Universidad de Valladolid

Escuela de Ingeniería Informática

TRABAJO FIN DE GRADO

Grado en Ingeniería Informática
Mención en Computación

Deep Learning aplicado a reconocimiento y caracterización de habla en usuarios con síndrome de Down

Autor: David Fernández García

Universidad de Valladolid

Escuela de Ingeniería Informática

TRABAJO FIN DE GRADO

Grado en Ingeniería Informática
Mención en Computación

Deep Learning aplicado a reconocimiento y caracterización de habla en usuarios con síndrome de Down

Autor: David Fernández García
Tutores: Valentín Cardeñoso Payo
César González-Ferreras

Me gustaría dedicar este trabajo a mis dos tutores, Valentín Cardeñoso y César González-Ferreras, y a David Escudero Mancebo, por su incalculable ayuda en el desarrollo de este trabajo.

"De las nubes más negras cae el agua más limpia"

Agradecimientos

Han sido muchas personas las que han ayudado a que este proyecto salga adelante, lo primero de todo me gustaría agradecer su paciencia, su ayuda y su colaboración a mis dos tutores, Valentín Cardeñoso Payo y César González-Ferreras. Sin su ayuda este trabajo no hubiera podido existir y mucho menos hubiera desembocado en lo que ha terminado siendo. También destacar a David Escudero Mancebo, cuya ayuda ha sido primordial en la realización del primer experimento y en el desarrollo del artículo que ha surgido de dicha parte.

Por otro lado, también me gustaría agradecer la ayuda de mis compañeros de laboratorio, los cuales me han ayudado no solo en el ámbito académico, sino en el ámbito social y de desconexión, clave para la realización del mismo.

Por último, agradecer a mi familia y a mi pareja la ayuda que me han brindado, leyendo borradores, ayudándome a corregir fallos, ayudándome a mejorar la redacción y una infinidad más de cosas que han sido muy importantes en el desarrollo de este estudio.

Resumen

Las personas con Síndrome de Down sufren muchos problemas a la hora de comunicarse debido a las dificultades propias de su condición. Los avances tecnológicos, en concreto la revolución del NLP, han hecho que nuestras vidas sean mucho mas fáciles y cómodas, pero estos sistemas no responden de igual forma a las personas con disfluencias. Por ello en este trabajo se pretende mejorar el rendimiento para habla anómala de estos modelos, para que sean usables por personas con disfluencias, y así poder mejorar su nivel de vida. Se plantean dos vías, la primera es el intento de conseguir un modelo reconocedor Voz-Texto para habla Down que garantice un rendimiento usable. Y la segunda corresponde a realizar una aproximación hacia la obtención de un evaluador automático del habla. En lo que respecta al reconocedor de Voz-Texto se han conseguido mejoras de hasta un 20 % con respecto al rendimiento que los mejores modelos de propósito general ofrecían para habla Down. Por último, en lo referente al evaluador automático, se puede concluir que dicha tarea no es para nada trivial y se necesita un estudio mucho más amplio y más profundo, que el que se ha realizado en este documento, para obtener un modelo usable y fiable.

Abstract

People with Down syndrome face many problems when communicating due to the difficulties inherent to their condition. Technological advances, specifically the revolution in NLP, have made our lives much easier and more comfortable, but these systems do not respond equally well to people with speech disfluencies. Therefore, this work aims to improve the performance of these models for anomalous speech so that they can be used by people with disfluencies, thereby improving their quality of life. Two approaches are proposed: the first attempt to achieve a Voice-to-Text recognition model for Down syndrome speech that ensures usable performance. The second approach involves working towards the development of an automatic speech evaluator. Regarding the Voice-to-Text recognizer, improvements of up to 20 % have been achieved compared to the performance of the best general-purpose models for Down syndrome speech. Finally, with respect to the automatic evaluator, it can be concluded that this task is by no means trivial, and a much broader and deeper study than what has been conducted in this document is needed to obtain a usable and reliable model.

Índice general

Índice de tablas

Índice de figuras

Capítulo 1

Introducción

1.1 Introducción

El Síndrome de Down (SD) es una condición genética causada, en un 95 % de los casos, por la presencia total de una copia extra del cromosoma 21. Otra de las posibles causas es lo que se denomina "Síndrome Down Mosaico", el cual es muy poco frecuente y consiste en solo algunas de las células del individuo contienen ese cromosoma extra. La última causa posible se denomina "Síndrome de Down por translocación", donde el cromosoma 21 se une a otro cromosoma, antes o durante la concepción. Este trastorno genético es una de las principales causas de discapacidad intelectual y presenta una amplia variedad de características físicas, cognitivas y médicas que afectan el desarrollo y la calidad de vida de las personas que lo padecen. Esta alteración genética afecta a un porcentaje considerable de la población mundial, aproximadamente a uno de cada 1000 recién nacidos, como indica la Organización de las Naciones Unidas (ONU). Además, se estima que unas 35.000 personas en España, 400.000 en Europa y cerca de los 8 millones en todo el mundo sufren, a día de hoy, esta alteración genética. Una de las principales características de esta condición es su heterogeneidad, lo cual quiere decir que los síntomas que padecen este tipo de personas no son para nada parecidos, sino todo lo contrario, dado que podemos encontrar personas que padezcan SD con síntomas totalmente diferentes unos de los otros. Dentro de esta heterogeneidad existen una serie de síntomas y rasgos que son muy comunes entre todas las personas con SD. El principal rasgo común, y muy diferencial de estas personas, es su apariencia física, la cual se define por un rostro aplanado, cabeza pequeña, cuello corto, baja estatura, párpados inclinados hacia arriba, pequeñas manchas blancas en el iris denominadas «manchas de Brushfield» y extremidades cortas. Alejándonos de estos rasgos comunes podemos hallar un conjunto mucho más amplio de síntomas no comunes como pueden ser: defectos cardíacos, defectos gastrointestinales, trastornos inmunitarios, apnea del sueño y muchos más.

De todo el abanico de posibles síntomas que pueden surgir en una persona padeciente de SD, en este trabajo nos vamos a centrar en los relacionados con el habla. Esta condición, para nada menor como ya se ha visto, generalmente puede acarrear problemas en el habla de muy diversos tipos, ya sea en la prosodia, fluidez, articulación, resonancia... Esto impide que muchas personas con síndrome de Down puedan comunicarse con normalidad, ya sea con otros seres humanos, o con ordenadores o sistemas similares, que se han ganado un hueco en nuestra vida. Todo lo anteriormente citado genera una gran barrera entre el mundo real y las personas que sufren esta condición, dado que la comunicación oral es una de las principales herramientas que tenemos los seres humanos para socializar y desarrollar nuestra vida cotidiana. Para hacer el debido énfasis sobre lo importante que es el la comunicación oral para nuestra sociedad vamos a recalar sobre dos artículos que escribió Albert Mehrabian [28] [27], donde se propone una fórmula denominada *55/38/7*, que quiere decir que el 55 % de nuestra

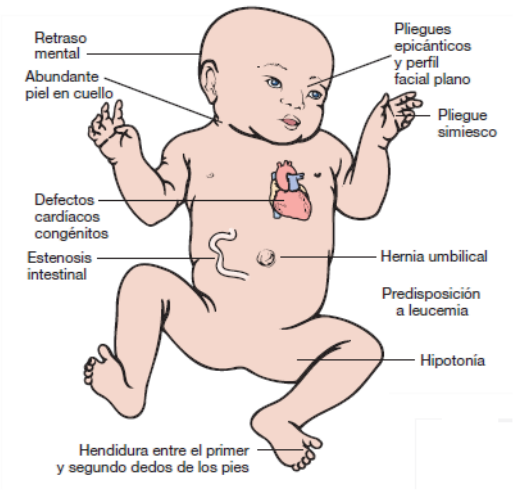


Figura 1.1: Descripción de las principales características físicas de un recién nacido con Síndrome de Down.

Imagen obtenida de la https://www.pinterest.es/pin/588353138792054923/

comunicación es lenguaje corporal, 38 % es tono de voz junto con pausas y entonaciones y el 7 % restante son palabras. Cabe destacar que esta fórmula se aplica a situaciones en las que el canal verbal y no verbal no son congruentes. Aunque parezca algo cogido con pinzas, en realidad no es tan así, existen muchas situaciones cotidianas en las que la comunicación oral se ve alterada por una gran cantidad de factores, como ruido externo, ritmo excesivo del habla, vocabulario muy especializado... En todas esas situaciones una persona con SD tendría que conseguir comunicarse usando prácticamente solo el 55 % de comunicación no verbal (lo cual es imposible) debido a sus problemas intrínsecos de comunicación oral.

Dada la situación expuesta anteriormente, en este trabajo se van a explorar dos caminos muy significativos que tienen como objetivo ayudar a que los problemas en el habla de las personas con SD se vean mejorados y paliados. El primero ambiciona conseguir un reconocedor automático del habla que tenga un rendimiento aceptable para personan que sufren SD. El reconocimiento automático de habla es una disciplina que ha avanzado significativamente en los últimos años y que tiene el potencial de hacer que la tecnología sea más accesible para los usuarios. Esta presente en infinidad de lugares en nuestro día a día. Algunos ejemplos donde podemos encontrar reconocedores del habla pueden ser: aplicaciones donde se permite el dictado por voz (Youtube, Word ...), asistentes virtuales como pueden ser Alexa o Siri, transcriptores Voz-Texto integrados en herramientas como Teams que ofrecen la posibilidad de transcribir conferencias en stream, traductores automáticos y una infinidad más de aplicaciones que forman parte de nuestra cotidianeidad y que nos facilitan la vida sin darnos cuenta. El uso de forma habitual de esta tecnología, que a partir de ahora se denominaremos ASR por sus siglas en ingles, es posible gracias al excelente rendimiento que ofrecen, cosa que como veremos en la sección 1.2, no sucede con las personas que padecen SD u otras difluencias en el habla.

El segundo camino que vamos a seguir en este documento es el de realizar un acercamiento hacia la obtención de un evaluador automático del habla. La evaluación del habla es una ciencia muy estudiada, y el uso de lingüistas o logopedas para mejorar ciertos problemas en el habla es algo muy común en nuestra sociedad. Tanto es así, que podemos encontrar estas figuras en centros de enseñanza, hospitales, residencias... y obviamente desempeñan un papel fundamental dado que la capacidad de comunicarse de forma correcta, haciendo que los receptores entiendan perfectamente el mensaje que se quiere transmitir, es algo vital para poder convivir en sociedad. Como ya se ha mencionado, las personas con SD sufren de diversas difluencias en el habla y gran parte de ellas acuden a logopedas para intentar mejorar su habla y poder comunicarse con normalidad como cualquier usuario de habla

típica.

1.2 Motivación

Las motivaciones para realizar este trabajo son muchas, y de muy diversa índole, dado que el estudio y desarrollo de herramientas que faciliten la vida personas que sufren discapacidades es una tarea muy interesante, gratificante y satisfactoria. La motivación principal para elegir una condición como el SD y no otra cualquiera, ha sido la heterogeneidad que se puede observar en la condición. Esto supone todo un reto ya que las herramientas o programas que se desarrollen no pueden ser de ámbito específico y centrarse solo en una anomalía como podría ser la disartria, que es algo muy común como se puede ver en [34], sino que deben ser de propósito general para poder abarcar al amplio espectro que el SD nos ofrece. Todo ello conlleva un apasionante reto que merece la pena afrontar.

Figura 1.2: Ejemplificación de la heterogeneidad presente en el Síndrome de Down

La razón por la cual se ha decidido tomar el camino de construir un sistema ASR para habla Down reside en el mal rendimiento que los sistemas más próximos al estado del arte ofrecen. El rendimiento de estos sistemas para habla típica es excepcional, ofreciendo errores cercanos al 2-3 %, como se verá a lo largo de este documento, en una gran cantidad de idiomas. Sin embargo, su eficacia y precisión pueden variar significativamente entre diferentes grupos demográficos y condiciones de habla [25, 4, 7]. En concreto, su aplicación a poblaciones específicas como las personas con síndrome de Down, presenta una serie de dificultades [16]. La solución adoptada en estos casos, es la adaptación de los sistemas de reconocimiento de habla para garantizar una mayor precisión y accesibilidad [19, 50]. Sin embargo, los sistemas de reconocimiento de voz actuales no logran obtener resultados comparables para personas con síndrome de Down en comparación con aquellas con desarrollo típico [5] obteniendo como mejores resultados errores entre un 20-30 %. Conseguir mejoras en estos rendimientos supondría acercar la tecnología actual a las personas con SD, lo cual podría beneficiar mucho su inserción social y su calidad de vida. Otra razón por la que se ha decidido seguir este camino, es que es un campo de estudio con bastante participación, lo cual ayuda y facilita el trabajo, al existir muchos estudios que pueden ofrecer ideas y resultados con los cuales inspirarse para el desarrollo del presente documento. Todo esto resulta ser de mucha utilidad para un trabajo de iniciación a la investigación, como es el caso de este.

Por último, el principal motivo realizar un acercamiento a la obtención de un evaluador automático del habla es que, la evaluación del habla suele ser una tarea muy costosa, tanto económicamente como en términos temporales, debido a que es una tarea muy difícil y poco automatizada. Lo más común en estos casos es que la persona que sufre ciertas anomalías en el habla interaccione o bien directamente con un lingüista, el cual le va corrigiendo y enseñando como corregir sus defectos en el habla, o bien con un juego/programa digital, donde el paciente interacciona con el juego y un lingüista observa la interacción

y corrige lo necesario. Como se puede observar, en todas las situaciones expuestas se necesita de un factor humano, lo cual limita mucho el aprendizaje, dado que la presencia de la especialista supone un costo económico, y dado que esta metodología no permite al paciente practicar desde su casa, al no tener la posibilidad de recibir un feedback sobre su desempeño. Otra de las razones es, que la evaluación del habla es una tarea con una fuerte carga de subjetividad, dado que diferentes evaluadores pueden reaccionar distinto ante la misma locución, o incluso un mismo evaluador puede modificar su opinión al escuchar una locución repetidas veces, siendo o no consciente de ello. Por ello, el desarrollo de un evaluador automático y fiable implicaría una facilidad adicional para los pacientes ya que podrían trabajar en casa, sin necesidad de un profesional supervisando, y con la certeza de que las correcciones que se le están realizando son correctas y acertadas, agilizando así su aprendizaje. Por otro lado, también ayudaría mucho a la comunidad de lingüistas ya que seria un gran acercamiento al desarrollo de una métrica evaluadora única, lo que facilitaría y mejoraría mucho su trabajo.

En conclusión, la motivación principal de este documento es la de ayudar y facilitar el día a día de las personas con SD, centrándose en la parte del habla, que toca muy de cerca a las personas que sufren esta alteración genética tan peculiar.

Capítulo 2

Objetivos y Alcance

2.1 Objetivos

El principal objetivo de este trabajo es desarrollar herramientas lingüísticas, que sean útiles y usables para personas que tengan SD. Principalmente, desarrollar un sistema ASR que ofrezca rendimientos aceptables en la tarea de transcripción voz-texto para personas con SD. Como ya se ha mencionado en la sección 1.1 los rendimientos de los sistemas ASR para personas que sufren de problemas en el habla son muy malos, e imposibilitan el uso de muchas tecnologías a personas de esta índole. Cumplir este objetivo no es para nada sencillo, debido a que el principal problema por el cual los modelos de *Deep Learning* no funcionan bien en estos casos es la escasez de datos. En general, estos modelos necesitan de muchas muestran para poder extraer correctamente los patrones que se buscan, y por tanto, el numero de muestras y su diversidad, se vuelven un factor crítico. Aun así, la astucia humana ha desarrollado técnicas y mecanismos que permiten, con un número de muestras limitadas, obtener resultados decentes, algunas de ellas se van a usar en el desarrollo de este trabajo, como veremos. Para la consecución de este objetivo se van a aplicar técnicas como *fine-tuning*, *data augmentation* ... sobre el modelo preentrenado Whisper [35] (todos estos términos y conceptos se explicarán en profundidad en el capítulo 4). Aun siendo el objetivo principal el desarrollo del modelo ASR, no es el único objetivo de esta parte. La base de querer que el trabajo sirva también como una iniciación a la investigación, hace que a raíz de este objetivo tan amplio surjan una serie de objetivos secundarios que también son relevantes e importantes en este trabajo:

- Familiarizarse con el uso de modelos de *Deep Learning*, más concretamente con la arquitectura propuesta en el famoso y tan influyente artículo de 2017, "Attention is all you Need" [47].
- Entender con detalle el funcionamiento del novedoso modelo de libre uso de OpenAI Whisper.
- Entender cual es el procedimiento del *fine-tuning* y sus principales diferencias con el entrenamiento básico de un modelo de *Deep Learning*.
- Realizar un pequeño acercamiento al mundo del aumento de datos y a sus diversas técnicas.
- Realizar una introducción académica al mundo del *Procesamiento del lenguaje natural* (PLN, o NLP, en ingles).

Como objetivo principal secundario tenemos el comenzar a recorrer el camino hacia la obtención de un evaluador automático del habla. Esta parte se cataloga como "objetivo principal secundario" porque se le ha dedicado mucho menos tiempo que a la parte de desarrollar el ASR. Para la consecución de este

objetivo se van a utilizar de nuevo, modelos preentenados de *Deep Learning*, en este caso el Wav2Vec [2], técnicas mencionadas anteriormente como *fine-tuning*, nuevas arquitecturas y una métrica usada para medir la calidad del habla denominada *Goodness of Pronunciation* (GoP). Al igual que anteriormente, también cabe destacar una serie de objetivos secundarios de esta parte:

- Entender con detalle el funcionamiento del modelo Wav2Vec
- Entender la base científica y matemática de la métrica Gop
- Familiarizarse con nuevas arquitecturas de *Deep Learning*
- Realizar un breve acercamiento al mundo de la gramática, fonología y fonética, enfocado al sector de la Inteligencia Artificial (IA).
- Realizar una introducción académica al mundo del *Procesamiento del lenguaje natural* (PLN, o NLP en ingles).

Por último, este trabajo también tiene como objetivo ser una introducción al mundo de la investigación académica. Es por ello, que a lo largo del documento se intenta seguir una metodología científica, y es por eso por lo que el trabajo no sigue una estructura habitual, sino una más enfocada a un trabajo de investigación académica. El enfoque académico hace que no exista, de base, una planificación muy estructurada y detallada, la cual se va seguir fielmente, dado que el desarrollo del trabajo esta muy influenciado por los resultados y las nuevas ideas que van surgiendo. En el comienzo del trabajo existían una serie de tareas a realizar, pero esas tareas se han ido modificando y cambiando al mismo tiempo que la investigación avanzaba, debido a: nuevas ideas, resultados que hacían tomar un camino u otro y una infinidad más de factores propios de un trabajo de investigación. Aun así, se van a detallar todas las tareas que se han realizado para la consecución de los objetivos propuestos y en la sección 3 se enseñará un "proto-plan", que más o menos se tenia en mente al inicio de la investigación, y se explicará como ha ido cambiando a lo largo del desarrollo.

2.1.1 Tareas a realizar

1. Investigación sobre los distintos modelos preentenados existentes y utilizados comúnmente para la realización de las metas anteriormente citadas.
2. Investigación sobre diversos corpus, tanto de habla anómala como de habla típica, que se puedan emplear en la experimentación del trabajo.
3. Estudio de los modelos elegidos (Whisper y Wav2Vec) y de los corpus seleccionados, para saber la naturaleza de los mismos y poder así explicar ciertos comportamientos.
4. Estudio e investigación de conceptos como *GoP*, *data augmentation* o *fine-tuning* con el fin de saber como y donde emplearlos en el desarrollo del trabajo.
5. Familiarización con el uso de las biblioteca y técnicas de programación necesarias para poder utilizar los modelos y los corpus.
6. Realización de la experimentación y obtención de resultados.
7. Discusión de los resultados y búsqueda de artículos con resultados similares que ayuden a entender los obtenidos.
8. Obtención de conclusiones y de las herramientas finales

Cabe destacar que el desarrollo de las tareas no ha sido tan lineal como lo mostrado anteriormente, debido a la naturaleza del trabajo ha habido sucesivos bucles a lo largo del desarrollo. El principal se encuentra entre los puntos *6-7-8*, donde ha habido bastantes iteraciones realizando diversas pruebas y observando que resultados se obtenían. También ha habido bucles en los puntos de investigación, debido principalmente a que según se iba avanzando y entendiendo más, nos percatábamos de cosas que podían funcionar y servir y volvíamos hacia atrás para investigar sobre ellas. Por último, el avance de la tecnología también ha generado algún bucle, como por ejemplo con el modelo Whisper, el cual empezamos usando su versión *whisper-large-V2*, pero a mitad del trabajo se lanzó su versión *whisper-large-V3*, lo cual nos hizo probarla y experimentar con ella también.

Capítulo 3

Metodología y Plan de Trabajo

Debido a la naturaleza propia del trabajo, el cual es un trabajo de investigación, se ha decidido seguir una metodología científica apropiada para los tipos de experimentos que vamos a realizar. Los pasos que vamos a seguir para la realización del trabajo de pueden ver a continuación:

- **Análisis de las soluciones existentes**
- **Preparación y realización de experimentación**
- **Evaluación de los resultados**
- **Extracción de conclusiones**

Dentro de un trabajo de índole científica lo primero que debemos realizar es un **análisis de las soluciones existentes**. Es fundamental que antes de empezar a pensar o a realizar cualquier tipo de prueba se conozca que es lo que se ha hecho con anterioridad con respecto al problema que estamos abordando. Esto puede ayudar a obtener nuevas ideas y a elegir que camino se quiere seguir para avanzar en la solución del problema. También, otro aspecto muy importante, es que un buen análisis de las soluciones anteriores puede evitar perder tiempo en la exploración de soluciones ya probadas por otros autores y que no fueron exitosas.

Una vez conocemos que es lo que se ha hecho y hemos decidido cual es la vía que vamos a seguir en base a lo que hemos leído e investigado, debemos realizar primero una **preparación de la experimentación**. Esto engloba la preparación de las cosas que necesitamos, como modelos, corpus u otras herramientas. Cabe destacar que dicha preparación no incluye solo la elección de los modelos y de los corpus, sino también la preparación de los mismos, ya que estos puede ser que necesiten sufrir ciertas modificaciones para que poder ser usados en el contexto que se desea. Por ultimo, dentro de la preparación, también debemos de establecer un procedimiento experimental, es decir, un plan de como se van a integrar las diferentes partes a la hora de realizar la experimentación. Esto también incluye la generación de un esquema el cual es el que van a seguir todos los experimentos que se hagan. Esta estructura es fundamental, debido a que facilita la replicación y comparación de los experimentos. Como último paso se ha de decidir cual es la experimentación que se va a realizar, es decir, el numero y orden de experimentos que se van a desarrollar en el trabajo, obviamente siguiendo el esquema anteriormente establecido.

Posteriormente, tras la realización de la experimentación, se deben recopilar los resultados obtenido y realizar una **evaluación de resultados**. Este análisis puede ser muy trivial, como realizar una simple comparación entre dos experimentos diferentes o puede acarrear un desarrollo más complejo. Si ese es el caso, se deberá realizar un esquema, similar al realizado en la experimentación, que todo el análisis

debe seguir con el fin de garantizar resultados justos y comparables. A lo largo del documento nos encontraremos con un ejemplo de cada tipo, por un lado en el experimento 1 (Reconocedor de Voz Down, capítulo 5) se nos presenta un claro ejemplo donde el análisis de los resultados no va más allá de la tarea de comparar una serie de valores e intentar buscar explicaciones a las diferentes variaciones que se puedan apreciar. En el caso del experimento 2 (Calidad del habla, capítulo 6) nos encontramos en el caso opuesto, debido a que realizar una simple comparación no nos enseña demasiado del resultado del experimento, lo cual nos obliga a realizar un análisis más profundo con el objetivo de extraer conclusiones y poder saber cual es el camino a seguir.

