Ronen, Leonardo Neumeyer y Horacio Franco. «Automatic detection of mispronunciation for language instruction.» En: *EUROSPEECH*. Citeseer. 1997.

[42] Marcelo Sancinetti et al. «A transfer learning based approach for pronunciation scoring». En: *arXiv preprint arXiv:2111.00976* (2021).

[43] Nitish Srivastava et al. «Dropout: a simple way to prevent neural networks from overfitting». En: *The journal of machine learning research* 15.1 (2014), págs. 1929-1958.

[44] Young Steve. «Large vocabulary continuous speech recognition: a review». En: *IEEE Signal Processing Magazine* 21 (1996), págs. 786-797.

[45] Ron I Thomson y Tracey M Derwing. «The effectiveness of L2 pronunciation instruction: A narrative review». En: *Applied Linguistics* 36.3 (2015), págs. 326-344.

[46] Daniela Trucco. «Educación y desigualdad en América Latina». En: (2014).

[47] Jazmın Vidal, Luciana Ferrer y Leonardo Brambilla. «EpaDB: A Database for Development of Pronunciation Assessment Systems.» En: *INTERSPEECH*. 2019, págs. 589-593.

[48] Shinji Watanabe et al. «Hybrid CTC/attention architecture for end-to-end speech recognition». En: *IEEE Journal of Selected Topics in Signal Processing* 11.8 (2017), págs. 1240-1253.

[49] Si Wei et al. «A new method for mispronunciation detection using support vector machine based on pronunciation space models». En: *Speech Communication* 51.10 (2009), págs. 896-905.

[50] Silke M Witt y Steve J Young. «Phone-level pronunciation scoring and assessment for interactive language learning». En: *Speech communication* 30.2-3 (2000), págs. 95-108.

[51] Yujia Xiao, Frank K Soong y Wenping Hu. «Paired phone-posteriors approach to ESL pronunciation quality assessment». En: *bdl* 1.782d (2018), pág. 3c89.

[52] Steve J Young, Julian J Odell y Phil C Woodland. «Tree-based state tying for high accuracy modelling». En: *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*. 1994.