Finalmente, el ultimo paso que nos queda es realizar la **extracción de conclusiones**. En este paso debemos reflexionar sobre los resultados del análisis anteriormente realizado y obtener conclusiones y explicaciones. Estas últimas no deben de ser solo en el sentido de saber porque los resultados han sido del tal forma, sino que deben ir más allá. Una vez terminadas estas conclusiones se deben de saber 3 cosas: porque los resultados han sido los que han sido, que innovaciones se han aportado al campo de estudio con el trabajo realizado y por último, cual es el camino a seguir después de este trabajo, es decir, lo que se conoce comúnmente como aproximaciones futuras.

3.1 Planificación

Como se ha mencionado anteriormente el carácter de investigación que tiene el trabajo hace que no exista una planificación fija y estable que se haya seguido a lo largo de todo el trabajo, sino que se ha ido sometiendo a cambios y actualizaciones. Es por eso, que el desarrollo de esta sección tendrá un formato un poco inusual, donde se va a proponer una planificación inicial y se va a ir mostrando su evolución y sus cambios. La tabla 3.1 ejemplifica como era la planificación inicial del trabajo.

Nombre de actividad	**Semanas**
Estudio de los conceptos básicos de PLN	1-2
Análisis y aprendizaje de las técnicas y herramientas ASR	3-4
Experimentación con diversos modelos ASR y elección del más adecuado	5
Comprensión profunda del modelo ASR elegido	6
Investigación y experimentación con dicho modelo	6-9
Obtención y discusión de los resultados	10
Redactar la memoria del TFG en base a los resultados obtenidos	11-14

Tabla 3.1: Primera iteración de la planificación del trabajo

Como bien se puede observar, al inicio del trabajo solo se planteaba el camino de conseguir un modelo ASR que tuviese un buen rendimiento para la tarea de transcripción voz-texto de habla Down castellana. Todas las tareas que se muestran en la tabla 3.1 tienen como intención conseguir ese objetivo, que es el objetivo principal final de este trabajo, como se puede ver en el capítulo 2. En un principio, la planificación consideraba empezar el desarrollo en diciembre y que su finalización se hallase en febrero, es por eso que se puede observar como la estimación inicial de la duración del trabajo es de 14 semanas.

Durante el desarrollo del trabajo, por el mes de enero, surgió la posibilidad de explorar una nueva vía, relacionada en cierta medida con el trabajo que se había venido realizando. Esta nueva vía es la denominada "objetivo principal secundario" en el capítulo 2, y consistía en empezar a recorrer el camino hacia la obtención de un evaluador automático del habla. Dado que se decidió recorrer esta vía, la fecha de finalización del trabajo se pospuso hasta abril, se dejo a un lado el trabajo desarrollado hasta entonces y se comenzó a explorar este nuevo camino. La planificación general del trabajo, dada esta nueva tesitura, quedaba entonces como se muestra en la tabla 3.2

Se puede apreciar como la segunda iteración engloba explorar los dos caminos que se han propuesto como objetivos. Dado que se añadió un nuevo bloque de trabajo considerable, la finalización del trabajo

Nombre de actividad	**Semanas**
Reconocedor Voz-Texto Down	
Estudio de los conceptos básicos de PLN	1-2
Análisis y aprendizaje de las técnicas y herramientas ASR	3-4
Experimentación con diversos modelos ASR y elección del más adecuado	5
Comprensión profunda del modelo ASR elegido	6
Investigación y experimentación con dicho modelo	6-7
Evaluador Automático	
Decisión de posponer el trabajo actual y seguir la nueva vía	7
Estudio de los conceptos básicos de la evaluación del habla	7-8
Estudio de los conceptos básicos de fonética y fonología	7-8
Análisis y aprendizaje de las técnicas y herramientas de evaluación	8-9
Comprensión profunda del código a utilizar	9-11
Investigación y experimentación	11-13
Obtención y discusión de los resultados	14
Reconocedor Voz-Texto Down	
Finalización de la experimentación con el modelo ASR	15-16
Obtención y discusión de los resultados	17
Redactar la memoria del TFG en base a los resultados obtenidos	17-20

Tabla 3.2: Segunda iteración de la planificación del trabajo

se pospuso hasta marzo, con la intención de conseguir explorar ambas vías.

La planificación que se muestra en la tabla 3.2 no es todavía la definitiva, ya que el camino de conseguir un evaluador automático resulto ser más complejo de lo esperado y se decidió tomar la medida de posponer, como trabajo futuro (fuera de este TFG), la experimentación y realizar solo un análisis del modelo base, sin ninguna implementación nueva ni nada por el estilo. Una vez tomada esta decisión se decidió terminar y ampliar el trabajo anterior (reconocedor Voz-Texto Down), dedicándole más tiempo, con la intención de obtener mejores resultados y conclusiones. En relación a todo lo citado anteriormente, la planificación global del trabajo, y esta vez si, la final, quedaría como se puede ver en la tabla 3.3.

La tabla 3.3 corresponde a la planificación final que se ha seguido y completado en este trabajo.

Nombre de actividad	**Semanas**
Reconocedor Voz-Texto Down	
Estudio de los conceptos básicos de PLN	1-2
Análisis y aprendizaje de las técnicas y herramientas ASR	3-4
Experimentación con diversos modelos ASR y elección del más adecuado	5
Comprensión profunda del modelo ASR elegido	6
Investigación y experimentación con dicho modelo	6-7
Evaluador Automático	
Decisión de posponer el trabajo actual y seguir la nueva vía	7
Estudio de los conceptos básicos de la evaluación del habla	7-8
Estudio de los conceptos básicos de fonética y fonología	7-8
Análisis y aprendizaje de las técnicas y herramientas de evaluación	8-9
Análisis y comprensión profunda del código y modelos a utilizar	9-11
Reconocedor Voz-Texto Down	
Finalización y ampliación de la experimentación con el modelo ASR	12-16
Obtención y discusión de los resultados	17
Redactar la memoria del TFG en base a los resultados obtenidos	17-20

Tabla 3.3: Segunda iteración de la planificación del trabajo

Capítulo 4

Marco Conceptual

Dentro de este capitulo se van a explicar de forma profunda y detallada lo conceptos más importantes que son necesarios para comprender de forma correcta este documento. Este capitulo se va a dividir en dos amplias secciones, una por cada camino que vamos se ha recorrido (*Reconocedor Voz-Texto Down* y *Evaluador Automático del Habla*). Dentro de cada sección se describirán los conceptos principales para el total entendimiento de dicho camino.

4.1 Reconocedor Voz-Texto Down

Como bien se ha mencionado con anterioridad, el objetivo de este camino es conseguir un reconocedor voz-texto para habla Down que ofrezca rendimientos usables y mejores que los que ofrecen los principales modelos ASR de la actualidad. A continuación se van a explicar los principales conceptos necesarios para garantizar la comprensión de esta parte:

4.1.1 Whisper

Whisper es un modelo multilenguaje y multitarea, entrenado con más 680.000 horas de audios etiquetados. He aquí donde reside su diferenciación. La gran mayoría de modelos, como Wav2Vec u otros, son entrenados con millones de horas de audio sin etiquetar, es decir, son modelos de aprendizaje no supervisado. Esto deriva en tres cosas fundamentales. La primera es que tienen una gran capacidad de codificar audios. La segunda es que la capacidad de su decodificación ni se acerca a la de su codificación. Finalmente, la tercera es que suelen generalizar mal y para ser usados en terrenos novedosos necesitan un fine-tuning con muchos datos. Y justo es eso lo que Whisper intentan paliar, el objetivo de Whisper es ser un modelo robusto frente a nuevos datasets sin la necesidad de realizarle un fine-tuning enorme.

Descripción del modelo

Para poder entender bien el modelo y como se ha llegado a donde estamos actualmente se necesita un poco de historia. A finales de los años 80, se presento una nueva arquitectura, las Redes Recurrentes [38]. En esta nueva arquitectura el output de una iteración se encadena con el input de la siguiente para crear así una forma de procesar secuencias. Haciendo que el output de una iteración se encadene con el input de la siguiente lo que estamos haciendo es crear una especie de "memoria", de forma que para generar el siguiente token se tienen en cuenta los anteriores, es decir, se esta creando un contexto. Pero esta arquitectura tenía un problema, y es que a medida que la secuencia crecía en tamaño, los primeros tokens procesados perdían peso, es decir, la red se olvidaba de ellos. Para paliar este efecto se

implementaban mecanismos de atención para que la red no se olvidase de estos primeros tokens. Y es aquí donde nacen los Transformers [47], cuando en 2017 los investigadores se dieron cuenta de que la red recurrente no era necesaria, y que con el mecanismo de atención bastaba para obtener incluso mejores resultados. Aunque la principal novedad de los Transformers es que permiten procesar la secuencia al completo en paralelo, es decir, todos los tokens a la vez, y lo hacen teniendo en consideración el orden de los tokens dentro de la secuencia, lo que ocasiona un entrenamiento mucho mas rápido y por tanto la posibilidad de entrenar con muchos mas datos.

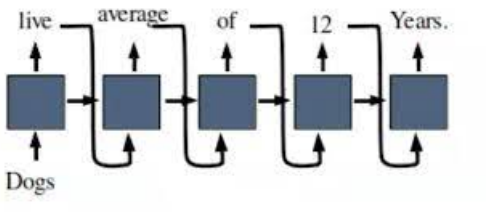


Figura 4.1: Arquitectura de Red Neuronal Recurrente

Una vez contextualizada un poco la historia de los Tranformers, y por tanto de Whisper, podemos pasar a entender su arquitectura. En la imagen 4.2, se puede ver una foto de la arquitectura. A continuación, vamos a describir cada bloque que la compone para así hacernos una idea de como funciona este modelo.

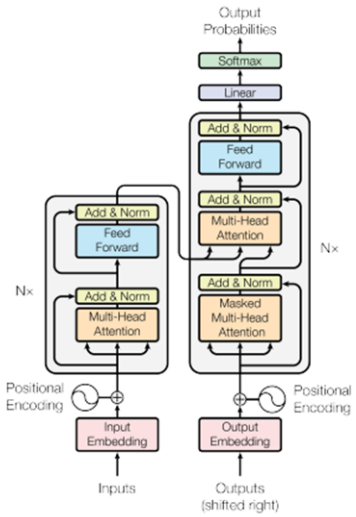


Figura 4.2: Arquitectura de Transformer.
Imagen obtenida del artículo [47].

Como podemos ver la arquitectura se divide en dos partes principales, el encoder y el decoder. El encoder es el que se encarga de codificar el input, es decir, transformar el input (un audio, representado vectorialmente, en nuestro caso, conocido como embedding) en una representación correcta y con sentido para el modelo. El decoder es la parte que se encarga de transformar ese embedding que ha generado el encoder en una representación vectorial que posteriormente se pueda convertir a un formato humano (en nuestro caso, la transcripción, es decir, texto). Cómo actuara el decoder sobre el embedding dependerá de para que tarea haya sido entrenado el modelo. Como se puede ver en la imagen 4.2, durante el entrenamiento, en el encoder se introduce como input el propio input del entrenamiento y al decoder se le introduce como input el etiquetado del input que se le ha introducido al encoder.

Lo verdaderamente revolucionario del Transformer reside en un modulo llamado *Positional Encoding*. Este modulo es el que hace que los Transformers sean tan sorprendentes, ya que este modulo genera un nuevo vector, utilizando una codificación sinusoidal, que se sumará al embedding y representara así la posición que ocupa el embedding dentro de la secuencia inicial.

Descripción del encoder

Vamos a centrarnos ahora en el encoder y en su arquitectura. Para facilitarnos la comprensión en vez de secuencia y token vamos a utilizar palabra y frase, pero no hay que olvidar que este es un modelo Secuencia-Secuencia y por tanto se puede utilizar con cualquier par de secuencias imaginable. Una vez ya tenemos un embedding con la posición codificada se introduce dentro del encoder. Como podemos ver el primer bloque que aparece es la *Multi-Head Attention*, en ella se introducen tres copias del embedding original, mas adelante explicaremos el porque, pero ahora vamos a centrarnos en ese embedding que como podemos ver no se introduce en la *Multi-Head Attention*, sino que se va directamente al bloque *Add & Norm*. Esta técnica se conoce como conexión residual y consiste básicamente en transportar el embedding sin procesar al final, donde en el bloque *Add & Norm* se sumara con el embedding procesado. Este mecanismo esta demostrado que ayuda al entrenamiento de modelos que son muy profundos [14].

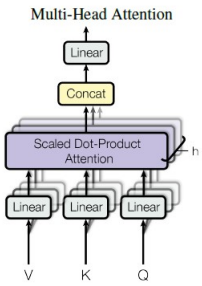


Figura 4.3: Arquitectura del bloque *Multi-Head Attention*.
Imagen obtenida del artículo [47].

Vamos a adentrarnos ahora dentro del bloque *Multi-Head Attention*. Se denomina "multi" porque no tiene una sola cabeza de atención sino que tiene varias. Vamos a explicar como funciona una, en la imagen 4.3 se puede ver como es el bloque *Multi-Head Attention* por dentro. Como se puede ver, este bloque recibe tres veces el embedding original, y eso es porque cada uno se transformará dentro de la atención y tendrá una función específica. Esta transformación se hace multiplicando cada embedding por una matriz de pesos, pesos que son los que se ajustan con el entrenamiento. Los tres embedding reciben los nombres de query (**q**), key (**k**) y value (**v**). Como se puede observar hay un bloque, el *Linear* del cual no hemos hablado. Al tener muchas cabezas, a cada cabeza no le damos como input los tres embeddings **q**, **k** y **v**, sino que lo que vamos a hacer es utilizar ese bloque y como por así decirlo, partir los embeddings originales y darle a cada cabeza un trocito. Cada trocito puede representar algo de la palabra o de la frase, como por ejemplo, características semánticas o gramaticales. De esta forma cada cabeza se especializará en procesar algo muy especifico. Por lo tanto, la salida definitiva del bloque *Multi-Head Attention* sera la concatenación de las salidas de cada una de las diferentes cabezas, como se puede ver en la imagen 4.4.

Cada atención realiza la siguiente operación con los vectores anteriormente mencionados: $softmax(\frac{QK^T}{\sqrt{d_k}})V$. Hemos dejado pendiente el explicar que son **q**, **k** y **v**. Si recordamos, cada uno de ellos se obtiene multiplicando el embedding original por una matriz de pesos distinta y cada uno representa una cosa. Se podría decir que **k** describe como es la palabra original, es decir, la que representa el embedding. Por otra parte, **q** describe lo que busca la palabra original, por ejemplo un verbo buscará un sujeto o un adjetivo buscara un nombre... Por ultimo, **v** es una representación precisa y correcta del

$$\text{MultiHeadAttention}(\mathbf{X}) = (\mathbf{head}_1 \oplus \mathbf{head}_2 ... \oplus \mathbf{head}_h)\mathbf{W}^O$$
$$\mathbf{Q} = \mathbf{X}\mathbf{W}_i^Q \; ; \; \mathbf{K} = \mathbf{X}\mathbf{W}_i^K \; ; \; \mathbf{V} = \mathbf{X}\mathbf{W}_i^V$$
$$\mathbf{head}_i = \text{SelfAttention}(\mathbf{Q}, \mathbf{K}, \mathbf{V})$$

Figura 4.4: Ecuación que realiza el bloque Multi-Head Attention.
Imagen obtenida del artículo [47].

embedding original, es decir, de la palabra original. No confundir **k** y **v**, **k** representa una descripción de la palabra y **v** representa la propia palabra. Se podría decir que el vector **v** seria lo que la red recurrente original (sin mecanismos de atención) nos daría al procesar una palabra.

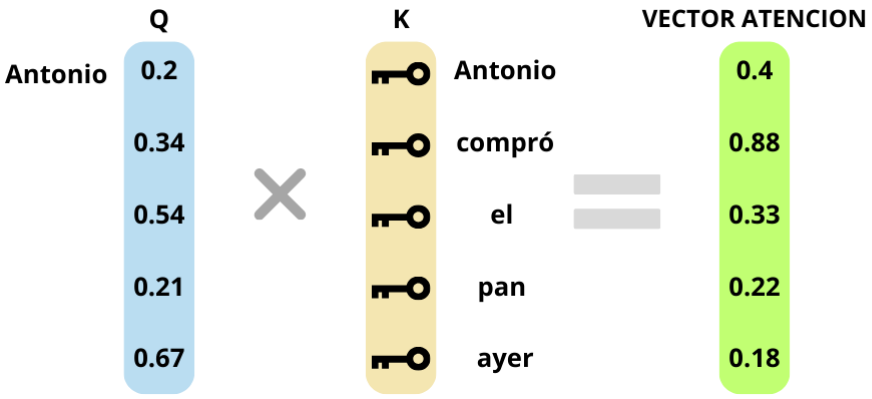


Figura 4.5: Descripción de como se obtiene el vector atención

Bien, y ahora, ¿por que realizamos esa operación $(softmax(\frac{QK^T}{\sqrt{d_k}})V)$?. Pues veamos, al multiplicar **q** por **k**, como se puede ver en la formula, lo que estamos haciendo es calcular una medida de similitud de dos palabras, es decir dado lo que busca una palabra i (q_i) y la descripción de otra palabra j (k_j), como de bien casan j e i. Ese es el resultado de esa multiplicación, que luego normalizamos dividiéndolo por $\sqrt{d_k}$. Recordemos que nosotros lo que estamos trabajando es con matrices y no con vectores individuales. Por lo tanto, lo que hará el modelo será multiplicar el vector **q** de una palabra por cada uno de los vectores **k** del resto, obteniendo así lo que se denomina vector de atención (ver imagen 4.5). En este vector podemos ver como de importante es cada palabra de la secuencia para la palabra original (de la que hemos usado el vector **q**). Este proceso se repetirá para todos los **q**, generando así la denominada matriz de atención (ver imagen 4.6). Finalmente al multiplicar la matriz de atención por la matriz de V lo que estamos es relacionando esas atenciones calculadas con la representación de la palabras originales. Es decir, por ejemplo multiplicaremos cada una de las componentes del vector de atención de la palabra Antonio por el vector al que hace referencia dicha componente (cada vector **v** es una fila la matriz **V**) obteniendo así el vector OUTPUT de la palabra Antonio (ver imagen 4.7) .

Con esto tenemos ya la salida del bloque *Multi-Head Attention*, que si recordamos bien se sumara al residuo y se normalizara antes de pasar al siguiente bloque, que es el bloque *Feed Forward*. Este bloque no es mas que una red neuronal normal. Se puede observar que también se usa el mismo mecanismo de conexiones residuales, esto sera algo que veremos en prácticamente todos los bloques de la red.

Descripción del decoder

Una vez hemos terminado de describir el encoder, toca pasar al decoder. Como se puede ver en la imagen 4.2 el decoder recibe como input el etiquetado (en nuestro caso seria la transcripción) pero desplazado a la derecha. Esto, junto con el enmascarado de la primera *Multi-Head Attention*, fuerza a que las predicciones no se hagan solo en función de las palabras anteriores como haría un modelo

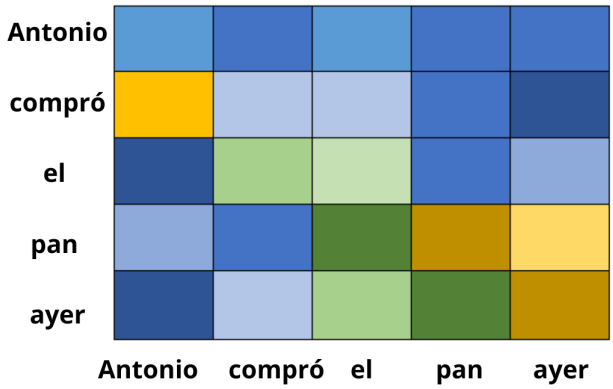


Figura 4.6: Representación gráfica de una matriz de atención

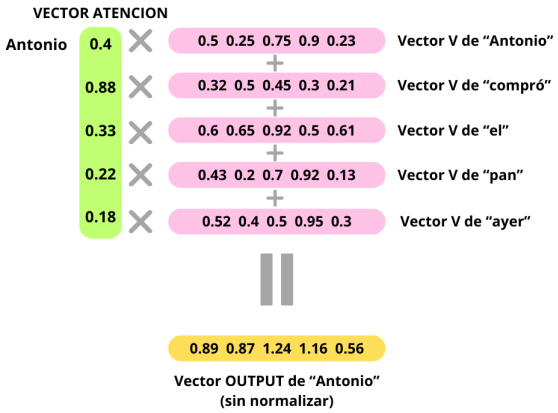


Figura 4.7: Descripción del proceso de obtención del vector OUTPUT para la palabra Antonio

del lenguaje, sino que se tengan en cuenta las posteriores también. La *Masked Multi-Head Attention* tiene un comportamiento idéntico al explicado en el encoder (ver sección 4.1.1) con la salvedad de que las posiciones futuras a la palabra que se esta procesando se enmascaran. De esta manera al aplicar el desplazamiento las ultimas palabras pasan a ser las primeras, y gracias al enmascaramiento las ultimas palabras (que realmente son las primeras de la frase original) se ocultan forzando a que las predicciones también se basen en las posiciones posteriores (ver imagen 4.8).

A continuación el resultado del bloque anterior se pasa a un nuevo bloque *Multi-Head Attention* pero donde en vez de recibir tres veces el mismo embedding, se va a recibir dos veces el embedding generado por el encoder y una vez el generado por el bloque *Masked Multi-Head Attention* del propio decoder. El funcionamiento de este bloque es idéntico al del encoder, al igual que el del bloque *Feed Forward* que le sigue.

Descripción del modelo de Whisper

Una vez hemos entendido como funciona un Transformer, vamos a ver en que se diferencia Whisper de ello. En realidad, la diferencia entre un Transformer estándar y Whisper es anecdótica. Como se puede ver en la imagen 4.9 la estructura de Whisper es idéntica a la del Transformer con la salvedad de que nos muestra como realiza la conversión de audio a embedding, la cual la realiza con una doble

Figura 4.8: Descripción del proceso de desplazamiento y enmascaramiento

red convolucional [10] y una función de activación GELU. Pero la estructura es la misma, la principal diferenciación de Whisper reside en como se ha entrenado. En el siguiente apartado ahondaremos más en este tema.

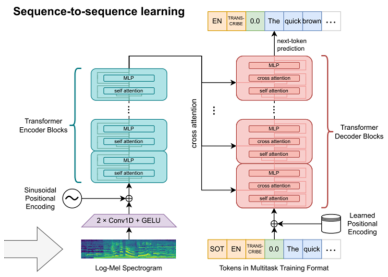


Figura 4.9: Representación de las estructura de Whisper.
Imagen obtenida del artículo [35].

Descripción del entrenamiento

Como ya se ha mencionado anteriormente, Whisper ha sido sometido a un proceso de aprendizaje supervisado, cosa no muy habitual en el ecosistema. Ha sido entrenado con 680.000 horas de audio etiquetado. La naturaleza de dichos audios es muy diversa, incluyendo audios en distintos ambientes, con distintos instrumentos de grabación, con distintos hablantes e incluso en distintos idiomas (96).

En el entrenamiento de Whisper se le da mucha importancia a la calidad de las transcripciones, y es por ello, que se aplican una serie de preprocesamientos a los datasets para quedarse solo con las mejores transcripciones:

- La primera medida consiste en implementar una serie de heurísticas para eliminar transcripciones no humanas (es decir, creadas por otro modelo ASR). Esto se realiza porque se ha demostrado que mezclar transcripciones humanas y no humanas en el entrenamiento desemboca en un mal rendimiento del modelo, sobre todo en tareas de traducción [11].

- Se usa un detector del lenguaje para corroborar que el idioma del audio y el de la transcripción es el mismo. Sino coincide se elimina dicha muestra, salvo si la transcripción está en inglés, en ese caso la muestra se almacena en otro directorio de entrenamiento que se usara para entrenar al modelo también en la traducción **IDIOMA EXTRANJERO – INGLES**. Es aquí donde nace la característica de multitarea que tiene Whisper.
- Los audios se parten en segmentos de 30 segundos (si es posible) con su correspondiente transcripción. Así se evita entrenar al modelo con audios muy largos.
- Para detectar y eliminar las transcripciones de baja calidad se utiliza un modelo inicial ya entrenado para clasificar las muestras según una métrica que engloba el error cometido y el tamaño de la muestra. Posteriormente se eliminan las muestras con mala transcripción manualmente.
- Se aplica deduplicacion para evitar la duplicidad de datos entre en conjunto de entrenamiento y el de validación. La deduplicacion es un proceso para eliminar copias de datos o de información.
- Se eliminan todas las muestras que incluyan el nombre del hablante, evitando así que este atributo sea algo que el modelo intente predecir.

Destacar que durante el entrenamiento de Whisper se probó con modelos de distinto tamaño para así poder estudiar su capacidad de escalado.

Model	Layers	Width	Heads	Parameters
Tiny	4	384	6	39M
Base	6	512	8	74M
Small	12	768	12	244M
Medium	24	1024	16	769M
Large	32	1280	20	1550M

Figura 4.10: Diferentes modelos de Whisper y su tamaño.
Imagen obtenida del artículo [35].

Los modelos que nosotros vamos a utilizar son una segunda y tercera versión del modelo Large que aparece en la figura superior. El tamaño y la estructura es igual que la versión Large normal, la diferencia reside en que el Large-V2 se ha entrenado con casi el triple (2.5) de epochs y Large-V3 ha sido entrenado con 1 millón de horas de audio débilmente etiquetado y 4 millones de horas de audio pseudo-etiquetado, todo generado usando el modelo Large-V2.

4.1.2 Corpora

FLEURS

El corpus FLEURS [6] es un corpus multi-lingüe de habla típica. El corpus se generó haciendo que 3 hablantes nativos de cada idioma leyesen las particiones de *train* y *validation* del corpus FLoRes-101 [12], que contienen 2009 frases extraídas de diferentes tópicos y dominios de artículos de Wikipedia. De esta forma se genera un dataset donde las particiones *train*, *validation*, *test* contienen 4527, 450, 1050 frases respectivamente, dando lugar así a 12 horas de audio por idioma. El corpus contiene 101 idiomas, entre los cuales se encuentra el español.

Este corpus le hemos obtenido de la página HuggingFace, más concretamente del siguiente enlace: https://huggingface.co/datasets/google/fleurs. En esta página podemos encontrar el corpus en su totalidad, con una partición para cada uno de los 101 idiomas.

VoxPopuli

El corpus VoxPopuli [49] un gran corpus de habla multilingüe que incluye 400.000 horas de audios no etiquetados en 23 idiomas diferentes. Es uno de los corpus más grandes existentes para realizar

entrenamiento no supervisado o entrenamiento semi-supervisado. Por otro lado, este corpus también incluye una parte dedicada al entrenamiento supervisado, debido a que en el podemos encontrar 1.800 horas de audios etiquetados en 15 idiomas diferentes (el español es uno de ellos). Esta parte en español consta de 166 horas de audio etiquetado, de intervenciones de 305 hablantes diferentes. Las intervenciones que componen la totalidad de este corpus han sido recopiladas de diferentes sesiones del parlamento europeo.

Este corpus le hemos obtenido de la página HuggingFace, más concretamente del siguiente enlace: https://huggingface.co/datasets/facebook/voxpopuli. En esta pagina se puede encontrar la parte etiquetado del corpus en su totalidad, dividida en cada uno de los 15 idiomas que la componen.

PRAUTOCAL

El corpus PRAUTOCAL [9] es un corpus de hablantes de español con SD del norte/centro peninsular español que permite el análisis de aspectos específicos del habla de las personas con SD. También incluye grabaciones comparables de usuarios con desarrollo típico (DT) que sirven de referencia. El corpus se ha construido grabando interacciones con un videojuego para entrenar las competencias orales de las personas con síndrome de Down. El corpus se recopiló en seis campañas de grabación y contiene 90 locutores, con 4175 ficheros de audio distribuidos en 40 actividades diferentes asociadas al desarrollo de un videojuego educativo supervisado por terapeutas.

Este corpus ha sido obtenido directamente del grupo de investigación ECA-SIMM, grupo el cual es el propietario del corpus debido a que sus autores forman parte de él. Este corpus hasta el momento no se encuentra disponible de forma libre.

4.1.3 Fine-Tuning

El *fine-tuning*, o en castellano *ajuste fino*, en una técnica que sirve para transferir nuevo conocimiento, variando los valores de los pesos, a un modelo que ya había sido entrenado con anterioridad. El conocimiento base del modelo y el nuevo tienen que estar estrechamente relacionados, por lo que muchas veces se dice que el *fine-tuning* sirve para realizar una especialización de un modelo más general. Dicho *fine-tuning* se puede realizar de muchas formas, ya que no hace falta realizarlo sobre todo el modelo, sino que se puede hacer sobre ciertas capas solo. Este es el caso de uno de los experimentos que mostraremos en este documento, donde congelamos una parte del modelo (el extractor de características) y entrenamos solo otra (la capa predictiva).

Este método se utiliza comúnmente en aprendizaje supervisado, donde el nuevo conocimiento con el que vamos a entrenar al modelo se encuentra etiquetado, o en su defecto, pseudo-etiquetado. De esta forma el **fine-tuning** se realiza siguiendo los pasos de un entrenamiento normal, construyendo un conjunto de *train* para entrenar el modelo y otro de *test* para evaluar el resultado del entrenamiento.

Esta práctica es especialmente común en PLN (procesamiento del lenguaje natural) debido a la reciente popularización de los LLMs (Large Language Models), los cuales son idóneos para someterse a un **fine-tuning**, ya que son modelos cuyo conocimiento suele ser muy general, y por tanto, una especialización suele ser muy efectiva.

4.1.4 Técnicas de Aumento de Datos

La escasez de datos en el mundo del ASR de habla anómala es un problemas con el que hay que lidiar diariamente. Con el fin de combatir este hecho una de las técnicas mas utilizadas es el aumento datos [15, 31], mediante técnicas como Voice Conversion, modificación de audio, TTS... Este problema también nos acontece a nosotros en nuestro experimento, y es por eso que vamos a realizar una primera aproximación al aumento de datos aplicando técnicas de modificación de audio muy básicas. Se ha decidido utilizar estas técnicas debido a la fácil implementación que tienen, si obtenemos un buen

resultado de su utilización, entonces, en futuros experimentos aplicaremos técnicas mas avanzadas. A continuación vamos a describir las técnicas que vamos a emplear y como las vamos a implementar.

Variación de la velocidad

Esta técnica consiste en variar la velocidad del audio, así se obtiene un audio un poco distinto con respecto al original. De esta forma se puede simular cambios en el ritmo y en la velocidad del habla, generando así "nuevos hablantes artificiales". Estos nuevos hablantes pueden ayudar al modelo a aprender a trabajar con audios que tengan un tono o un ritmo al cual el modelo no esta acostumbrado. En la siguiente imagen se puede ver cual es la transformación que sufre un audio al aplicarle esta técnica:

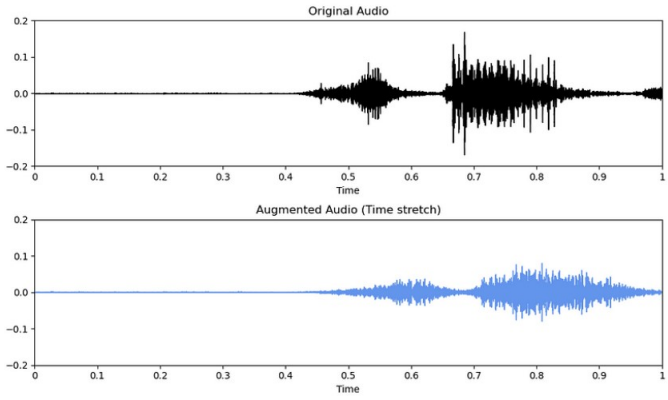


Figura 4.11: Variación de velocidad en un audio
Imagen obtenida de https://towardsdatascience.com/data-augmentation-techniques-for-audio-data-in-python-15505483c63c

Para aplicar esta técnica hemos usado la librería *wave* (https://docs.python.org/3/library/wave.html). Esta librería nos permite coger diferentes características de un audio, como por ejemplo, número de canales, número de frames, frame rate... Dado esta posibilidad lo que se ha realizado es lo siguiente: se obtiene el frame rate del audio y se multiplica por un valor, típicamente 0.8, 0.9, 1.1 u 1.2. De esta forma modificamos de la velocidad del audio generando tonos mas graves o agudos.

Introducción de ruido

Esta técnica consiste en añadir ruido al audio original. El ruido que se puede introducir en estos audios es de 2 tipos. El primero es el ruido blanco, cuya principal característica es que tiene una densidad espectral de potencia constante. Esto quiere decir que el audio contiene todas las frecuencias y todas ellas tienen la misma potencia. El segundo tipo es el ruido de color, este ya no tiene la densidad espectral de potencia constante, sino que dependiendo de la forma que tenga diremos que es ruido de un color o de otro. De esta forma podemos ayudar al modelo a aprender a trabajar con audios que tengan ruido de fondo o cuya calidad de grabación no sea del todo buena. En la siguiente imagen se puede ver cual es la transformación que sufre un audio al aplicarle esta técnica:

Para aplicar esta técnica hemos usado las librerías *librosa* (https://librosa.org/doc/latest/index.html), *numpy* (https://numpy.org/) y *colornoise* (https://github.com/felixpatzelt/colorednoise). *Librosa* al leer un audio lo convierte en un vector numérico, entonces para la introducción de ruido blanco lo que haremos es con la librería *numpy* generar un vector de igual longitud del audio y multiplicarlo por factor de ruido, que típicamente sera 0,005. Posteriormente solo tendremos que sumar ese vector al audio original y habríamos añadido ruido

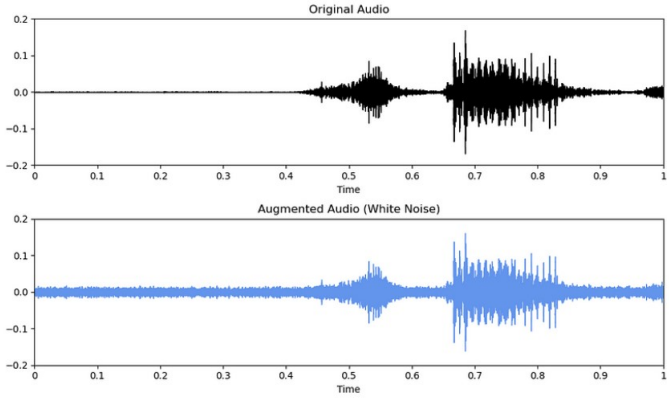


Figura 4.12: Introducción de ruido blanco en un audio
Imagen obtenida de https://towardsdatascience.com/data-augmentation-techniques-for-audio-data-in-python-15505483c63c

blanco. Para el ruido de color usaremos la librería *colornoise* que utiliza el algoritmo propuesto en [43] para crear el ruido de color.

Variación de tono

Esta técnica consiste en variar el tono (pitch) del audio original, dando así a nuevos audios con características diferentes que simulan ser de nuevos hablantes. Al igual que con la variación de la velocidad (ver sección 4.1.4) la introducción de nuevos hablantes en el dataset ayuda al modelo a generalizar. En la siguiente imagen se puede ver cual es la transformación que sufre un audio al aplicarle esta técnica:

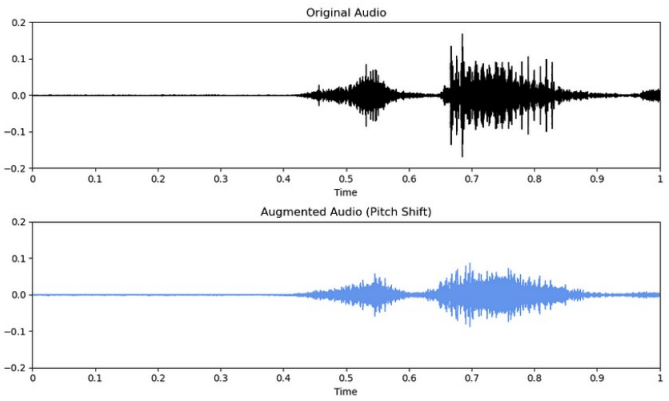


Figura 4.13: Variación del tono en un audio
Imagen obtenida de https://towardsdatascience.com/data-augmentation-techniques-for-audio-data-in-python-15505483c63c

Para aplicar esta técnica hemos usado las librerías *librosa* (https://librosa.org/doc/latest/index.html). Esta librería contiene una función que permite cambiar el tono de un audio, esta función se llama *librosa.effects.pitch_shift()* (https://librosa.org/doc/main/generated/librosa.effects.pitch_shift.html). Esta función te permite bajar/subir semitonos al audio en función del parámetro *n_steps*.

4.1.5 WER

El **Word Error Rate** es una metrica muy utilizada en la evaluación de sistemas de ASR. Como su propio nombre indica esta métrica se aplica a nivel de palabra, lo que quiere decir que su aplicación se basa en realizar comparaciones entre las diferentes palabras de dos frases, donde normalmente una corresponde a la predicción y la otra es la referencia.

El concepto de esta métrica es muy simple, se basa en calcular el numero mínimo de *inserciones*, *borrados* y *sustituciones* necesarios para transformar una frase (la predicción) en otra (la referencia). Esta medida se basa en la *distancia de Levenshtein* aplicada a nivel de palabra. Esto ultimo quiere decir que las comparaciones entre palabra no se harán letra a letra, sino que la comparación se hace con toda la frase. Por ejemplo, si en una frase hay una palabra menos que en la otra se contara como un borrado, si hay palabras extras se contaran como inserciones y si el numero de palabras es el mismo pero hay palabras que no coinciden se contaran como sustituciones.

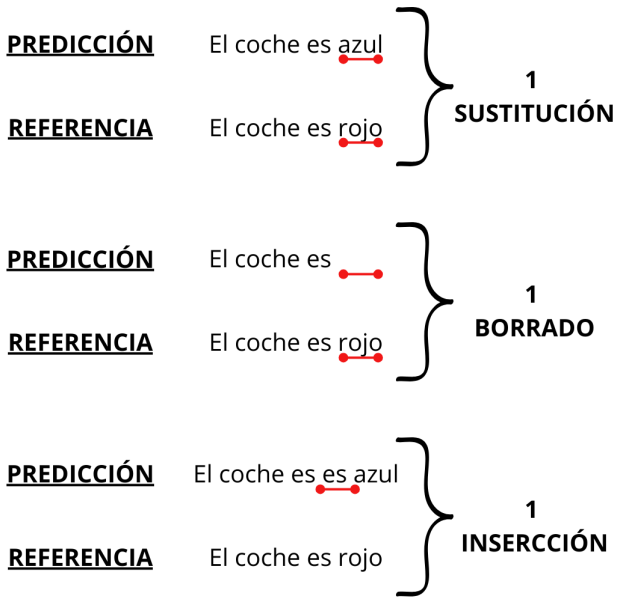


Figura 4.14: Posibles errores que conforman el WER

Una vez ya hemos calculado la *distancia de Levenshtein* y tenemos el total de *inserciones*, *borrados* y *sustituciones* podemos proceder a calcular el WER aplicando la siguiente fórmula:

$$WER = \frac{S + B + I}{N} \tag{4.1}$$

donde

- **S** es el número de sustituciones.
- **B** es el número de borrados.
- **I** es el número de inserciones.
- **N** es el número total de palabras.

Algo importante a destacar es que esta métrica no se calcula frase por frase, sino se calcula usando todas las frases evaluadas en su conjunto. Es decir, si evaluamos el modelo con 100 frases que tienen un total de 1000 palabras, el proceso consistirá en ir calculando la *distancia de Levenshtein* frase

por frase e ir acumulando el numero de errores. Finalmente cuando ya se hayan evaluado todas las frases se calculara el WER usando: el numero de sustituciones/borrados/inserciones acumulado de la evaluación de todas las frases. Esto supone ciertos problemas, ya hace que la métrica WER se vea muy influenciada por valores atípicos, es decir, por frases donde, por alguna razón, la predicción es nefasta. En la siguiente imagen se puede ver un ejemplo real con el que nos hemos encontrado durante la experimentación y que ejemplifica muy bien el ejemplo anterior.

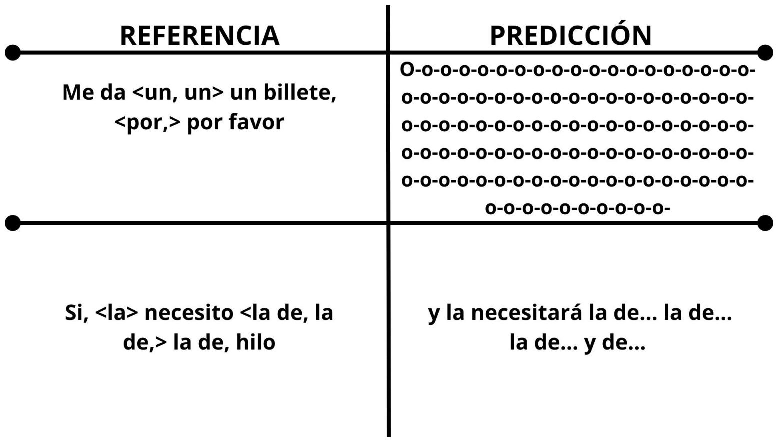


Figura 4.15: Predicciones nefastas que estropean el WER

4.1.6 BERT

Analizar el rendimiento de un modelo en base al número de palabras coincidentes entre la referencia y la predicción puede llegar a ser algo pobre. Pueden existir casos con WER alto pero debido solamente a fallos puntuales en ciertas palabras que no son del todo importantes para comprender el significado de la frase. Para solucionar este problema surge el BERT F1 Score, que ha sido ya utilizada para evaluar el rendimiento de sistemas ASR [44]. Esta métrica no se fija en la similitud ortográfica de las frases, sino en su similitud semántica.

BERTScore[52] es una métrica de evaluación automática para la generación de texto que calcula una puntuación de similitud para cada token en la oración candidata con cada token en la oración de referencia. Aprovecha las incorporaciones contextuales previamente entrenadas de los modelos BERT y relaciona palabras en oraciones candidatas y de referencia mediante similitud de coseno.

Dadas dos frases, la predicción y la referencia, usamos el modelo BERT [8] para generar un vector de *embbedings* por frase. Dicho vector contendrá en la posición i el embedding correspondiente a la palabra que se encuentra en la posición i de la frase. Una vez terminado todo el proceso, dispondremos de dos vectores de *embbedings*, uno para la frase referencia y uno para la frase predicción. Finalmente, cogemos ambos vectores y calculamos su similitud coseno posición por posición, es decir, calculamos la similitud de la primera palabra de cada frase, luego de la segunda... obteniendo al final lo que podríamos denominar un vector de similitud.

El BERTScore, esta compuesto por 3 métricas: **Bert-Precision**, **Bert-Recall** y **Bert-F1**, las cuales se calculan todas a partir de la similitud coseno. La forma de calcular la **Bert-Precision** es muy sencilla, primero buscamos para cada palabra del vector referencia, la palabra más similar del vector predicción. Como segundo paso sumamos dichas similitudes y calculamos su media. Si se mira con detenimiento es bastante similar al calculo de un precisión normal, donde primero se contabilizan los aciertos (en nuestro caso se contabilizan las similitudes) y luego se hace la media. Esto se hace debido a que lo que nos interesa es el nivel semántico de la frase, y por tanto, el orden de las palabras es indiferente siempre y cuando la frase final guarde el mismo significado.

En lo que respecta a **Bert-Recall**, es exactamente igual que el **Bert-Precision**, lo que pasa es que

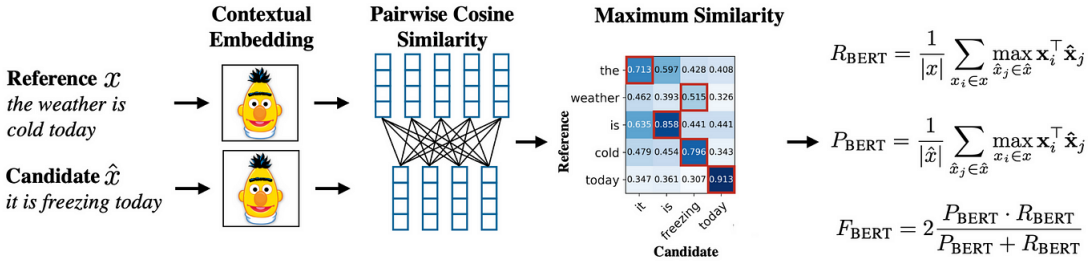


Figura 4.16: Resumen del calculo de la métrica BertScore.
Imagen obtenida del artículo [52].

$$P_{\text{BERT}} = \frac{1}{|\hat{x}|}\sum_{\hat{\mathbf{x}}_\mathbf{j} \in \hat{x}} \underbrace{\max_{\mathbf{x}_\mathbf{i} \in x} \overbrace{\mathbf{x}_\mathbf{i}^\top \hat{\mathbf{x}}_\mathbf{j}}^{\text{cosine similarity}}}_{\text{greedy matching}}$$

Figura 4.17: Formula Bert-Precision.
Imagen obtenida de https://docs.kolena.com/metrics/bertscore/

en vez de buscar la palabra del vector predicción mas similar a cada palabra del vector referencia, lo hacemos al revés, y buscamos la palabra mas similar del vector referencia para cada palabra del vector predicción. En este caso vemos que esta formula también es muy similar a la formula clásica del *recall*.

$$R_{\text{BERT}} = \frac{1}{|x|}\sum_{\mathbf{x}_\mathbf{i} \in x} \underbrace{\max_{\hat{\mathbf{x}}_\mathbf{j} \in \hat{x}} \overbrace{\mathbf{x}_\mathbf{i}^\top \hat{\mathbf{x}}_\mathbf{j}}^{\text{cosine similarity}}}_{\text{greedy matching}}$$

Figura 4.18: Formula Bert-Precision.
Imagen obtenida de https://docs.kolena.com/metrics/bertscore/

Finalmente, nos faltaría la ultima componente del BertScore, el **Bert-F1**. En este caso es la misma formula que la F1-Score, pero sustituyendo la precisión y el *recall*, por la **Bert-Precision** y **Bert-Recall**.

La **Bert-F1** es la métrica que nosotros vamos a utilizar para evaluar nuestros modelos, como bien recomienda el paper original.

4.2 Evaluador Automático del Habla

Este segundo camino se va a centrar en realizar un primer acercamiento a la tarea de evaluación de la pronunciación, en este caso centrándonos en la evaluación del habla Down. Los principales conceptos referentes a esta sección son:

$$F_{\text{BERT}} = 2 \times \frac{P_{\text{BERT}} \times R_{\text{BERT}}}{P_{\text{BERT}} + R_{\text{BERT}}}$$

Figura 4.19: Formula Bert-F1.
Imagen obtenida de https://docs.kolena.com/metrics/bertscore/

4.2.1 Wav2Vec

El modelo Wav2Vec [2] es un modelo acústico, cercano al estado del arte, cuya arquitectura tiene una base muy similar al modelo ya explicado Whisper (ver sección 4.1.1), dado que se basa en la combinación de CNNs y Transformers. La gran mayoría de los modelos con mejores rendimientos hoy en día siguen una arquitectura similar. La explicación de como funciona la arquitectura transformer se puede ver en la sección 4.1.1

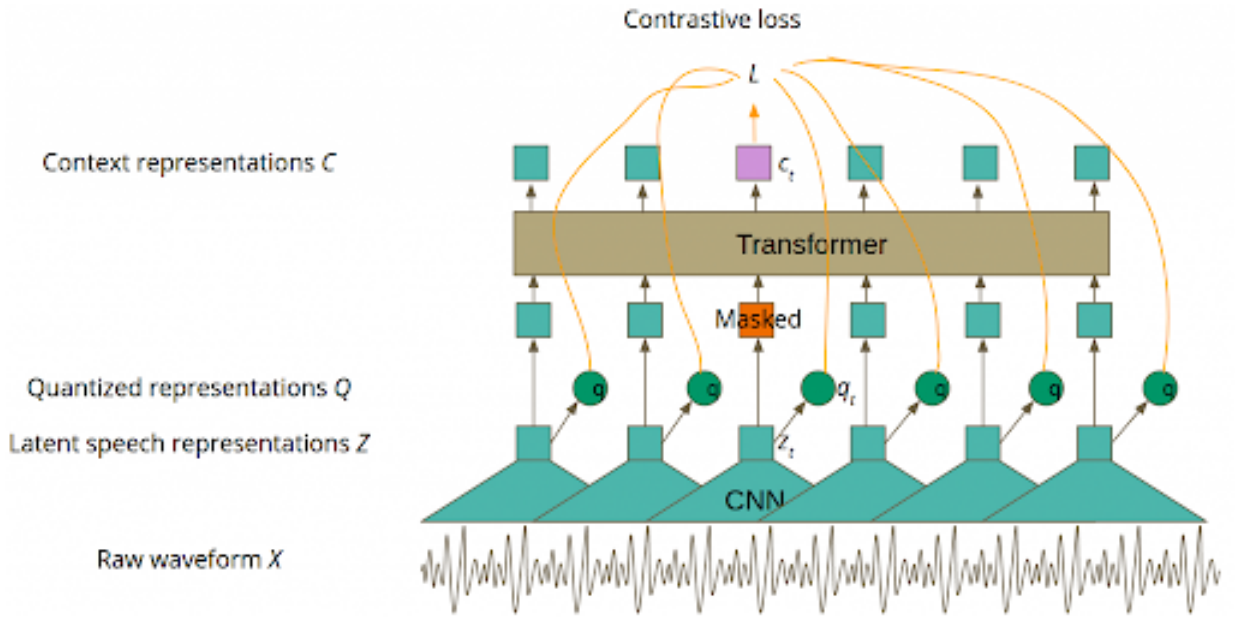


Figura 4.20: Arquitectura Wav2Vec.
Imagen obtenida de https://blog.kairosds.com/reconocimiento-del-habla-con-hugging-face/

La diferenciación de un modelo como Wav2Vec en relación al resto, reside en como se realiza su entrenamiento, que pretende asemejarse más a como un recién nacido aprendería a hablar. Se basa en lo que se denomina aprendizaje auto-supervisado, el cual se puede dividir en dos parte muy diferenciadas. Primeramente, el modelo comienza su entrenamiento en modo auto-supervisado, donde se le suministran datos sin etiquetar y el modelo tiene la tarea de crear la mejor representación posible de esos datos. Esta parte se puede comparar con la etapa en la que un bebe humana empieza a escuchar a sus padre y personas cercanas hablar y él, sin entender lo que están diciendo empieza a realizar un esquema mental de los sonidos que esta escuchando.

La segunda parte consiste en realizar un segundo entrenamiento, denominados ajuste de precisión supervisado, en el cual se le dan al modelo nuevos datos, pero esta vez ya etiquetados. Este paso se puede comparar con la etapa en la que los padres intentan enseñar a su bebe, y donde el bebe ya no solo recibe estímulos en forma de audio sin sentido, sino que estos nuevos estímulos que recibe llevan con ellos un segundo estimulo que ayuda al bebe a relacionar el sonido con su significado. De esta

forma se va ajustando su mapa mental a la realidad.

4.2.2 Corpora

CommonPhone

El CommoPhone [23] es un subcorpus del popular corpus CommonVoice que ha sido creado por la corporación Mozilla. El CommonPhone contiene un total de 116.5 horas de ejemplos de habla de 11.246 hablantes distintos en 6 idiomas diferentes. La peculiaridad de este corpus reside en que esta compuesto por ejemplos de pronunciación de fonos individuales, mientras que los corpus usados habitualmente en ASR, como el CommonVoice, se componen de ejemplos de locuciones de frases.

Language	Speakers	Hours
	train / dev / test	train / dev / test
English	4716 / 771 / 774	14.1 / 2.3 / 2.3
French	796 / 138 / 135	13.6 / 2.3 / 2.2
German	1176 / 202 / 206	14.5 / 2.5 / 2.6
Italian	1031 / 176 / 178	14.6 / 2.5 / 2.5
Spanish	508 / 88 / 91	16.5 / 3.0 / 3.1
Russian	190 / 34 / 36	12.7 / 2.6 / 2.8
Total	8417 / 1409 / 1420	85.8 / 15.2 / 15.5

Figura 4.21: Distribución de las locuciones en los diferentes idiomas que componen el CommonPhone. Imagen obtenida de [23].

Las locuciones de los fonos que aparecen en el corpus han sido extraídas de la segmentación fonética de diferentes locuciones presentes en el CommonVoice. Dicha segmentación fue realizada usando BAS Web Services (https://clarin.phonetik.uni-muenchen.de/BASWebServices/interface/Pipeline. Con el fin de poder garantizar una unidad fonética entre los diferentes idiomas el CommonPhone usa el Alfabeto Internacional Fonético (IPA), debido a que dicho alfabeto resulta ser muy útil para el trabajo fonético multilingüe. El IPA esta compuesto por 101 símbolos fonéticos, entre los cuales podemos encontrar un símbolo especifico para el silencio.

Este corpus fue obtenido de la web de Zenodo, mas concretamente del siguiente enlace: https://zenodo.org/records/5846137.

4.2.3 GoP

El *Goodness of Pronunciation* es una métrica para medir la calidad de la pronunciación en base a una habla modélica. Con el fin de establecer esta correlación entre dos hablas distintas y poder cuantificar cual es mejor se propuso en un trabajo la siguiente fórmula:

$$GoP(p) = \frac{1}{NF(p)} \cdot \left|\log \frac{P(O^{(p)}|p)}{\max_{q \in Q} P(O^{(q)}|q)}\right|$$

Para el calculo del GoP se usa un modelo predictivo, es decir, un modelo que dado un input (una locución de un fono) da una probabilidad para cada uno de los fonos pertenecientes al diccionario de fonos. Por ejemplo, si el IPA tiene 101 simbolos y es el diccionario que utilizamos, para una locucion de un fono X el modelo debe dar una probabilidad para cada uno de los 101 fonos. Obviamente la suma

de las probabilidades de los 101 fonos debe dar 1. Por lo tanto, se usa el modelo como un comparador, haciéndole decidir como de parecido es el fono que ha escuchado en relación al modelo que el tiene de cada uno de los 101 fonos, en este caso.

Por otro lado, se puede apreciar que para poder aplicar esta fórmula necesitamos tener la locuciones etiquetadas, es decir, tenemos que saber a que fono corresponde cada sonido, para poder realizar la futura comparación usando el GoP.

Una vez aclarado esto obtener el valor GoP de una locución se vuelve trivial. EL numerador $P(O^{(p)}|p)$ no es más que, sabiendo que el fono pronunciado es p, la probabilidad que el modelo predictor da para el fono p. Por otro lado, el denominador $\text{máx}_{q \in Q} P(O^{(q)}|q)$ no es más que el fono que el modelo ha predecido como el más probable de todo el conjunto de fonos Q, incluyendo el fono p. De esta forma el cociente no es más que, la probabilidad de que el fono de la locución sea el que se supone que se debería haber pronunciado, partido por el fono que modelo reconoce como el más probable. Así, si el fono más probable coincide con el fono etiquetado se obtiene el valor máximo que es 1 y si la diferencia entre el fono más probable y el etiquetado es máxima se obtiene el peor valor que es 0.

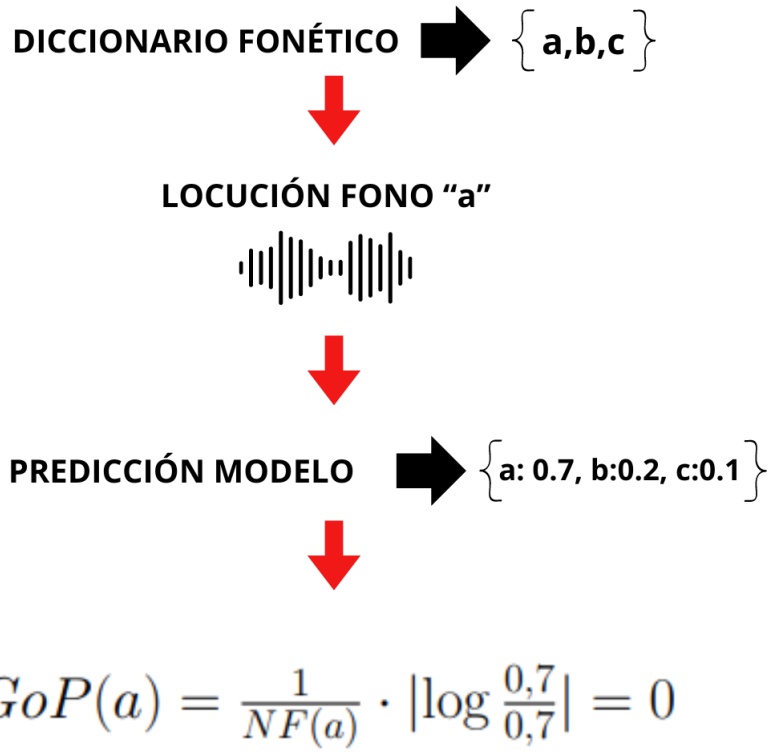


Figura 4.22: Ejemplo del Calculo del GoP

Como se puede observar en la fórmula se aplica el logaritmo a la comparación anteriormente explicada, que hace que la escala se invierta y el mejor valor pase a ser 0 y el peor pase a ser 1. Por lo tanto, al analizar un conjunto de fonos, que pueden constituir una palabra o una frase, cuanto mayor sea el valor del GoP peor habrá sido la pronunciación.

Finalmente nos falta por explicar un ultimo termino, $\frac{1}{NF(p)}$. $NF(p)$ corresponde al número de frames en el que se pronuncia el fono p. De esta forma se podria decir que lo que hacemos es calcular la media del GoP de un fono p. Si no hiciéramos esto estaríamos teniendo en cuenta de la misma forma fonos que son muy cortos de pronunciar (y presumiblemente más sencillos) con fonos más largos, donde la probabilidad de error se presupone mayor. Por lo tanto, al aplicar este termino realizamos

una especie de normalización de todos los fonos con el fin de que los valores de GoP sean justos y comparables.

Por ultimo, destacar que la fórmula anteriormente explicada es la fórmula que se propuso originalmente. Con el tiempo y los cambios de paradigmas se han ido desarrollando muchas variantes y formulas distintas de GoP, como se verá a lo largo de este trabajo. Pero en definitiva todas ellas siguen el mismo concepto en el que se basa la fórmula original aquí explicada.

Capítulo 5

EXP 1: ASR para Habla Down

5.1 Soluciones Existentes

El reconocimiento automático del habla para el habla patológica es un área de investigación en crecimiento. Varios estudios han resaltado la importancia del ASR para ayudar a individuos con trastornos del habla [36, 22, 39]. Uno de los trastornos del habla más estudiados es la disartria [1, 18, 20, 40, 3]. Los estudios han explorado el uso de sistemas ASR para reconocer patrones de habla disártrica, con un enfoque centrado en el análisis de los factores que afectan el rendimiento del sistema. El estado actual del ASR para el habla patológica, en particular la disartria, está avanzando rápidamente, ofreciendo soluciones innovadoras para ayudar a individuos con trastornos del habla en sus procesos de evaluación y terapia. La integración de la tecnología ASR tiene un gran potencial para mejorar las vidas de aquellos afectados por condiciones de habla patológica.

El habla de los individuos con SD no es necesariamente disártrica [24], pero en general no es un habla típica, debido a problemas relacionados con el tono muscular bajo, el tamaño grande de la lengua y las frecuentes infecciones del oído. Estos individuos pueden experimentar habitualmente retrasos en el habla y dificultades en la articulación cabe destacar que cada persona con SD es única, y los problemas de habla pueden variar ampliamente entre individuos [24].

La disponibilidad de corpus de entrenamiento en ocasiones es un obstáculo para la investigación y el desarrollo de estos sistemas, porque recopilar este tipo de corpus es un proceso costoso al que hay que dedicar gran cantidad de recursos. En el marco del proyecto Euphonia[1], desarrollado por Google Research, se ha recopilado un corpus de habla de personas con trastornos del habla en inglés. Se trata del corpus más grande de este tipo de hablantes grabado hasta la fecha, puesto que han participado más de 1000 personas y se han grabado más de 1 millón de locuciones, lo que supone más de 1300 horas. El corpus incluye grabaciones de 105 personas con SD, lo que supone el 18,1 % del corpus [26]. Empleando este corpus se han realizado diversos experimentos. Se han empleado diferentes técnicas de deep learning para clasificar la inteligibilidad del habla de 661 locutores con diversas patologías, incluyendo SD [48]. Por otro lado, para mejorar el rendimiento de los sistemas de reconocimiento automático del habla en hablantes con trastornos del habla, se ha propuesto la personalización de modelos [45, 13], como vía para evitar la degradación significativa del rendimiento en habla patológica que se produce en los actuales sistemas de reconocimiento automático de habla.

Otro problema que suele afectar a este tipo de habla son las disfluencias y las variaciones en la pronunciación del habla, que pueden degradar gravemente el rendimiento del reconocimiento de habla, ya que los sistemas actuales de ASR se entrenan principalmente con habla fluida de hablantes típicos.

[1] https://sites.research.google/euphonia/about/ (visitado 12-03-2024).

Un enfoque sencillo para mejorar el rendimiento es ajustar los parámetros de decodificación en un sistema de reconocimiento de habla existente, lo que puede mejorar la tasa de errores de palabras (WER) para personas con trastornos de fluidez [29].

5.2 Adaptación de los datos

Los corpora que habitualmente se usan en tareas de reconocimiento de Voz-Texto suelen tener una estructura bastante consolidada. Esta estructura consiste en una serie de audios y un fichero (habitualmente *.csv*) que asocia cada audio con su transcripción. La totalidad de los corpora que vamos a utilizar siguen esta estructura, por lo tanto, no es necesaria ninguna adaptación.

El problema en nuestro caso se encuentra, tanto en el formato de las transcripciones de los distintos corpora, como en el formato de las predicciones del modelo Whisper. Dichos formatos no coinciden en todos los casos, debido a que, por ejemplo, algunos corpus no contienen signos de puntuación y el modelo Whisper si que los genera cuando realiza sus predicciones (todos los aspectos referentes a los formatos de los corpora se pueden ver en [49, 9] y los referentes al modelo Whisper en [35]). Por lo tanto, se debe realizar una adaptación de cada elemento para obtener así un formato único y que los resultados sean validos. A continuación se van a explicar las adaptaciones que se les a realizado a cada elemento:

PRAUTOCAL

El etiquetado del corpus PRAUTOCAL Down contiene ciertas marcas indicativas de disfluencias que deben ser eliminadas para obtener una referencia limpia y precisa. A continuación se va a explicar el significado de cada marca, y el tratamiento que se ha aplicado a cada una de ellas:

- Términos que han sido marcados entre los símbolos <y >indican que se ha producido una disfluencia. En este caso, los símbolos se elimina pero no el contenido de su interior.

FD11041D0310R01

Hola, **<t><t>**tienes lupa, **<que><que>**queria comprar una

Figura 5.1: Locución con símbolos de repetición

- Palabras precedidas por el símbolo #, lo cual indica que dicha palabra es un filler, es decir, una palabra de relleno. El procedimiento aquí es el mismo que en el caso anterior, se elimina el símbolo pero se mantiene el texto que lo sigue.

FD22024D2200R02

<#br> buenos dias, que triste, estar tan solo

Figura 5.2: Locución con símbolo de filler

- Signos de puntuación (puntos y comas) que pierden su significado ortográfico y pasan a referirse a pausas que el hablante a hecho durante la locución. Finalmente debido a la confusión que pueden originar este uso atípico de estos símbolos se han decidido eliminar de las transcripciones.

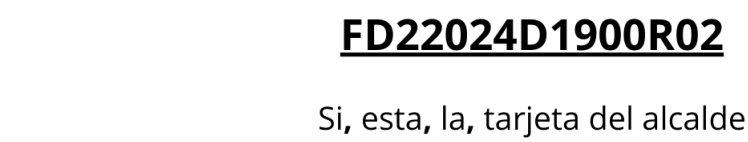


Figura 5.3: Locución con símbolos de pausas

Las marcas indicativas de disfluencias que se han eliminado, se han anotado en un fichero *.csv* externo, en el cual se puede ver el numero total de apariciones de cada tipo de marca por frase. Esta información ha sido almacenada con la esperanza de poder usarla en futuros estudios.

Además de los anteriormente citados, también se ha eliminado cualquier otro signo de puntuación, todas las tildes (debido a que el corpus no estaba correctamente etiquetado en ese sentido) y todos los signos ortográficos, como guiones, corchetes..., que pudiesen aparecer.

También se han eliminado 67 ficheros de audio del corpus PRAUTOCAL Down que se determino que contienen pseudo-habla (ruidos sin sentido). Por ultimo, se ha decidido añadir un segundo de silencio al comienzo y al de cada fichero de audio del corpus, lo que se comprobó que mejoraba el rendimiento de los reconocedores en esas frases, como ya se detecto en el trabajo de [31].

Por otra parte, debido principalmente al tamaño reducido del corpus PRAUTOCAL Down, el cual contiene tan solo 2 horas de audio Down, se ha optado por un esquema de partición 60/40: 60 % de los ficheros de audio para *train* y 40 % para *test*, obviando la partición de *validation*. Esta partición no es para nada la estándar, la cual suele ser un 80/10/10, para las particiones *train*, *test* y *validation* respectivamente.

La participación *validation* se ha omitido debido a que no era necesaria en nuestro experimento. La participación de *validation* se usa a la hora de realizar el ajuste de hiper-parámetros de un modelo, para realizar una evaluación tras cada *epoch* y así, tanto controlar el *overfitting*, como ver que conjuntos de hiper-parámetros son más optimos para el modelo. En nuestro caso, el ajuste de hiper-parámetros se realiza con el corpus FLEURS, debido al reducido tamaño de PRAUTOCAL, y por lo tanto, no nos es necesario disponer una partición de *validation*.

En cuanto a la participación *train*, el porcentaje de muestras se ha elegido teniendo en cuenta que se aplicaran técnicas de aumento de datos (ver sección 4.1.4) sobre el corpus, que harán que el conjunto de entrenamiento crezca considerablemente, lo que podrıa generar que la partición *test* tuviera un tamaño muy reducido en comparación a la de *train* si se optase por otros esquemas mas tradicionales.

Siguiendo el esquema anterior de partición, se han realizado dos divisiones diferentes, que se describen a continuación:

La **división por actividad** consiste en separar en *train* y *test* las diferentes actividades que contiene el corpus PRAUTOCAL. Todas las locuciones del 60 % de las 40 actividades (ver Imagen 5.4) diferentes que contiene el corpus irán al conjunto *train* y el 40 % restante al de *test*. Entendemos por actividad un tipo de frase pronunciada por cualquier locutor. Para cada actividad y locutor existen diferentes locuciones. Realizando esta división conseguimos que ninguna partición tenga frases en común. De esta manera, una vez realizado el *fine-tuning*, al evaluar el modelo con el conjunto *test*, se enfrentara a frases nunca vistas, lo que nos permitirá observar cual es su verdadero aprendizaje de las caracterısticas propias del habla Down, al ser un modelo independiente de locutor y de tarea.

La **division aleatoria** consiste en distribuir directamente las locuciones entre *train* y *test* siguiendo los porcentajes 60-40 %, sin tener en cuenta la actividad. Con esta división es muy probable que algunas frases coincidan en ambos conjuntos, y por tanto, es posible que el modelo no solo aprenda las caracterısticas intrınsecas del habla Down, sino también la pronunciación de esas frases particulares, resultando solo independiente de locutor pero no de tarea.

Activity code	Expected utterance	Prosodic Function			Language Function			Prod. Mode	
		Modality	Boundaries	Prominence	Function	Politeness	Emotion	First attempt	Other attempt
D0120	¡Hasta luego, tío Pau! (*See you later, Uncle Paul*)	E	IF		S	F		R	R
D0211	¿Dónde hay libros de historia, por favor? (*Where are there history books, please*)	QD	IF		HS	C		E	I
D0220	¡Muchas gracias Juan! (*Thank you very much Juan*)	D	IF		S	T		E	I
D0310	Hola, ¿tienen lupas? Quería comprar una. (*Hello, do you have magnifying glasses. I wanted to buy one.*)	DQD	FFF		SHR	G		E	I
D0320	Sí, la necesito. ¿Cuánto vale? (*Yes, I need it. How much does it cost?*)	DQ	IFF		NH			R	R
D0420	Hola tío Pau. Ya vuelvo a casa. (*Hello, Uncle Pau. I'm coming home now.*)	DD	IFF		SR	G		E	I
D0430	Sí, esa es. ¡Hasta luego! (*Yes, that's it. See you later!*)	DE	IFF		RS	F		R	R
D0510	¡Hola, tío Pau! ¿Sabes dónde vive la señora Luna? (*Hello Uncle Pau¡Do you know where Mrs. Luna lives?*)	EQ	IFF		SH	G		R	R
D0820	Si no tardo, sí. Por favor. (*If it doesn't take long, YES please.*)	DD	IFF	W	OS	C		R	R
D0910	Hola. Necesito una escalera. (*Hi, I need a ladder.*)	D	IF		SN	G		E	I
D0930	No, no. La de CUERDA, por favor. (*no, no, the ROPE one please*)	D	IIF	W	OS	C		E	I
D0940	No, eso es todo. Gracias. (*No, that's all. Thank's*)	DD	IFF		RS	T		R	R
D1110	Buenos días, Señora Molina. ¿Está Pedro en casa? . (*Good morning, Miss Molina. Is Pedro at home?*)	DQ	IFF		SH	G		R	R
D1120	Buenos días, Señora Molina. (*Good morning, Miss Molina.*)	D	IF		S	G		I	I
D1130	¿Está Pedro en casa? (*Is Pedro at home?*)	Q	F		H			I	I
D1140	De acuerdo. Muchas gracias. (*All right. Thank you very much.*)	DD	FF		RS	T		R	R
D1210	¡Ey, Pedro! ¿Cómo estás? (*Hey Pedro, how are you?*)	EQ	FF		SH	G		R	R
D1220	Ojalá pudieras ¿Me dejas tu linterna? (*I wish you could, can I have your flashlight?*)	DQ	FF		PH		D	E	I
D1230	¿Me dejas tu linterna? (*Can I have your flashlight?*)	Q	F		H			E	
D1240	Me tengo que ir ya, Pedro (*I have to go now, Pedro.*)	D	IF		R			E	I
D1510	Soy quien busca la piedra mágica. Necesito ver al alcalde. (*I am the one who seeks the magic stone. I need to see the Mayor.*)	DD	FF		RN			R	R
D1511	Soy quien busca la piedra mágica. (*I am the one who seeks the magic stone.*)	D	F		R			R	R
D1512	Necesito ver al alcalde. (*I need to see the mayor*)	D	F		N			R	R
D1520	¿Sabe cómo ir a su casa, señor? (*Do you know how to get his house, sir?*)	Q	IF		H			R	R
D1610	No es necesario, señor alcalde, pero se lo agradezco. (*It is not necessary, Mr. Mayor, but I thank you.*)	D	IIF		S	T		R	R
D1710	Es que NO sé dónde está la piedra mágica. (*I just DON'T know where the magic stone is.*)	D	F	W	R			R	R
D1720	Tranquilo. Yo le ayudaré con mucho gusto. *It's okay. I will be happy to help you.*)	DD	FF		OR	R		R	R
D1730	Sí, claro. Aquí está. (*Yes, of course. Here it is.*)	DD	IFF		SR	C		R	R
D1740	Muchas gracias, señora alcaldesa. (*Thank you very much, Madam Mayor.*)	D	IF		S	T		R	R
D1900	Sí. El alcalde me ha dado esta tarjeta. (*Yes. The Mayor gave me this card.*)	DD	FF		RR			R	R
D2000	Seguro que sí. Buenas noches, Lolo. (*I'm sure you do. Good night, Lolo*)	DD	FIF		IS	F		R	R
D2200	Buenos días. ¡Qué triste estar solo en este bosque! (*Good Morning. How sad to be alone in this forest!*)	DE	FF		SP	G	S	R	R
D2400	SOlo la PIEdra de FUEgo Abre la PUERta del TEMplo. (*Only the fire stone opens the temple door.*)	D	F	S	I			R	R
D2500	¡Hala! ¡Esa puerta no estaba aquí antes! *Wow! That door wasn't here before!*	EE	FF		PR		O	R	R
D2600	No sé qué piedra elegir... (*I don't know which stone to choose ...*)	D	F		P	D		R	R
D2900	Ábrete PUERta y que BRIlle la PIEdra. (*Open up, door, and let the stone shine.*)	E	IF	S	O			R	R
D3000	free speech				RP		H	S	
D3100	Muchas gracias. Lo recordaré (*Thank you very much, I will remember it.*)	DD	FF		SR	T		E	I
D3400	Hola, ¿cómo estás?. (*Hi how are you doing?*)	DQ	IF		SS	G		E	I
D3700	¿Me da un billete, por favor?. (*Can I get a ticket, please?*)	QD	IF		OS	C		E	I

Figura 5.4: Listado de las actividades existentes en el corpus PRAUTOCAL. Imagen obtenida del artículo [9].

Corpora de habla típica

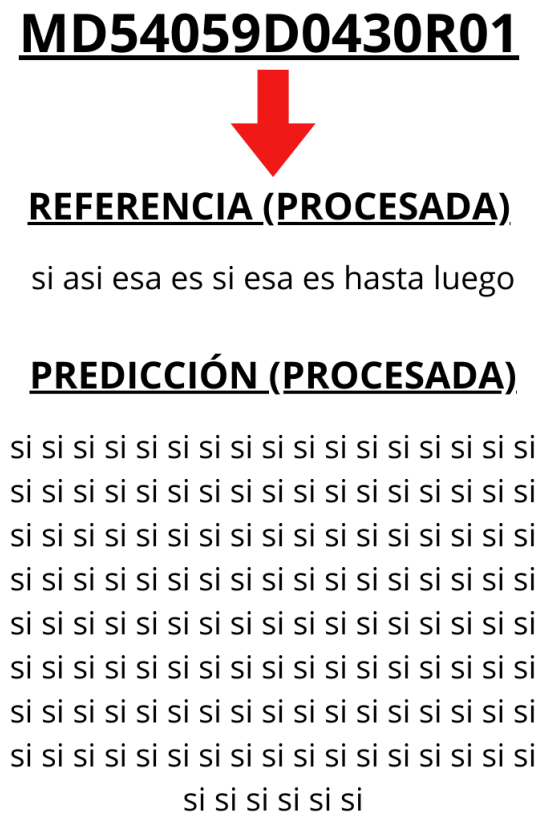
En lo que respecta a los corpus FLEURS y VoxPopuli, simplemente se les ha aplicado el preprocesamiento básico que el propio modelo whisper proporciona. Este procesamiento esta enfocado al trabajo plurilingüe y se basa en eliminar todo tipo de signos ortográficos y tildes, manteniendo la ñ. Para el caso especial de PRAUTOCAL Tıpico, se le ha aplicado la misma normalización que a PRAUTOCAL Down, debido a que, aunque carece de anotaciones, incluye numerosos errores en la ubicación de los signos de puntuación (interrogaciones y exclamaciones) y de las tildes.

Whisper

A las predicciones generadas por el modelo Whisper se les ha aplicado el preprocesamiento básico (mismo que a los corpus de habla típica) que el mismo ofrece. Con esto, finalmente conseguimos igualar el formato de las referencias de los diferentes corpus, con el formato de las predicciones del propio modelo. De esta forma cuando se quieran comparar las predicciones con las referencias, se podrá realizar una comparación justa y podremos ver así cual es el rendimiento real del modelo.

Por otro lado, como bien se cita en [35], el modelo whisper tiene un defecto que hace que, en ocasiones, el proceso de predicción entre en un bucle infinito (que solo termina al alcanzar el numero

máximo de caracteres permitidos en la generación) donde solo escribe una y otra vez la misma palabra, letra o conjunto de palabras. El propio artıculo documenta que se espera que dicho comportamiento desaparezca al hacer un fine-tuning del modelo. En toda la experimentación que hemos realizado nos hemos topado con este problema, tanto con el modelo base, como en la gran mayorıa de los fine-tuning que hemos realizado. El problema con estas frases es que normalmente son muy influyentes en el WER del modelo, debido a que generan una gran cantidad de inserciones que disparan su numero de errores, y por tanto, el WER general. En resumen, se ha procedido a eliminar las que denominaremos “frases en bucle”.


MD54059D0430R01
REFERENCIA (PROCESADA)
si asi esa es si esa es hasta luego
PREDICCIÓN (PROCESADA)
si si si

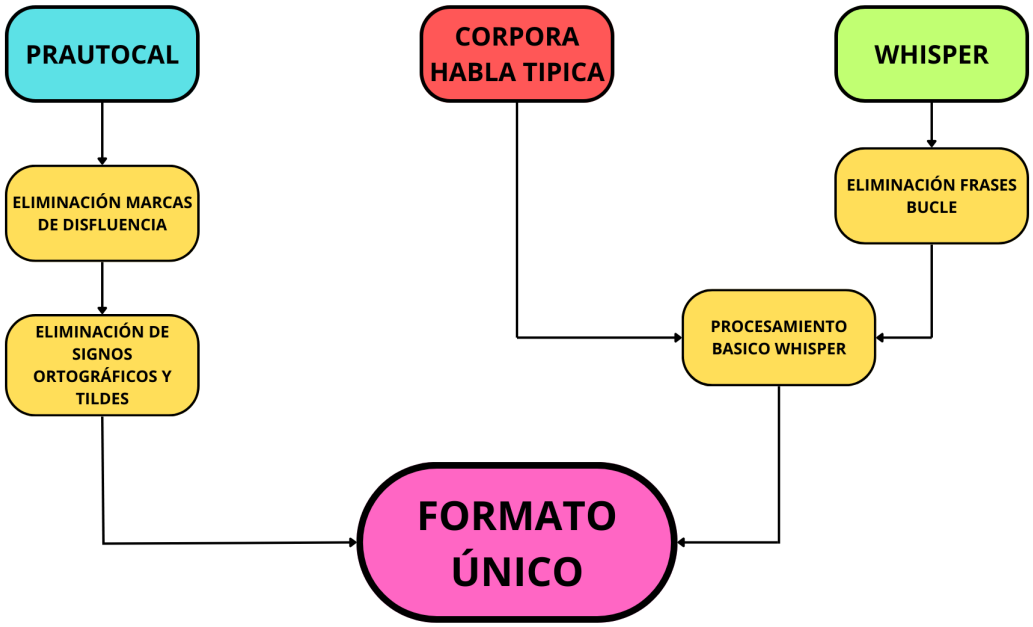


Figura 5.6: Resumen de las medidas de adaptación de los datos tomadas

5.3 Adaptación de los modelos

En el caso de la obtención del Reconocedor Voz-Texto para habla Down se ha decidido escoger el método de ajuste de hiper-parámetros de cara a la adaptación del modelo Whisper. Esta decisión se debe a que el modelo preentrenado escogido para esta tarea ya había sido entrenado para realizar la tarea de reconocimiento voz-texto (aunque de una forma muy general), por lo tanto, no era necesario modificar la arquitectura del modelo de ninguna forma, tan solo hacia falta preparar el modelo para la especialización posterior.

El ajuste de hiper-parámetros se ha realizado usando el corpus FLEURS. Lo normal ante estas situaciones es ajustar el modelo con el mismo corpus con el que luego se realizará el *fine-tuning*, pero el pequeño tamaño de PRAUTOCAL nos impide realizar una división correcta en tres particiones (*train*, *test* y *validation*), y por tanto, no podemos realizar el ajuste de forma apropiada. Si se hubiese seguido el estándar normal de repartición, que es 80-10-10, las particiones de *validation* y *test* serian muy pequeñas y no serian representativas, lo que quitaría valor a los resultados obtenidos. Otra opción que se podía barajar es, eliminar la partición *validation* y usar solo la partición *test*. Los resultados de esta opción tampoco serian validos, ya que al usar la misma partición, tanto para realizar el ajuste de hiper-parámetros, como para medir los resultados del modelo final, se podría incurrir en un proceso de *overfitting*.

La elección de FLEURS, sobre otros corpus de habla típica española, se debe principalmente a dos motivos: es un corpus bien conocido, usado en muchos artículos de esta índole [21, 35], y es un corpus pequeño, por lo que nos permite realizar muchas pruebas en un tiempo asumible, ya que el uso de corpus mayores requeriría de una capacidad de computo de la que no se dispone. Además, existen muchos artículos que arrojan resultados sobre este corpus aplicando la métrica WER, que es la métrica que vamos a emplear para decidir que conjunto de hiper-parámetros es mejor, lo cual nos facilita el trabajo y la obtención de conclusiones. La naturaleza y el funcionamiento de la métrica WER se puede ver explica en la sección 4.1.5.

El modelo Whisper tiene una elevada cantidad de hiper-parámetros que se pueden modificar, concretamente 109. Este valor tan elevado hace imposible optimizar todos, por lo que se ha escogido un subconjunto de todos ellos: *learning_rate*, *epochs*, *weight_decay* y formato de coma flotante. La elec-

ción de este conjunto se debe a que son los hiper-parámetros más comúnmente optimizados en trabajos de este estilo [33, 32].

Una vez se han seleccionado los hiper-parámetros podemos realizar la optimización de los mismos, siguiendo la metodología estándar: establecer un valor para cada hiper-parámetro a optimizar, posteriormente realizar un *fine-tuning* con dichos valores y con la partición *train* y evaluar el resultado con la partición *validation*. Para cada parámetro se han probado valores dentro de un intervalo inicialmente muy amplio que posteriormente se ha ido estrechando hasta alcanzar los valores óptimos.

En esta primera aproximación se escogía un valor, de dentro del intervalo, de cada uno de los hiper-parámetros, formando así un conjunto de valores diferentes en cada iteración. Debido a que los intervalos en un principio eran muy amplios no se podían probar todas las combinaciones, y por lo tanto, se uso el método de bayes para la elección de los valores. Esta fase nos sirvió para reducir los intervalos y hacernos una idea de donde se encontraba el valor óptimo de cada hiper-parámetros. Con la metodología anteriormente explicada se realizaron una gran cantidad de pruebas, como se puede ver a continuación:

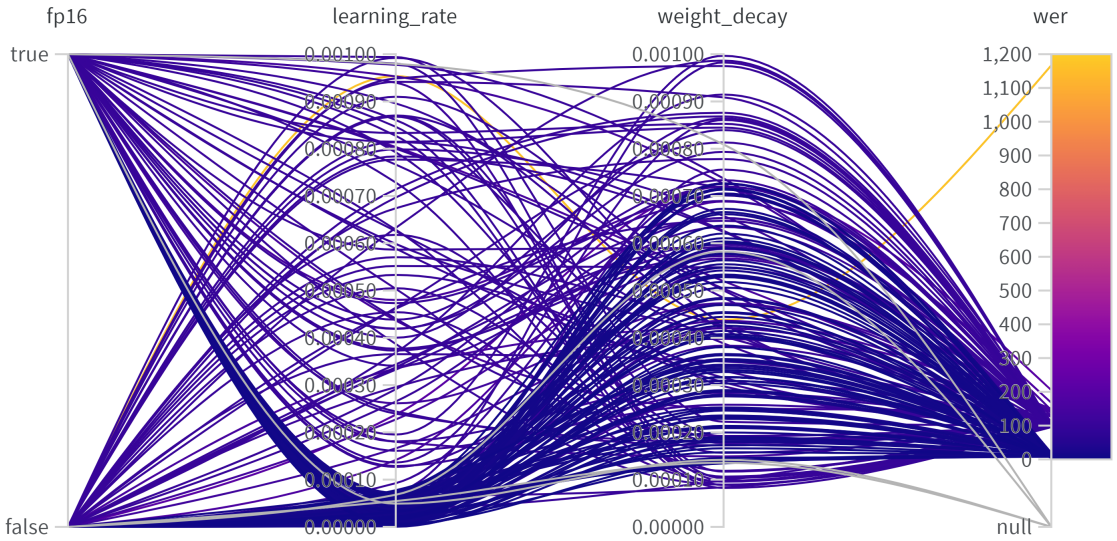


Figura 5.7: Representación de todos los experimentos realizados en la primera aproximación de la búsqueda de hiper-parámetros

Como siguiente, y último paso, se realizo un búsqueda en profundidad alrededor del valor que mejores resultados había dado en la etapa anterior para cada hiper-parámetro. En este caso, la optimización se hizo de manera individual, es decir, solo se variaba el valor de un hiper-parámetros y el resto se dejaban congelados. En este caso no se se escogió un rango de valores sino que se selecciono una lista de valores conformada por valores tanto a la izquierda como a la derecha del valor óptimo obtenido en la etapa anterior.

Finalmente, tras la finalización de las dos etapas anteriores se obtuvieron los siguientes resultados para los dos modelos que se han utilizado en esta sección (*whisper-large-v2* y *whisper-large-v3*):

5.4 Procedimiento Experimental

Como ya se explico en la sección 1.2 el objetivo de este desarrollo es conseguir un modelo ASR que consiga garantizar un rendimiento usable en la tarea de reconocimiento de habla Down castellana.

MODELO	LEARNING_RATE	WEIGHT_DECAY	EPOCHS
Whisper-Large-V2	2.155 x 10^{-5}	3.168 x 10^{-4}	1
Whisper-Large-V3	0.000019	0	1

Tabla 5.1: Valores de los Hiper-Parametros para la realización de los fine-tunings.

Para ello se va a seguir un procedimiento basado en 3 pasos. Dicho procedimiento es el siguiente:

- **Elección del modelo**: Lo primero que se debe hacer es elegir con cual de las dos versiones de Whisper [35] se desea realizar el experimento. Las dos versiones que se ha decidido probar en este documento son *whisper-large-V2* y *whisper-large-V3*.
- **Selección del corpus de entrenamiento**: Lo siguiente es escoger con que corpora se desea realizar la especialización del modelo anteriormente seleccionado. Dado que queremos un modelo especializado en habla Down, los corpora entre los que podemos elegir son el corpus PRAUTOCAL o cualquiera de sus variantes aumentadas.
- **Evaluación del modelo obtenido**: Una vez tenemos el modelo ya especializado, el ultimo paso es medir cual ha sido el efecto de dicha especialización. Aunque nuestro objetivo es conseguir un modelo especializado en habla Down, también nos interesa ver como es la evolución del rendimiento del modelo para habla típica, es por eso por lo que en la evaluación se incluirán dos corpus de habla típica. Los corpus que se utilizaran en la evaluación son fijos, para que los resultados entre diferentes experimentos puedan ser comparables. Estos corpus son: *VoxPopuli*, *PRAUTOCAL Típico*, *PRAUTOCAL Down* y *PRAUTOCAL Down Clean*.

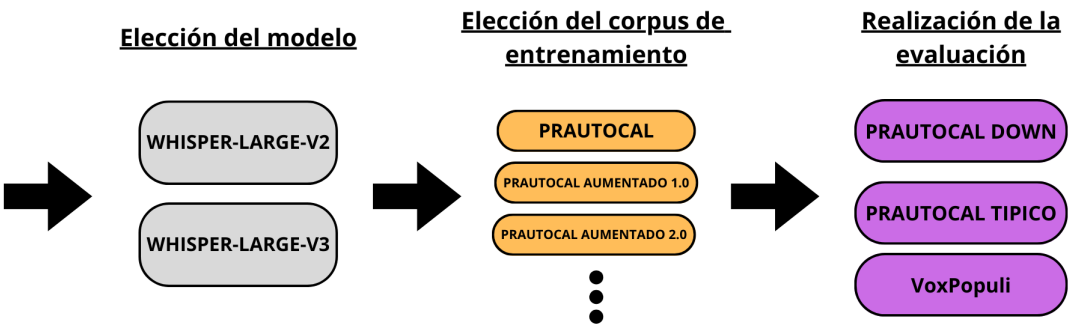


Figura 5.8: Resumen del procedimiento experimental que se va a seguir para conseguir el reconocedor Voz-Texto para habla Down

Con el fin de estructurar la experimentación se han decidido crear 4 clases de experimentos. Dichas clases siguen el procedimiento anteriormente explicado y toman diferentes decisiones en cada uno de los pasos del procedimiento. Dichas clases son las siguientes:

- **Experimentos Base**: Se evalúa el modelo base whisper-large-V2 con diferentes corpus y se realiza un primer fine-tuning con cada división del corpus PRAUTOCAL Down. El objetivo de este primera clase de experimentos es sentar una base con lo que poder comparar los posteriores resultados.
- **Experimentos Base con aumento de datos**: Se realizan 4 experimentos distintos por división, utilizando corpus aumentados con una sola técnica de las descritas. De esta forma podemos ver cual de las cuatro técnicas propuestas en la sección 4.1.4 ofrece mejores resultados.
- **Experimentos Mixtos con aumento de datos**: Se llevan a cabo 11 experimentos distintos para cada tipo de división, combinando dos, tres o cuatro de las técnicas de aumento de datos anteriormente descritas. Es importante destacar que las técnicas se aplican de manera aislada, es decir, si aplicamos al corpus PRAUTOCAL (único corpus que se le aplica aumento de datos) las técnicas de *Variación de Velocidad* y *Variación de Tono*, se generara un nuevo corpus donde cada fichero de audio solo contendrá una variación. Por ejemplo, de un fichero X obtendremos: fichero X, el fichero X con variación de velocidad y el fichero X con variación de tono. Por lo tanto, las variaciones no se aplican todas sobre el mismo fichero, sino que cada una genera su propio fichero modificado.
- **Experimentos con whisper-large-v3**: Se replican 3 experimentos sobre el nuevo modelo *whisper-large-v3*. Los tres experimentos seleccionados son los que se han considerado más significativos para la comparación con la versión V2 y los que se creen que podrían arrojar mejores resultados.

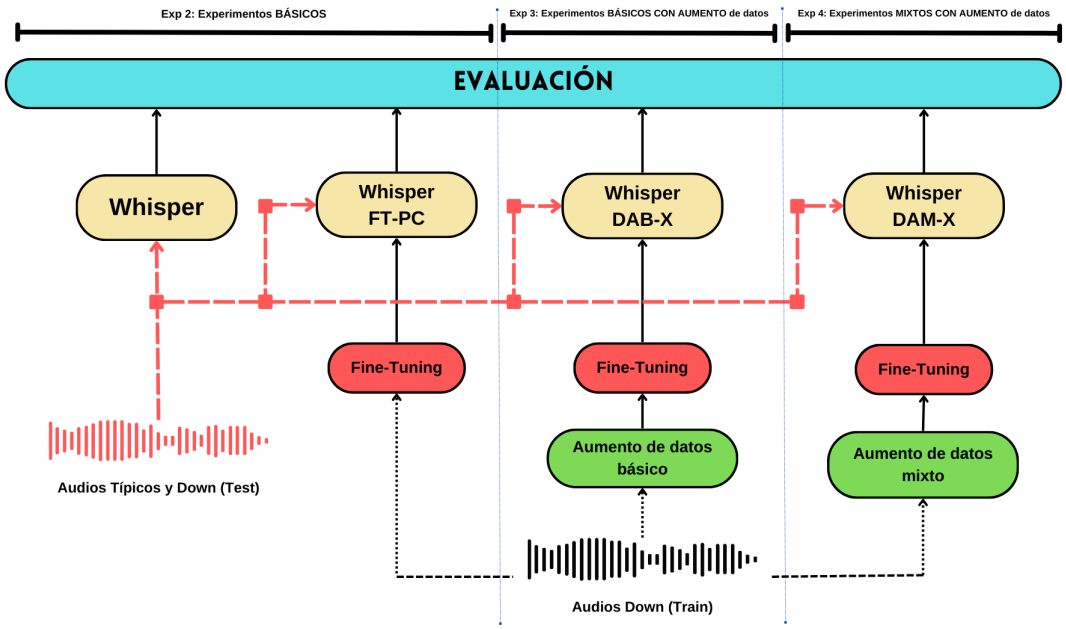


Figura 5.9: Resumen de como se van a ejecutar cada una de las diferentes clases de experimentos. En este resumen no se diferencia entre *whisper-large-v2* y *whisper-large-v3* dado que el procedimiento no varia de uno a otro.

5.5 Resultados

Como ya se ha mencionado en la sección 5.4 la experimentación de este apartado consta de 4 clases de experimentos. A continuación se mostraran, explicaran y analizaran los resultados obtenidos en

cada una de las diferentes clases:

5.5.1 Experimentos Base

Este conjunto de experimentos incluye 2 experimentos por división: **BASE** y **FT-PC**. **BASE** consiste en evaluar el modelo sin ningún *fine-tuning* realizado, para poder obtener así una primera referencia y un punto de partida para el desarrollo del trabajo. *FT-PC* consiste en realizar un primer *fine-tuning* del modelo con el corpus PRAUTOCAL, para tener una primera visión de como de influyente puede ser un *fine-tuning* sobre el modelo Whisper. Los resultados de ambos experimentos se pueden ver en las tablas 5.2 y 5.3.

Corpus	**BASE**	**FT-PC**	**DAB-1**	**DAB-2**	**DAB-3**	**DAB-4**
División Aleatoria						
VoxPopuli (ES)	8.422	15.585	15.390	16.083	16.940	17.602
PRAUTOCAL Típico (ES)	3.409	2.193	1.929	1.982	2.167	2.511
PRAUTOCAL Down (SD)	64.315	23.565	14.653	14.576	22.559	22.463
PRAUTOCAL Down Clean (SD)	35.858	17.354	14.653	**14.576**	15.747	15.021
Nº ficheros de audio (Train)	-	1208	2416	2416	3624	3624
División por Actividades						
VoxPopuli (ES)	8.422	15.941	16.662	15.409	19.100	18.742
PRAUTOCAL Típico (ES)	4.239	7.781	6.815	9.149	10.789	10.572
PRAUTOCAL Down (SD)	56.166	44.645	29.091	27.283	28.164	29.671
PRAUTOCAL Down Clean (SD)	37.668	28.376	27.355	**27.029**	28.164	29.551
Nº ficheros de audio (Train)	-	1275	2550	2550	3825	3825

Tabla 5.2: WER (%) para cada corpus y experimento realizado con *whisper-large-v2*. BASE: sin fine-tuning. FT-PC: Fine-tuning con PRAUTOCAL Down. DAB-X: Fine-tuning con PRAUTOCAL Down, pero aplicándole una técnica de aumento de datos (1- Ruido Blanco, 2- Ruido Rosa, 3- Variación Velocidad, 4- Variación Tono). El conjunto test contiene: 875 para la división aleatoria y 808 para la división por actividades.

Como se puede apreciar los resultados del modelo base para castellano típico son muy buenos, obteniendo en VoxPopuli un WER de 8.4 % y un BERT Score F1 de 0.97. En el otro corpus de habla típica que se presenta como corpus evaluador, PRAUTOCAL Típico, se obtienen todavía mejores resultados, con un WER de 3.4 % y un BERT Score F1 de 0.98. Esta diferencia, que es considerablemente grande en términos de WER, se debe principalmente a la naturaleza de los corpora. El VoxPopuli es un corpus bastante más complejo, ya que se basa en una recopilación de intervenciones parlamentarias, mientras que el PRAUTOCAL Típico esta constituido solo de una serie de actividades mucho mas cortas y de lenguaje mas coloquial.

La aplicación del *fine-tuning* con PRAUTOCAL tiene un gran impacto, tanto en el rendimiento de habla típica, como en el rendimiento de habla Down. Los rendimientos para VoxPopuli se degradan entre un 8 % y 9 %, mientras que los resultados para PRAUTOCAL Down mejoran considerablemente. Se puede apreciar como existe una bajada de casi un 20 % para PRAUTOCAL Down Clean en división aleatoria y otra de un 10 % en división por actividad. Esta diferencia tan grande entre las diferentes divisiones se debe a que en la división aleatoria existe la posibilidad de que el modelo haya visto en el entrenamiento actividades que se repiten en la evaluación, y por tanto, existe un minúsculo proceso de *overfitting* en el cual el modelo memoriza ciertas características no generales de las muestras de *train*, que al verse repetidas en la muestra de *test* hacen que el WER disminuya. En el caso de división por actividad, al no coincidir ninguna actividad en los conjuntos *train* y *test*, los resultados que podemos

apreciar son los referentes a lo que realmente ha aprendido el modelo del habla Down y no de las actividades.

La justificación por la cual ocurre un proceso similar con el corpus PRAUTOCAL Típico, es similar a la del PRAUTOCAL Down. En el caso de la división aleatoria el modelo no se esta centrando solo en aprender características propias del habla Down, sino que también aprende de las frases y por eso mejora el rendimiento. Pero cuando el modelo no es capaz de aprender de las frases porque la propia división se lo impide, se ve como su comportamiento es el esperado y ocurre lo mismo que con VoxPopuli, que su rendimiento cae. Este suceso que se nos muestra con VoxPopuli y PRAUTOCAL Típico nos ejemplifica muy claramente lo influyente que es el *fine-tuning*, que hace que el modelo se adapte mucho al nuevo conocimiento, generando que pierda parte de su conocimiento anterior.

Por ultimo, cabe destacar que los resultados obtenidos para PRAUTOCAL Down (NO CLEAN) hacen sospechar que la aparición de frases bucles pueda tener una cierta relación con el grado de seguridad que tenga el modelo sobre su predicción. Esta suposición se debe a la gran diferencia de 20 puntos porcentuales existente entre los resultados de las dos divisiones del experimento **FT-PC**.

5.5.2 Experimentos Base con aumento de datos

Este conjunto de experimentos engloba los siguientes 4 experimentos por división: **DAB-1**, **DAB-2**, **DAB-3** y **DAB-4**. Estos experimentos consisten en aplicar, de forma separada, cada una de las 4 técnicas descritas en la sección 4.1.4 al corpus PRAUTOCAL Down y posteriormente realizar un *fine-tuning* con cada una de las nuevas versiones aumentadas del PRAUTOCAL Down. Con estos experimentos podemos comprobar varias cosas, como cual de las 4 técnicas es la más eficiente, como de útil es el aumento de datos para el problema al que nos estamos enfrentando, como de influyente es que el número de ficheros de *train* crezca, etc. Cabe destacar que en este caso, la métrica BERT se ha calculado solo para el experimento **DAB-2**, que es el mejor resultado obtenido en este sección, con el objetivo de corroborar la mejora vista en el WER Los resultados de estos experimentos se pueden ver en las tablas 5.2 y 5.3. En la tabla 5.6 se puede ver que combinación de técnicas constituyen cada versión aumentada del corpus PRAUTOCAL.

Modelo	Corpus	BASE	FT-PC	DAB-2
V2	**División Aleatoria**			
	VoxPopuli (ES)	0.975	0.953	0.951
	PRAUTOCAL Típico (ES)	0.988	0.991	0.992
	PRAUTOCAL Down Clean (SD)	0.882	0.947	0.953
	División por Actividades			
	VoxPopuli (ES)	0.975	0.950	0.951
	PRAUTOCAL Típico (ES)	0.984	0.976	0.973
	PRAUTOCAL Down Clean (SD)	0.876	0.914	0.916
V3	**División Aleatoria**			
	VoxPopuli (ES)	0.965	0.961	0.959
	PRAUTOCAL Típico (ES)	0.989	0.992	0.989
	PRAUTOCAL Down Clean (SD)	0.893	0.948	0.952
	División por Actividades			
	VoxPopuli (ES)	0.965	0.959	0.961
	PRAUTOCAL Típico (ES)	0.987	0.976	0.976
	PRAUTOCAL Down Clean (SD)	0.895	0.925	0.925

Tabla 5.3: Análisis con la métrica BERTScore F1 de los modelos más significativos que se han generado a lo largo de toda la experimentación.

Se puede comprobar que el comportamiento para habla típica sigue empeorando a medida que se entrena al modelo con más muestras de habla Down. Esta tendencia es independiente de la división, por lo que se puede suponer que es dependiente del tamaño de la partición *train*, ya que a mayor numero de ejemplos de Down que se le muestran al modelo, mas intenta adaptase a ellos, y por lo tanto, mas olvida su conocimiento anterior. En el caso de PRAUTOCAL Típico, la tendencia es la misma que la vista en la sección 5.5.1, donde para la división aleatoria los valores WER y BERT siguen mejorando, debido a que ahora hay una mayor repetición de las actividades al realizarse el aumento de datos, y donde para la división por actividad los valores WER y BERT empeoran al sufrir el mismo proceso que sufre VoxPopuli.

En cuanto al rendimiento en habla Down, podemos apreciar como si que existe una gran mejora para el caso de la división aleatoria. La explicación es la misma que la de los resultados en PRAUTOCAL Típico, la mayor repetición de frases en el conjunto *train* que luego van a estar también en el conjunto *test* hace que el modelo no solo se adapte a la voz Down, sino también a las frases, y por tanto, el rendimiento mejore. En el caso de la división por actividad se puede apreciar una ligera mejora de alrededor del 1 %, en PRAUTOCAL Down Clean, al aplicar las técnicas básicas de aumento de datos. Eso si, esta mejora no es constante para todas las técnicas como en la división aleatoria, sino que solo aparece en *DAB-1* y *DAB-2*, lo que corresponde a las dos técnicas que se basan en la introducción de ruido.

Los resultados para cada una de las diferentes técnicas en ambas actividades nos muestran claramente que las técnicas mas eficientes son *Introducción de Ruido Rosa* e *Introducción de Ruido Blanco*, que curiosamente son ambas las referentes a la introducción de ruido en los audios. Esto se puede deber a que la introducción de ruido en ciertos casos puede llegar a simular ciertas conductas del habla Down, como trabadas, elongaciones, pseudo-habla... Esto puede ayudar a que el modelo generalice mejor esos tipos de audios, y por tanto, se obtenga un mejor rendimiento. Quede pendiente para un trabajo futuro, el comprobar si la mejora de rendimiento se encuentra en los audios que mas conductas de este tipo tienen.

En lo que respecta a la conclusión que habíamos obtenido en la sección 5.5.1 sobre la aparición de frases bucle, esta nueva experimentación nos muestra que es errónea. En este caso las diferencias entre PRAUTOCAL Down y PRAUTOCAL Down Clean son menores en el caso de la división por actividad. Esto nos hace suponer que el comportamiento de este suceso tiene un fuerte grado de aleatoriedad.

5.5.3 Experimentos Mixtos con aumento de datos

Este conjunto de experimentos es el más amplio de todos, ya que se compone de un total de 11 experimentos por división: **DAM-5**...**DAM-15**. Esta experimentación consiste en probar todas las combinaciones posibles de dos, tres y cuatro técnicas de aumento de datos básicas, de las probadas en la sección 5.5.2. De esta forma podemos analizar como de bien o de mal interactúan las diferentes técnicas entre si, y obtener así más pruebas para comprobar cual es la mejor forma de aumentar el rendimiento del modelo. Debido a los resultados obtenidos en el WER en esta sección, se ha decidido omitir el calculo del BERT para dicha experimentación. Los resultados de esta sección se pueden ver en la tabla 5.4. En la tabla 5.6 se puede ver que combinación de técnicas constituyen cada versión aumentada del corpus PRAUTOCAL.

Como se puede apreciar con VoxPopuli y PRAUTOCAL Típico, su comportamiento no varia con respecto a los experimentos anteriores. VoxPopuli siguen empeorando progresivamente, en ambas divisiones a medida que el numero de muestras de habla Down aumenta. Y PRAUTOCAL Típico, mejora su rendimiento a medida que la partición de *train* crece en el caso de la división aleatoria y ante el mismo suceso, empeora su rendimiento en el caso de la división por actividad. Esto nos reafirma en que el modelo se ve muy influido por el *fine-tuning*.

Con lo que respecta al habla Down, no se ve ninguna mejora en ninguno de los experimentos con respecto al mejor resultado obtenido en la sección 5.5.2. Esto nos indica que el modelo ha alcanzado el máximo aprendizaje que podía realizar del corpus PRAUTOCAL, tanto para la división aleatoria,

EXP	VoxPopuli	PRT	PRD	PRDC	Nº Audios
Divison Aleatoria					
MEJOR	19.122	2.498	24.478	**14.182**	3624
DAM-5	17.732	2.199	23.338	15.647	3624
DAM-6	19.685	2.241	24.266	14.770	4832
DAM-7	20.005	2.135	23.126	15.009	4832
DAM-8	19.297	2.295	21.774	15.665	4832
DAM-9	18.651	2.124	19.339	15.447	4832
DAM-10	19.609	2.049	15.243	15.150	6040
DAM-11	18.753	2.124	20.672	15.033	6040
DAM-12	19.522	2.199	22.083	14.740	6040
DAM-13	20.039	2.284	17.755	14.765	7248
DAM-14	21.224	2.199	19.571	14.865	7248
DAM-15	22.855	2.241	15.823	15.731	8456
División por Actividades					
MEJOR	18.922	9.149	27.283	**27.029**	3825
DAM-5	18.412	8.988	29.648	29.192	3825
DAM-6	20.370	9.230	30.992	30.992	5100
DAM-7	20.804	11.376	46.755	32.952	5100
DAM-8	20.439	8.506	30.111	30.111	5100
DAM-9	21.658	12.745	34.145	34.145	5100
DAM-10	23.264	13.845	32.290	31.959	6375
DAM-11	20.395	11.886	31.317	31.016	6375
DAM-12	23.609	12.477	36.996	31.019	6375
DAM-13	24.914	15.347	33.171	33.015	7650
DAM-14	24.190	15.643	33.333	33.061	7650
DAM-15	24.590	13.308	35.582	32.923	8925

Tabla 5.4: WER (%) para cada corpus y experimento aumentado mixto realizado en este trabajo. MEJOR: Tomaremos como punto de comparación el mejor resultado obtenido en el conjunto de experimentos anterior. DAM-X: DAM indica que es un experimento mixto con datos aumentado. X es un número que indica la versión de PRAUTOCAL DOWN AUMENTADO que se ha usado en el experimento. PRD: PRAUTOCAL DOWN. PRDC: PRAUTOCAL DOWN CLEAN. PRT: PRAUTOCAL TIPICO.

como para la división por actividad. También se puede comprobar que los mejores resultados obtenidos en esta sección corresponden a combinaciones de técnicas que incluyen al menos una de introducción de ruido, lo cual nos confirma que estas dos técnicas son las mas eficaces y las que mejores resultados ofrecen.

Por ultimo, en lo que se refiere a la aparición de frase bucle, es decir, la diferencia de rendimiento entre PRAUTOCAL Down y PRAUTOCAL Down Clean, se puede ver una tendencia similar a la vista en la sección 5.5.2, donde este suceso se veía muy incrementado para la división aleatoria, y ocurría en menor medida en la división por actividad. Esta tendencia nos hace sospechar que probablemente la primera hipótesis planteada en la sección 5.5.1 no sea para nada correcta, y que la aparición de frases bucle este más relacionada certidumbre de la predicción, que con la incertidumbre, como habíamos dicho anteriormente. Aun asi, es un suceso muy extraño y necesita un estudio en profundidad para poder sacar conclusiones fiables.

5.5.4 Experimentos con whisper-large-v3

Finalmente, en este ultimo conjunto de experimentos se van a repetir los 3 experimentos que se consideran mas importantes y representativos de toda la experimentación anterior pero con *whisper-large-v3*: **BASE**, **FT-PC** y **DAB-2**. El objetivo de esta ultima experimentación es realizar una pequeña aproximación a esta nueva versión de Whisper (que fue lanzada a mientras se desarrollaba este trabajo), para comprobar si realmente existe una mejora en dicha versión, con lo que respecta a habla Down. Los resultados de esta experimentación se pueden ver en las tablas 5.5 y 5.3.

Corpus	**BASE**	**FT-PC**	**DAB-2**
División Aleatoria			
VoxPopuli (ES)	11.214	13.116	13.657
PRAUTOCAL Típico (ES)	3.013	1.929	2.748
PRAUTOCAL Down (SD)	48.734	24.995	22.695
PRAUTOCAL Down Clean (SD)	32.232	15.908	**15.471**
División por Actividades			
VoxPopuli (ES)	11.214	12.714	12.726
PRAUTOCAL Típico (ES)	3.488	7.137	7.352
PRAUTOCAL Down (SD)	52.851	51.553	32.221
PRAUTOCAL Down Clean (SD)	32.919	25.336	**25.175**

Tabla 5.5: WER (%) para cada corpus y experimento realizado con *whisper-large-v3*. BASE: sin fine-tuning. FT-PC: Fine-tuning con PRAUTOCAL Down. DAB-2: Fine-tuning con PRAUTOCAL Down, pero aplicándole la técnica que mejores resultados ha dado en la experimentación con *whisper-large-v2* (Ruido Rosa).

En lo que respecta al habla castellana típica se puede ver que el modelo base ha empeorado hasta un 3 % en VoxPopuli, aunque se ve una ligerisima mejora en lo que respecta a PRAUTOCAL Típico. Esto seguramente de deba a la naturaleza del nuevo entrenamiento al que ha sido sometido el modelo. En lo referente a los dos experimentos que involucran la realización de un *fine-tuning* se puede ver como la tendencia de evolución es la misma que la que veníamos observando.

En lo que respecta al habla Down, los valores del modelo base son hasta un 5 % mejores, lo que es un gran aumento del rendimiento del modelo. En lo que respecta a la división aleatoria, se puede ver que el rendimiento del modelo ha disminuido, mientras que para la división por actividad el rendimiento ha mejorado prácticamente un 2 %. Esto nos indica que la capacidad de aprendizaje de las características propias del habla Down (lo correspondiente a la división por actividad) ha aumentado, pero por otro lado ya no es capaz de adaptarse tan bien a las frases (lo correspondiente a la división aleatoria), seguramente como consecuencia del nuevo entrenamiento del modelo. Esto nos beneficia, ya que nuestro mayor interés es que el modelo sea capaz de aprender características generales del habla Down.

El comportamiento de la aparición de frases bucle vuelve a ser contradictorio con las conclusiones anteriores, lo que nos lleva a pensar que depende de factores o bien totalmente aleatorios, o bien que no estamos teniendo en cuenta. Queda como tarea pendiente el realizar una estudio mas exhaustivo del porque de este suceso.

5.6 Conclusiones

Los resultados presentados en este trabajo muestran que los modelos base *whisper-large-v2* y *whisper-large-v3* tienen un rendimiento muy malo en el reconocimiento de habla Down española. Aun así cabe destacar que su desempeño en esta tarea, o similares, es superior a la gran mayoría de modelos

Corpus	Velocidad	Ruido Blanco	Ruido Rosa	Tono	Nº audios
Prautocal					1275
Aumentado 1.0		X			2550
Aumentado 2.0			X		2550
Aumentado 3.0	X				3825
Aumentado 4.0				X	3825
Aumentado 5.0		X	X		3825
Aumentado 6.0	X		X		5100
Aumentado 7.0			X	X	5100
Aumentado 8.0	X	X			5100
Aumentado 9.0		X		X	5100
Aumentado 10.0	X			X	6375
Aumentado 11.0	X	X	X		6375
Aumentado 12.0		X	X	X	6375
Aumentado 13.0	X	X		X	7650
Aumentado 14.0	X		X	X	7650
Aumentado 15.0	X	X	X	X	8925

Tabla 5.6: Resumen de las técnicas de aumento de datos usadas en cada corpus y del tamaño final de cada uno de ellos

ASR actuales [4]. Por otro lado, se ha demostrado que aplicar la técnica de *fine-tuning* con un corpus Down sobre estos modelos, proporciona grandes mejoras, tanto en la capacidad de adaptación del modelo al habla Down (división por actividad), como en la adaptación específica a la tarea del corpus PRAUTOCAL (división aleatoria). La mejora anteriormente citada, siempre lleva consigo un decremento en el rendimiento del modelo para habla típica, por lo que podemos concluir que el *fine-tuning* del modelo *whisper* tiene una capacidad de aprendizaje limitada.

En lo que respecta a la aplicación de técnicas de aumento de datos, se observa que su eficacia es limitada cuando nos centramos en obtener mejoras en características generales del habla Down, pero sí resultan útiles para mejorar el rendimiento en escenarios dependientes de tarea, en el corpus en cuestión.

En relación a la experimentación realizada, podemos concluir que la versión v3 del modelo *whisper* no es esencialmente mejor que la versión v2, en lo que a reconocimiento de habla Down española respecta.

Por último cabe destacar que el trabajo realizado en este experimento fue enviado al congreso SEPLN 2024[2] y dicho artículo ha sido aceptado y será expuesto en la edición de este año de dicho congreso. El articulo completo se puede ver adjuntado en el apéndice C

5.6.1 Trabajo futuro

Como trabajo futuro queda la exploración de técnicas de aumento de datos más complejas, como puede ser conversión de voz, o la aplicación de un TTS para simular la voz Down y aumentar la variedad léxica del corpus PRAUTOCAL. También queda como trabajo pendiente realizar un estudio más en profundidad de los resultados obtenidos en este artículo, centrando la atención en explicar porque se han obtenido estos resultados y su relación con las características propias de los hablantes Down. Por otro lado, queda pendiente realizar un estudio y una experimentación más profunda y completa sobre el nuevo modelo *whisper-large-v3*. Y finalmente, se propone como trabajo futuro realizar un estudio

[2] http://sepln2024.infor.uva.es/

del fenómeno de las “frases bucle”, intentando explicar todo el enigma que, como se ha visto en este documento, las rodea.

Capítulo 6

EXP 2: Calidad del Habla Down

6.1 Soluciones Existentes

La aplicación de la métrica GoP para la evaluación del habla tuvo mucho éxito y fue muy estudiada con la aparición de los modelos de Markov. Estos modelo tenían la ventaja de tener como output probabilidades, lo cual es muy beneficioso de cara a la aplicación de esta técnica.

Con el resurgimiento de los modelos profundo de inteligencia artificial se reavivo una linea de investigación que propone usar el output de estos modelos (LLMs sobre todo) para calcular la métrica GoP. Dicha tarea no resulto ser tan trivial, debido a que el output de estos modelo no son probabilidades, sino que simplemente son valores que tienen un significado para el propio modelo. Ante esta situación surgieron muchas soluciones novedosas, una de ellas fue mezclar lo nuevo con lo antiguo y fusionar estos modelos profundos con los modelos de Markov [42, 37, 17]. Otros investigadores han decidido optar por añadir capas extras a los modelos para así conseguir transformar su salida en las probabilidades pertinentes, consiguiendo así sustituir el los modelos de Markov por estructuras más simples, como DNNs de una sola capa [51]. Por último, otra corriente actual es la de realizar modificaciones a la fórmula actual del GoP para adecuarla más a los problemas y modelos actuales [42, 46].

El GoP, aun siendo la métrica que vamos a usar en este estudio, no es para nada la única que se usa en el ámbito de la evaluación de la pronunciación. Estudios como [41] exploran diferentes métodos de evaluación como por ejemplo, arboles de decisión, LDA-APF, LDA-MFCC... Todas estas métricas tienen como objetivo final es el mismo que el GoP, dar una medida de la calidad del habla.

6.2 Adaptación de los datos

En el caso del evaluador automático del habla el formato estándar compuesto por pares audio-transcripción no es valido. Esto se debe al código que hemos decidido emplear para esta sección [1] y al tipo de evaluación que se ha decidido emplear, que requiere de un formato específico denominado TextGrid, dado que la evaluación de una frase no va a ser palabra a palabra, sino fono a fono. Debido a ello debemos segmentar los corpora para pasar de pares audio-transcripción a pares de fono - representación fonética.

Vamos a emplear solo dos corpora para este experimento: CommonPhone, para la parte correspondiente al entrenamiento del modelo, y PRAUTOCAL, para la parte de evaluación. En el caso del CommonPhone su formato original ya es un formato basado en TextGrid, ya que se creo con el fin de usarse en experimentos fonéticos, como es nuestro caso. Pero en el caso del PRAUTOCAL, su formato

[1] https://github.com/juice500ml/ dysarthria-gop

es de pares audio-transcripción, por lo que debemos aplicar una transformación sobre el para conseguir los TextGrid. Afortunadamente, dicha conversión ya había sido realizada con anterioridad, y por lo tanto, solo se tuvo que solicitar y una vez obtenida se pudo empezar a usar.

Cabe destacar, que no es simple encontrar buenos corpus que tengan este formato, ya que es un formato mucho más costoso y laborioso de obtener. Conseguir pares audio-transcripción es algo mucho más trivial, ya que el único trabajo consiste en etiquetar los audios recogidos. Pero obtener los TextGrid requiere primero de una segunda etapa, en la cual se debe obtener la transcripción fonética de la frase. Y finalmente existe una tercera etapa en la cual se deben aplicar técnicas de alineamiento forzado para conseguir emparejar cada fono con el intervalo del audio en el que se pronuncia.

La salida del modelo Wav2Vec tampoco necesita ningún tipo de adaptación, ya que no estamos interesados en tareas de traducción o transcripción, sino que lo que nos interesa son los valores que nos da el modelo, y con ellos calcular las diferentes versiones del GoP. Por lo tanto, la única transformación necesaria para este experimento es obtener los TextGrids de los corpora que se deseen utilizar.

6.3 Adaptación de los modelos

La adaptación del modelo *Wav2Vec* para nuestro experimento es compleja, y consta de 3 fases: primero una fase en la cual desechamos parte del modelo, luego una segunda fase en la cual añadimos un nuevo modulo y finalmente una tercera fase en la cual realizamos un ajuste de hiper-parámetros al nuevo modelo que hemos construido. A continuación se detallan las 3 fases:

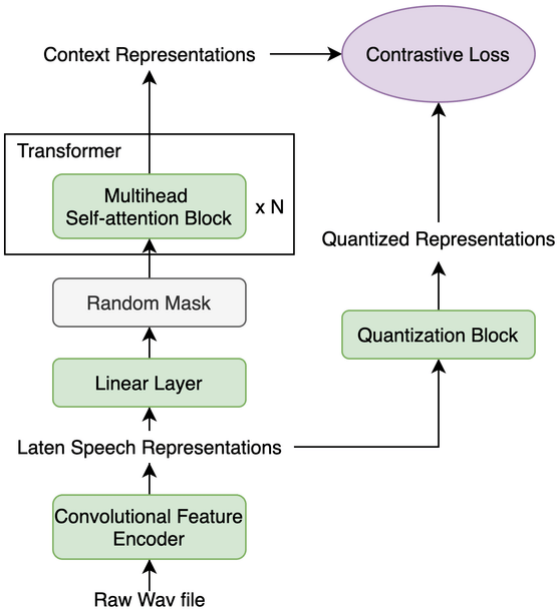


Figura 6.1: Diagrama arquitectura Wav2Vec.
Imagen obtenida de https://www.researchgate.net/publication/350720107_Interpreting_A_Pretrained_Model_Is_A_Key_For_Model_Architecture_Optimization_A_Case_Study_On_Wav2Vec_20

El modelo Wav2Vec no esta preparado para ser usado de la forma en la cual lo queremos utilizar en este experimento. Wav2Vec es un modelo ASR que es capaz de realizar tanto transcripción como traducción, pero en ningún caso evaluación. En la imagen 6.1, se puede apreciar como es la arquitectura del modelo, viendo que esta formado por diferentes módulos, cada uno de distinta naturaleza (módulos convolucionales, módulos de atención...). En esta primera fase, vamos a desechar gran parte del modelo, quedándonos solo con la parte convolucional. Esta parte convolucional es la que se denominada comúnmente, extractor de características, es decir, es la parte que se encarga de convertir

la representación binaria del audio de entrada, en un vector de n posiciones, donde cada posición corresponde a una característica del audio original. El extenso y diverso entrenamiento al que ha sido sometido Wav2Vec hace que este extractor de características nos sea muy útil de cara a obtener una representación manejable de los audios que queremos evaluar.

Ahora ya tenemos una forma de transformar nuestros audios a vectores, por lo tanto, solo necesitamos una forma de modificar dichos vectores para obtener de ellos la información que deseamos, en este caso una medida de como de buena ha sido la pronunciación. En la segunda fase de la adaptación, vamos a coger el extractor de características anterior y lo vamos unir a una red Feed Forward de una sola capa. Con esta red lo que queremos es convertir la salida del extractor de características en un vector de dimensión m, donde m es el numero de fonos del idioma que se esta evaluando. De esta forma conseguimos un modelo que, dado un audio donde se pronuncia un fono nos devuelve un vector de probabilidades, donde cada probabilidad hace referencia a la certeza que tiene el modelo de haber escuchado dicho fono. Por lo tanto, acabamos de crear un reconocedor de fonos. Posteriormente juntando esas probabilidades con la métrica GoP se obtendrá la evaluación de la pronunciación.

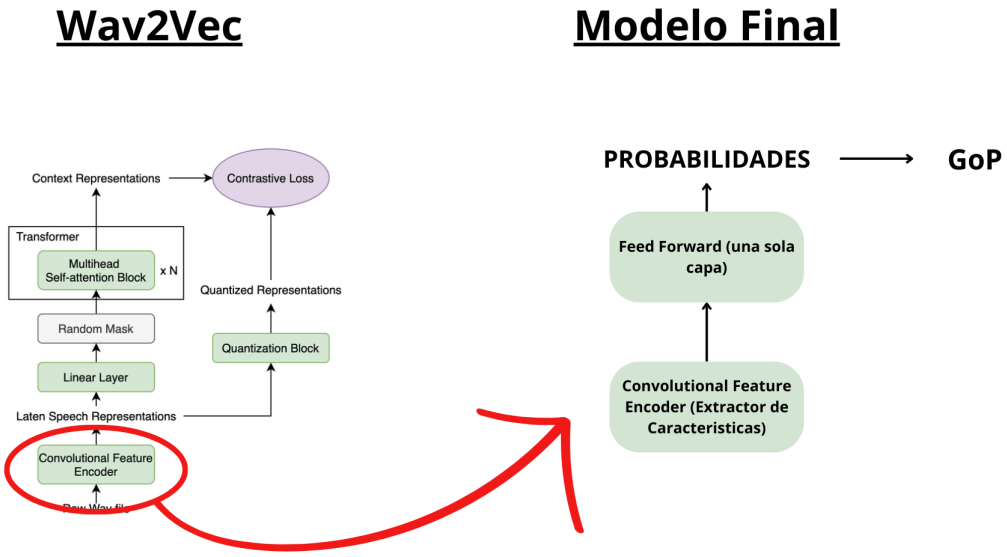


Figura 6.2: Resumen de la transformación del Wav2Vec en el modelo que se va a usar

Finalmente, una vez ya hemos construido el modelo solo queda ajustar sus hiper-parámetros. El procedimiento de ajuste es el mismo que el visto en la sección 5.3. El resultados de dicho ajuste fue: **epochs**: 1, **learning_rate**: 0.001 y **batch_size**: 4. Destacar que el entrenamiento solo se realiza sobre la capa *Feed Forward*, el resto se mantiene congelado.

6.4 Procedimiento Experimental

El caso del evaluador del habla es un poco peculiar, como ya se ha ido mencionando a lo largo del documento. En esta sección se pueden ver dos partes fundamentales, una parte de experimentación y una parte de análisis. La experimentación esta compuesta por un total de 6 experimentos, y el análisis es una exploración en profundidad del mejor resultado obtenido en la experimentación.

En lo que respecta a la experimentación si que se puede apreciar un esqueleto común a cada uno de los 6 experimentos diferentes. Dicho esquema cuenta con dos pasos, y es el siguiente:

- **Selección de la partición de CommonPhone**: Como ya se explico en la seccíon 4.2.2 el

corpus Common Phone esta compuesto por diferentes particiones, donde cada una corresponde a un conjunto de fonos en un determinado idioma. En este paso se decide variar el conjunto de particiones, es decir de idiomas, que forman el conjunto de entrenamiento, dando así lugar a diferentes modelos. Este se decide realizar con la intención de analizar cual es la importancia del multilingüismo en el entrenamiento de un modelo de este estilo.

- **Evaluación de correlación del modelo obtenido**: En este paso se evalúa cada uno de los modelos resultantes del paso anterior. Aquí existe otra decisión a tomar, y es si se quiere evaluar el modelo sobre todo el corpus PRAUTOCAL o solo sobre una partición de él. Una vez elegido esto se pasa el corpus PRAUTOCAL por el evaluador, obteniendo así los valores GoP por frase, y posteriormente se calcula la correlación lineal entre los valores GoP y las clases en las que se clasifican los audios del PRAUTOCAL, que fueron definidas por una lingüista y que se explican en [9]. Se decidió usar la correlación como métrica evaluadora porque era la métrica que se usa en el paper original, en el cual se basa este trabajo [51].

Al seguir el esquema anterior con los 6 experimentos que se deseaban realizar se obtienen 6 modelos diferentes, cada uno con una correlación distinta. Como mejor modelo se tomo el que tenia mayor correlación, ya que es el que demuestra tener una mayor capacidad de clasificación, y por tanto en este caso, de evaluación. Dicho modelo es el que sera sometido a un análisis para poder estudiar el porque de dicha correlación y para poder ratificar lo bueno o malo que es el modelo.

Dado que solo se realiza un único análisis, no podemos decir que existe un esquema común, ya que solo hay un caso. Aun así, se ha decidido establecer un esquema de análisis para poder replicarlo con los futuros experimentos y que sea mas fácil realizar, ya no solo el propio análisis, sino también la comparación de resultados. El esquema de análisis cuenta con 4 fases, las cuales son:

- **Análisis por Frases**: Para realizar este análisis se escogen los valores de GoP de la variación de la métrica GoP que mayor correlación proporcionaba. Una vez obtenidos se pasa a realizar un análisis individual de cada frase, generando así una ficha por frase que consta de: transcripción de la frase, transcripción fonética de la frase, clase de la frase, GoP medio de la frase, GoP acumulado de la frase y GoP por fono de la frase.

  El GoP medio de la frase se calcula fácilmente sumando el valor del GoP de cada fono que compone la frase y dividiendo entre el numero total de fonos de la misma. Esta medida nos sirve para poder resumir la frase en un único valor, el cual nos puede ayudar a comparar diferentes frases y nos sirve para futuras evaluaciones como el análisis de intervalos.

  El GoP acumulado de la frase no es más que la suma del GoP de cada fono que compone la frase. Este medida nos sirve para comprobar la importancia de la longitud de la frase y poder estudiar así como se comporta el modelo con frases cortas y como lo hace con frases largas.

  Finalmente, **el GoP por fono de la frase** es la métrica mas interesante de todas, ya que nos dice como de bien se ha pronunciado cada fono de la frase. Esta información puede ser muy útil para saber en que fonos tiene mas problemas el paciente y por tanto, cuales son los que mas hay que trabajar. A parte de esto, la comparación de un mismo fono pronunciado varias veces con sus diferentes valores GoP puede darnos una idea de que considera el modelo una buena pronunciación y que no.

- **Análisis de diccionarios**: En este estudio lo que vamos hacer es comparar el diccionario que crea el programa, con el diccionario que realmente usa. Llamamos diccionario a un conjunto de fonos, por lo que el diccionario que crea el programa es el conjunto de fonos formado por la concatenación de todos los fonos diferentes que el modelo ve en el proceso de entrenamiento. Por otro lado, el diccionario que utiliza es el conjunto de fonos que el modelo evalúa durante el proceso de evaluación (que se realiza con PRAUTOCAL). Realizando esta comparación podemos ver como de eficiente esta siendo el entrenamiento, dado que si el diccionario del programa tiene mucho

mas fonos que el que realmente utiliza, entonces eso nos diría que el proceso de entrenamiento es mejorable, ya que estamos entrenando al modelo con muchos fonos que no le sirven, porque nunca los va a ver en la evaluación.

- **Análisis de Medias**: En este análisis se cogen los valores GoP de las tres variaciones del GoP que mejor correlación han generado. Posteriormente se calcula la media de GoP por frase, para tener un valor correcto que represente a la frase, ya que el GoP acumulado es muy dependiente de la longitud de la frase. Finalmente se calcula la media de dichos valores pero por clase, es decir, se calcula la media de todas las frases que pertenezcan a la clase 1, luego de todas las de la clase 2 y así sucesivamente.

  Dado que lo que estamos buscando es construir un evaluador automático lo que necesitamos es que los valores de la clase 1 sean peores/mejores (depende de como sea la formula de GoP que usemos) que los de la clase 2, que deberían ser peores/mejores que los de la clase 3 y que finalmente deberían ser peores/mejores que los de la clase 4. De esta forma podríamos establecer un evaluador que nos diga como de bien pronunciada esta una frase.

- **Análisis por fonos**: Al igual que en los casos anteriores, aquí nos vamos a quedar con las tres variaciones de GoP que mayor correlación proporcionan.

  El GoP medio por fono consiste en calcular la media de todos los valores GoP de cada fono. Estos valores posteriormente se ordenan de menor a mayor, obteniendo así una clasificación. Esta medida no nos da ninguna información especifica de cada individuo, pero si que nos muestra cuales son los fonos que en general las personas con SD peor pronuncian. Esta información puede ser muy útil para desarrollar medidas correctivas generales, como pueden ser ejercicios de mejora de pronunciación generales.

  Por otro lado, en cuanto **la desviación media por fono y variación** consiste en calcular la desviación media, no de los valores GoP, sino de la clasificación de fonos que habíamos realizado anteriormente. Es decir, calculamos la desviación media de la posición que cada variante le ha dado a un mismo fono. De esta forma obtenemos una medida que nos permite ver que nivel de consenso tienen las diferentes variantes, y ver así si los resultados son validos, ya que si cada variación dijese una cosa distinta, los resultados carecerían de valor.

$$DM = \frac{\Sigma \left| X_i - X \right|}{N}$$

Σ = Suma de los términos	X = Media del conjunto de datos
$\lvert Xi - X \rvert$ = Valor absoluto de la diferencia entre cada dato y la media	N = Número de datos en el conjunto

- **Análisis de Intervalos**: Este análisis se basa en representar un intervalo (limite inferior y limite superior) por cada clase. Si el programa fuese perfecto estos intervalos deberían estar perfectamente separados, ya que todos los valores GoP de cada frase (representaremos cada frase por su GoP medio) de la clase n deberían de ser menores/mayores que los de la clase $n+1$. Esto ultimo, indicaría que la clasificación del modelo coincidiría con la de la lingüista (una experta), y por tanto, denotaría el buen rendimiento del modelo.

  El problema en este análisis viene al intentar representar los diferentes intervalos, ya que si, por ejemplo, cogemos el valor más pequeño y mas grande de cada clase, estaríamos seguramente

usando como límites outliers. Para tratar de eliminar dichos valores y que así no interfieran en el estudio, vamos a usar el método Turkey. Este método utiliza el *rango intercuartilico* (diferencia entre tercer cuartil y primer cuartil) para delimitar que valores son atípicos y cuales no. Por lo tanto, denominaremos valor atípico a cualquier valor que se encuentre $1{,}5 * IQR$ por debajo del primer cuartil o cualquiera que se encuentre $1{,}5 + IQR$ por encima del tercer cuartil.

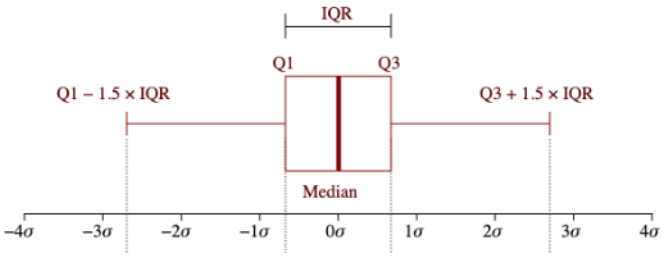


Figura 6.3: Representación del Test de Turkey

Por lo tanto, para nuestro intervalos estableceremos como limite inferior de cada clase el $max(Q1 - 1{,}5 * IQR, minValor)$, siendo $minValor$ el valor mínimo de la clase. Como limite superior de cada clase se establecerá el $min(Q3 - 1{,}5 * IQR, maxValor)$, siendo $maxValor$ el valor máximo de la clase. Calculamos el min/max porque es posible que al calcular el límite que establece el método Turkey tomemos un valor menor/mayor que el valor mas pequeño/grande de la clase (el caso en el que no existen valores atípicos).

Por ultimo, con la intención de cuantificar la representación por intervalos anteriormente mencionada vamos a crear una nueva métrica llamada **Frases clasificadas de forma única** (FCU), es decir, frases cuyo valor GoP medio no se encuentra en una intersección de intervalos, sino que pertenece a un único intervalo. Con esta métrica podremos dar un valor numérico a la representación por intervalos, lo que nos permitirá comparar fácilmente si los pasos que damos hacer mejorar el modelo o no. Finalmente, dividiremos las FCU entre el numero total de frase para obtener una medida similar a la precisión. Esto lo haremos por clase y de forma total, con el fin de conseguir un entendimiento mejor del modelo, ya que se podría dar el caso en que el modelo funcione muy bien para ciertas clases pero no para otras, y que esto empañe el resultado final haciendo parecer que el modelo tiene un rendimiento aceptable.

$$FCU = FrasesClasificadasdeformaUnica$$
$$PrecisionFCU = \frac{FCU}{Ntotalfrases}$$

6.5 Resultados

Como se explico en la sección 6.4 esta sección consiste en dos partes muy diferenciadas, la experimentación y el análisis del mejor resultado. Por lo tanto, vamos a comenzar explicando los diferentes experimentos que se han realizado y cuales han sido los resultados obtenidos.

6.5.1 Experimentación

La experimentación consiste en la realización de 6 experimentos, donde cada uno usa un conjunto de idiomas distintos. Esto se debe a que se quiere analizar y comprobar cual es la influencia que tiene el multilingüismo en la mejora del rendimiento de un modelo cuya tarea es la evaluación del habla.

Además es muy interesante el poder ver como evolucionan los resultados según se cambia el conjunto de finos con los que el modelo se entrena. En la tabla 6.1 se puede ver un pequeño resumen de como se ha construido la partición de entrenamiento de cada experimento.

Tabla 6.1: Descripción básica de los experimentos realizados en este documento. **BASE**: Tomaremos como resultado base el obtenido por César en el paper que mando al LREC. **EXP-X-Y**: Indicaremos los diferentes experimentos usando la siguiente nomenclatura: **X** es una cadena formada por la concatenación, **separados por barras bajas (_)**, de las particiones del Common Phone utilizadas para entrenar el reconocedor de fonos. Las particiones posibles son : *ES* (español), *FR* (francés), *IT* (italiano), *DE* (alemán), *EN* (ingles), *RU* (ruso) y *TOT* (todas las particiones). Por otro lado, **Y** es una letra que indica si se ha evaluado sobre todo el corpus PRAUTOCAL (*T*, 3739 audios), o si se ha evaluado solo sobre una subconjunto del mismo (*P*, 1539 audios).

EXP	Nº FONOS VOCABULARIO	Nº FONOS UTILIZADOS
BASE	53	28
EXP-TOT-T	53	28
EXP-ES-P	30	28
EXP-ES-T	30	28
EXP-ES_FR_IT-P	42	28
EXP-ES_FR_IT-T	42	28

Como ya se explico, cada modelo sera evaluado, usando como métrica la correlación lineal de los valores GoP con la clasificación de la lingüista, con cada una de las diferentes variantes de GoP. Una mayor correlación se tomara como un mejor rendimiento, ya que el modelo es capaz de simular mejor el comportamiento de un experto. Los resultados de los diferente experimentos se pueden ver en la tabla 6.2.

Tabla 6.2: Resultados de las correlaciones de los 6 experimentos realizados.

Norm.	GoP	BASE	EXP TOT - T	EXP ES - P	EXP ES - T	EXP ES_FR_IT - P	EXP ES_FR_IT - T
None	GMM	0.311	0.312	0.309	0.325	0.306	0.318
None	NN	0.306	0.332	0.326	0.364	0.313	0.335
Prior	DNN	0.354	0.419	0.363	0.426	0.363	0.417
NONE	Entropy	0.095	0.159	0.091	0.156	0.091	0.150
	Margin	0.335	0.385	0.349	0.401	0.346	0.387
	MaxLogit	0.379	0.443	0.054	0.014	0.105	0.077
	LogitMargin	0.376	0.410	0.364	0.410	0.374	0.414
SCALE	Entropy	0.041	0.093	-0.031	0.021	-0.007	0.044
	Margin	0.366	0.402	0.366	0.412	0.379	0.418
	MaxLogit	0.379	0.443	0.054	0.014	0.105	0.077
	LogitMargin	0.376	0.410	0.364	0.410	0.374	0.414
PRIOR	Entropy	0.096	0.159	0.091	0.156	0.091	0.150
	Margin	0.335	0.385	0.349	0.401	0.346	0.387
	MaxLogit	0.379	**0.443**	0.054	0.014	0.105	0.077
	LogitMargin	0.377	0.410	0.365	0.410	0.375	0.415

Se puede observar como los mejores valores se obtienen para el experimento **EXP-TOT-T**, más concretamente para las variaciones: *MaxLogit*, *ScaleMaxLogit*, *PriorMaxLogit*. Para todas ellas se obtiene un valor de **0.443**. Si obtenemos más decimales vemos que los resultados obtenidos son: **0.443208**, **0.443208** y **0.443424**, respectivamente. Por lo tanto, podemos concluir que *el mejor valor es el de la variación* ***PriorMaxLogit*** *del experimento* ***EXP-TOT-T***.

Algo que puede llamar la atención mirando la tabla es que en ella aparecen valores tanto positivos como negativos. Esto se debe a que la medida de correlación aplicada es *Kendall rank correlation*

coefficient, que proporciona valores entre -1 y 1, siendo 1 una correlación positiva, siendo -1 una correlación inversa y siendo 0 una correlación nula.

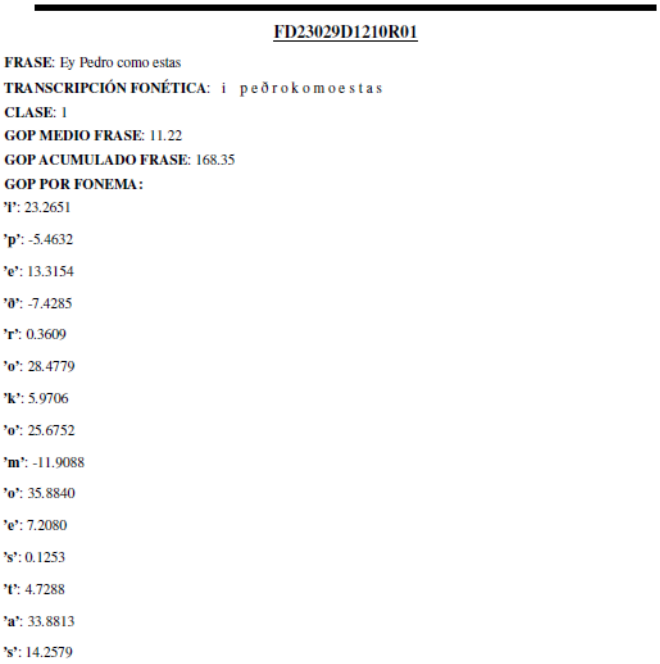
Por lo tanto, el posterior análisis se va a realizar solo sobre el modelo **EXP-TOT-T** y solo sobre las métricas: *MaxLogit*, *ScaleMaxLogit*, *PriorMaxLogit*.

6.5.2 Análisis

Siguiendo con el esquema planteado en la sección 6.4 lo primero que vamos a realizar es el **análisis de frases**

Análisis de frases

Este análisis se va a realizar únicamente sobre los resultados de la mejor variante de GoP, es decir, **PriorMaxLogit**. Este documento nos puede ser muy útil para la clasificación de actividades, es decir, se pueden ver que actividades dan mejores y peores resultados y clasificarlas por dificultad. Esto seria una buena forma de poder elegir que actividad se le debe presentar a cada persona con SD, dependiendo del nivel que tenga. También, usando el identificador del audio para diferenciar los diferente pacientes, se pueden seleccionar todos los ejercicios de un paciente y ver en que fonos tiene mas dificultad, y así poder enfocar de una mejor forma su trabajo. Lamentablemente este documento no ha podido ser analizado en este trabajo. Esto se debe a que la utilidad de este documento esta muy ligada a que el evaluador automático del habla tenga un rendimiento aceptable, cosa que en nuestro caso todavía no sucede.



FD23029D1210R01

FRASE: Ey Pedro como estas
TRANSCRIPCIÓN FONÉTICA: i peðrokomoestas
CLASE: 1
GOP MEDIO FRASE: 11.22
GOP ACUMULADO FRASE: 168.35
GOP POR FONEMA:
'i': 23.2651
'p': -5.4632
'e': 13.3154
'ð': -7.4285
'r': 0.3609
'o': 28.4779
'k': 5.9706
'o': 25.6752
'm': -11.9088
'o': 35.8840
'e': 7.2080
's': 0.1253
't': 4.7288
'a': 33.8813
's': 14.2579

Figura 6.4: Ejemplo de un análisis de una frase

Más ejemplos de este análisis se pueden ver en el apéndice D. El documento entero se puede encontrar en https://gitlab.inf.uva.es/davifer/deep_learning_aplicado_reconocimiento_y_caracterizacion_habla_Down.git.

Análisis de diccionarios

Lo que llama la atención al ver la tabla 6.1 es que el número de fonos del diccionario que genera el programa es siempre mayor que el número real de fonos que utiliza al evaluar PRAUTOCAL. Eso quiere decir que hay muchos fonos en el diccionario del programa que no son utilizados en la evaluación. A continuación mostraremos una comparación entre el diccionario del experimento **EXP-TOT-T** y el diccionario real que se utiliza al evaluar PRAUTOCAL:

Diccionario Utilizado

'i', 'a', 'b', 'e', 'r', 's', 'l', 'n', 'θ', 't', 'o', 'k', 'w', 'p', 'u', 'v', 'ɣ', 'j', 'm', 'd', 'ŋ', 'g', 'ð', 'ʌ', 'f', 'tʃ', 'x', 'ʃ'

Diccionario Programa

'(...)', 'a', 'aɪ', 'aʊ', 'b', 'd', 'dz', 'dʒ', 'e', 'eə', 'eɪ', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n',

'o', 'p', 'r', 's', 't', 'ts', 'tʃ', 'u', 'v', 'w', 'x', 'z', 'æ', 'ð', 'ŋ', 'œ', 'ɔ', 'ɔɪ', 'əʊ', 'ɛ', 'g',

'ɣ', 'ɥ', 'ɨ', 'ɪə', 'ʁ', 'ʃ', 'ʊ', 'ʊə', 'ʌ', 'ʎ', 'ʔ', 'θ'

Esto nos indica que estamos haciendo un esfuerzo inutil, ya que estamos entrenando al modelo con fonos que nunca va a ver. Aun asi, se puede llegar a pensar que esos fonos a los cuales el modelo nunca se enfrenta son utiles y mejoran el rendimiento del modelo, debido a que el experimento que mayor fonos tiene en el diccionario del programa es el que mayor correlación obtiene. Esta conclusion es probablemente errónea, la razon por la cual el experimento **EXP-TOT-T** es el que mejores resultados nos da , es que dicho experimento es el que mas idiomas engloba y aunque englobar mas idiomas supone entrenar con fonos inutiles, tambien supone que existen muchos mas ejemplos de los fonos utiles. Como trabajo futuro queda pendiente el seleccionar solo los fonos utiles del conjunto de entrenamiento de **EXP-TOT-T** y entrenar con ellos un nuevo modelo para verificar esta hipotesis.

Analisis de medias

Una forma de ver si obtenemos el comportamiento deseado por parte del evaluador es realizar la media de los GoP de cada frase de cada clase. De esta forma si apareciese un patrón descendiente/ascendiente seria un indicador de que esta funcionando bien.

Como se ha mencionado antes, el análisis lo vamos a hacer sobre los mejores resultados obtenidos, que son: *MaxLogit*, *ScaleMaxLogit*, *PriorMaxLogit* del experimento **EXP-TOT-T**:

La media del MaxLogit para cada clase (*1_ errfrecuentes*, *2_ erresporadicos*, *3_ correcta*, *4_ td*) es:

- **Media clase 1**: 12.733814328449373
- **Media clase 2**: 13.870971910868636
- **Media clase 3**: 14.55498297167623
- **Media clase 4**: 19.276748397804653

El funcionamiento de esta métrica es muy básico. Para dar un valor de GoP a un fono lo único que hace es la suma de todos los valores de los logits de ese fono entre el número de frames que aparece. Es importante destacar que lo que se usa para esta métrica, son los logits y no las probabilidades.

$$MaxLogit(p) = \frac{1}{F}\sum_{f\in F} L^f(p|f)$$

Vemos que los valores de esta métrica tienen bastante sentido, ya que siguen una tendencia escalada, donde en este caso cuanto mejor sea la pronunciación mayor sera el valor de la métrica.

La media del ScaleMaxLogit para cada clase (*1_ errfrecuentes*, *2_ eresporadicos*, *3_ correcta*, *4_ td*) es:

- **Media clase 1**: 1.6247290798456 28
- **Media clase 2**: 1.7698209574554442
- **Media clase 3**: 1.857095089575909
- **Media clase 4**: 2.4595531925351475

El funcionamiento de esta métrica es igual que el de la anterior, pero en este caso esta escalada por la temperatura, que en el caso del experimento tiene un valor de 7.8375:

$$ScaleMaxLogit(p) = \frac{1}{F}\sum_{f\in F} \frac{L^f(p|f)}{7{,}8375}$$

Se puede observar que, en el ámbito de la media, esta métrica también tiene sentido ya que sigue un patrón de escalada, igual que en el caso anterior.

La media del PriorMaxLogit para cada clase (*1_ errfrecuentes*, *2_ eresporadicos*, *3_ correcta*, *4_ td*) es:

- **Media clase 1**: 11.679227715368002
- **Media clase 2**: 12.815749876217659
- **Media clase 3**: 13.498971614249026
- **Media clase 4**: 18.221078805825943

El funcionamiento de esta métrica es igual que el del primer caso, pero en este caso esta normalizada por e^{prior} de cada clase:

$$PriorMaxLogit(p) = \frac{1}{F}\sum_{f\in F} L^f(p|f) - e^{P(p)}$$

Al igual que las otras dos métricas, esta sigue un buen patrón de escalada que hace que el comportamiento del programa sea el deseado.

A la vista de los resultados se puede decir que las diferentes métricas siguen un comportamiento correcto, aunque seguramente insuficiente. Es decir, aunque la tendencia es correcta, la diferencia entre las medias de cada clase es minúscula y por tanto, muy probablemente no sea suficiente como para obtener un evaluador fiable y correcto. Esto se podrá comprobar con los posteriores análisis (concretamente el análisis de intervalos) que se van a realizar.

Análisis por fonos

En este apartado vamos a mostrar el GoP medio por fono que se obtiene al evaluar todas las frases. Al igual que en el caso anterior, evaluaremos solo para los 3 mejores tipos de Gop (*MaxLogit*,*ScaleMaxLogit* y *PriorMaxLogit*).

Destacar que los símbolos que aparecen representando a los fonos en la siguiente página en algunos casos son una aproximación debido a la incapacidad de representación con Latex. Por otro lado, aclarar que las listas de fonos se encuentran ordenadas de peor a mejor, es decir, el primer fono de cada lista es el peor pronunciado y el último es el mejor pronunciado.

Se puede ver como claramente existe un consenso casi total entre las tres métricas. Esto no debería de sorprender viendo que prácticamente son la misma fórmula, salvo cambiando un par de detalles. Se puede observar como solo difieren en cuatro fonos: '$\int$' , '**l**', '**f**' y '**n**'.

Tabla 6.3: Posición de cada fono en función de la métrica utilizada, posición media según las tres métricas y desviación media.

FONO	**MaxLogit**	**ScaleLogit**	**NormLogit**	**POS. MEDIA**	**DES. MEDIA**
't∫'	1	1	1	1	0.0
'x'	2	2	2	2	0.0
'g'	3	3	3	3	0.0
'w'	4	4	4	4	0.0
'θ'	5	5	5	5	0.0
'$\eth$'	6	6	6	6	0.0
'b'	7	7	7	7	0.0
'γ'	8	8	8	8	0.0
'η'	9	9	9	9	0.0
'j'	10	10	10	10	0.0
'λ'	11	11	11	11	0.0
'd'	12	12	12	12	0.0
'p'	13	13	13	13	0.0
'v'	14	14	14	14	0.0
'u'	15	15	15	15	0.0
'k'	16	16	16	16	0.0
'm'	17	17	17	17	0.0
'r'	18	18	18	18	0.0
't'	19	19	19	19	0.0
'∫'	20	20	21	20	0.33
'l'	21	21	20	21	0.33
'f'	22	22	23	22	0.33
'n'	23	23	22	23	0.33
'o'	24	24	24	24	0.0
'e'	25	25	25	25	0.0
's'	26	26	26	26	0.0
'i'	27	27	27	27	0.0
'a'	28	28	28	28	0.0

Como medida para indicar como de dispares son las clasificaciones de cada métrica entre si se ha decidido usar la desviación media, cuya fórmula se puede ver en la imagen superior. Consiste básicamente en hacer la media de las diferencias absolutas entre las clasificaciones de cada GoP y la media de todas las clasificaciones.

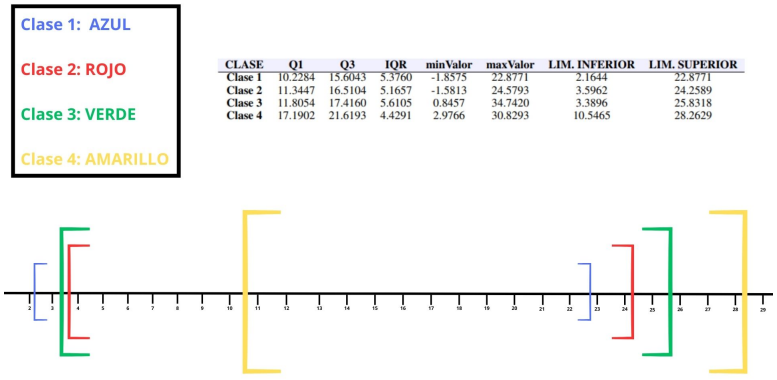
En cuanto a los resultados de la tabla se puede ver como los fonos que difieren en su clasificación difieren todos en una posición. Al ser la mínima diferencia que puede existir, no considero que sea algo a tener en cuenta y creo que se podría decir que las tres métricas coinciden en la clasificación de los fonos.

Por ultimo, se puede observar como los fonos que según el programa se pronuncian mejor (tomando como medida la media de las tres métricas) son los fonos vocálicos, ya que el mejor fono pronunciado es '**a**', el segundo mejor es '**i**', el cuarto mejor es '**e**' y el quinto mejor fono es '**o**'. Esto puede deberse, o bien a que la pronunciación de las vocales es mas sencilla (cosa que se debería de contrastar con un lingüista) y por tanto se realiza correctamente, o bien que el modelo tiene un cierto sesgo que hace dar probabilidades superiores a las vocales.

Análisis de intervalos

Como ya se explico en la sección 6.4, si el programa fuese perfecto lo que haría sería dar unos intervalos para cada clase, donde si un valor del GoP cae en un intervalo se diría que pertenecería a dicha clase. Estos intervalos deberían ser conjuntos totalmente separados y continuos, es decir, donde uno acaba debería empezar el siguiente, de esta forma cualquier ejemplo podría ser clasificado en una clase. Por otro lado, no debería existir ninguna intersección entre ellos, porque si las hubiese no se sabría como clasificar el ejemplo que cayese dentro de ellos.

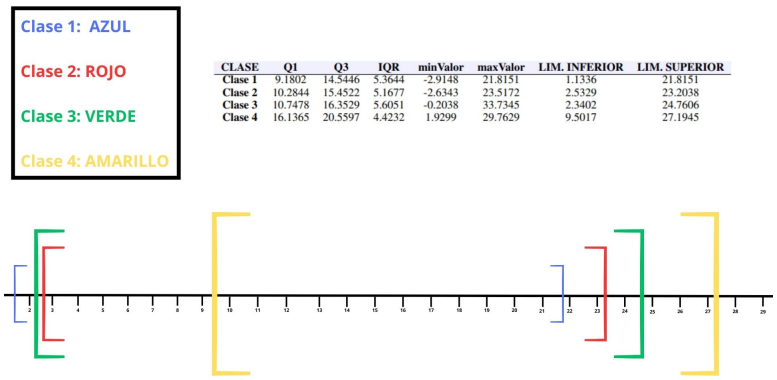
MaxLogit == ScaleMaxLogit



CLASE	Q1	Q3	IQR	minValor	maxValor	LIM. INFERIOR	LIM. SUPERIOR
Clase 1	10.2284	15.6043	5.3760	-1.8575	22.8771	2.1644	22.8771
Clase 2	11.3447	16.5104	5.1657	-1.5813	24.5793	3.5962	24.2589
Clase 3	11.8054	17.4160	5.6105	0.8457	34.7420	3.3896	25.8318
Clase 4	17.1902	21.6193	4.4291	2.9766	30.8293	10.5465	28.2629

Como se puede observar la representación de *MaxLogit* y *ScaleMaxLogit* es la misma, esto se debe a que lo único que cambia entre ellas es la escala, y por tanto, la distribución de los intervalos es la misma.

PriorMaxLogit



CLASE	Q1	Q3	IQR	minValor	maxValor	LIM. INFERIOR	LIM. SUPERIOR
Clase 1	9.1802	14.5446	5.3644	-2.9148	21.8151	1.1336	21.8151
Clase 2	10.2844	15.4522	5.1677	-2.6343	23.5172	2.5329	23.2038
Clase 3	10.7478	16.3529	5.6051	-0.2038	33.7345	2.3402	24.7606
Clase 4	16.1365	20.5597	4.4232	1.9299	29.7629	9.5017	27.1945

Si nos fijamos en los límites de los intervalos de ambos casos nos percataremos que todas las clases contienen valores atípicos tanto por encima como por debajo, salvo la clase 1 que no tiene valores atípicos por encima.

Se puede ver que ambas representaciones son prácticamente idénticas. Podemos observar en ambas representaciones que la premisa de: «intervalos sin intersecciones y continuos», no se cumple en absoluto. Todos los intervalos están mezclados entre si, dando lugar a un caos de clasificación. Algo positivo, y que es reflejo de que la media siga un patrón de escalada, es que los limites siguen una tendencia creciente, salvo para el caso del límite inferior de la clase 2 y 3.

Claramente, la clase que mejor clasifica es la clase 4, donde se puede ver como su intervalo es el que esta más distanciado del resto. Esto es normal, dado que el reconocedor de fonos, ha sido entrenado para reconocer hablantes típicos que son los que componen la clase 4.

A continuación vamos finalizar el análisis de los intervalos calculando la métrica FCU, que ya se explico en la sección 6.4.

GoP	CLASE	FCU	NO FCU	Precisión (%)
MaxLogit	Clase 1	7	571	1.21
	Clase 2	0	755	0.0
	Clase 3	0	816	0.0
	Clase 4	32	1557	2.01
	TOTAL	39	3699	1.05
PriorMaxLogit	Clase 1	3	575	0.51
	Clase 2	0	755	0.0
	Clase 3	0	816	0.0
	Clase 4	67	1522	4.21
	TOTAL	74	3699	2.00

Tabla 6.4: Análisis de los resultados GoP obtenidos usando la métrica FCU

Se puede observar como tanto la métrica de *correlación* como la *precisión FCU* dan como mejor métrica a la *PriorMaxLogit*. En el caso de la *precisión FCU* es prácticamente un 100 % mejor *PriorMaxLogit* que *MaxLogit* (no se compara con *ScaleMaxLogit* porque es igual que *MaxLogit*).

Aun así, se puede ver claramente como los resultados son bastante malos. Que los intervalos estén totalmente entremezclados ocasiona que las clase 2 y 3 tengan un 0 % de *precisión FCU*. Se puede ver claramente como solo hay dos zonas en las cuales si un ejemplo cae en ellas se clasificará a una sola clase: la primera es el espacio entre el límite inferior de la clase 1 (azul) y el límite inferior de la clase 3 (verde), y la segunda es el espacio entre el límite superior de la clase 3 (verde) y el limite superior de la clase 4 (amarillo). En el primer caso los ejemplos se clasificarán como clase 1 y en el segundo como clase 4.

6.6 Conclusiones

El experimento realizado nos muestra que el planteamiento actual es todavía bastante malo. Los resultados obtenidos nos dicen que el modelo tiene un desempeño muy malo, y para nada usable en una situación real. El modelo no es capaz de diferenciar entre los 4 niveles que se le plantean. Aunque viendo la correlación aparentemente parezca que el modelo tiene cierta idea, cuando nos introducimos en el análisis y vemos los resultados del mismo se ve muy claramente como el desempeño del modelo es muy malo. Aun así, merece la pena seguir explorando esta vía, dado que con una mayor dedicación a ella parece que se podrían obtener buenos resultados.

Por otro lado, se puede apreciar como no se ha conseguido hacer que el modelo funcione como un evaluador, y por ahora funciona solo como un reconocedor. Esto se puede apreciar muy claramente a lo largo de todo el análisis, donde se ve que cuando mejor funciona el modelo es cuando la locución es de una persona típica. Esto es algo normal debido al entrenamiento al que ha sido sometido, que ha sido exclusivamente con locuciones típicas.

Por último, se puede concluir que este primer acercamiento ha sido exitoso metódicamente hablando, ya que se ha establecido una muy buena base y un muy buen procedimiento para seguir explorando este camino. Esto facilitará mucho los futuros experimentos y las futuras modificaciones, haciendo mucho más simple el trabajo.

6.6.1 Trabajo futuro

En cuanto al trabajo futuro queda pendiente seguir explorando distintas vias de entrenamiento, como por ejemplo coger de todos los idiomas el conjunto de fonos que el modelo usa al evaluar PRAUTOCAL y entrenarlo con ese conjunto. O por ejemplo, probar a aumentar el corpus de entrenamiento añadiendo otros corpus en castellano que se encuentren segmentados como podría ser Albayzin [30].

Por otro lado, queda pendiente el plantearse nuevas arquitecturas o modificaciones del modelo actual, que puedan ser beneficiosas de cara a la evaluación del habla. En lo que respecta a modificaciones también queda pendiente el plantearse una variante de la métrica GoP que sea más apropiada para la tarea que queremos realizar

Appendices

Apéndice A

Manual de Instalación

Lo primero que debemos hacer es clonar este repositorio en su directorio de trabajo, para ello:

```
git clone https://gitlab.inf.uva.es/davifer/
    deep_learning_aplicado_reconocimiento_y_caracterizacion_habla_Down.git
```

Una vez hecho esto debemos dirigirnos a la carpeta instalación. Alli encontraremos un script llamado "build" que debemos ejecutar:

```
./build
```

Con esto lo que vamos a crear es un docker, donde ya se encuentran instaladas todas las librerías y todos los recursos necesarios para hacer funcionar el código de una manera correcta.

Para poder ejecutar el docker tenemos dos opciones:

- **Ejecutar el docker directamente y trabajar sobre su terminal**: Para ello solo deberemos escribir "dowhisper.[en] la linea de comandos y nos introduciremos en el propio docker.
- **Lanzar trabajo al docker**: Para ello utilizaremos el mismo comando que antes, pero esta vez le añadiremos la orden que queremos ejecutar dentro del docker:

  ```
  dowhisper 'python3 TFG/pruebas/pruebas_whisper/pruebas.py'
  ```

Con esto ya podremos ejecutar cualquier script de este repositorio. Dentro de cada carpeta (cuyo contenido se explicara a continuación) se encuentra un fichero run"donde se pueden encontrar todas las ordenes de ejecución que se pueden hacer en la carpeta. Basta con descomentar la que queremos ejecutar y comentar el resto y simplemente ejecutando ese archivo podremos ejecutar el script

```
#!/bin/bash

# Ejecutar fine-tuning sobre un corpus
dowhisper 'python3 TFG/pruebas/fine-tuning/python/fine-tuning.py TFG/
    pruebas/fine-tuning/args_ft.txt'

# Busqueda de hiper-parametros
#dowhisper 'python3 TFG/pruebas/fine-tuning/python/ajusteHiperparametros.
    py TFG/pruebas/fine-tuning/args_hp.txt'
```

Apéndice B

Código fuente de los experimentos

El código fuente utilizado en los experimentos de est trabajo se puede encontrar en el siguiente repositorio: https://gitlab.inf.uva.es/davifer/deep_learning_aplicado_reconocimiento_y_caracterizacion_habla_Down.git

En dicho repositorio podemos encontrar lo siguiente:

- **pruebas_whisper**: En esta carpeta se encuentra el código para poder probar el modelo Whisper sobre un corpus determinado y poder ver así su rendimiento. Los resultados del experimento se guardan en *mediciones.log* (medidas básicas de tiempo y memoria y WER), *resultados.csv* (medidas más detalladas de tiempo y memoria) y en *trans-pred.csv* (comparación entre transcripción y predicción). Para ejecutarlo hay que ejecutar el archivo *evalWhisper.py* el cual coge sus argumentos del archivo *args.txt*, donde se debe especificar los siguiente:

  - Nombre del dataset con el que se quiere evaluar el modelo.
  - Partición que se desea usar, en caso de ser un corpus propio hay que escribir *OWN* en este campo.
  - Conjunto que se quiere utilizar para la prueba (*train*, *validation* o *test*).
  - Tipo del modelo a usar. Si el modelo es propio hay que especificarlo con *OWN* en caso contrario da igual el input de este campo.
  - Idioma en el que se va a utilizar el modelo, en nuestro caso *spanish*.

  Para ejecutar este script debemos ir al script "args.txt" allí escribir los argumentos necesarios y por último ejecutar el siguiente comando:

  ```
  sh run
  ```

- **fine_tuning**: En esta carpeta se encuentra el código para poder realizar búsqueda de hiperparámetros y fine-tuning. Existen dos scripts ejecutables, *ajusteHiperparametros.py* y *fine-tuning.py*. Los argumentos para cada programa se deben introducir en *args_hp.txt* y *args_ft.txt* respectivamente. Los argumentos que se deben introducir en *args_hp.txt* son los siguientes:

  - Nombre de corpus a utilizar para la búsqueda.
  - Partición que se desea usar, en caso de ser un corpus propio hay que escribir *OWN* en este campo.
  - Nombre del modelo a utilizar

- Nombre del Sweep que se creara en WANDB (plataforma que se usa para visualizar la ejecución y los resultados de la busqueda)

En el caso de *args_ft.txt* los argumentos son los mismo salvo por el último que en vez de ser el nombre del Sweep es el nombre del nuevo modelo que se genera al realizar el fine-tuning.

Para ejecutar uno de estos script debemos, primero decidir que script queremos ejecutar e ir al archivo runz descomentar la linea que ejecute dicho script. Posteriormente debemos ir al archivo de argumentos correspondiente (*args_hp.txt* o *args_ft.txt*) y redactar allí los argumentos que queremos en nuestra ejecución. Finalmente debemos ejecutar:

```
sh run
```

- **data_augmentation**: En esta carpeta se encuentra el código para realizar aumento de datos sobre un corpus. El script que hay que ejecutar es **data_augmentated.py**, marcando las opciones correspondientes dependiendo de las técnicas que se deseen usar.

  Para ejecutar uno de estos script debemos, primero decidir que argumentos queremos usar e ir al archivo "run" y escribir allí los argumentos que queramos. Una vez ya hemos definido el experimento tan solo debemos ejecutar:

```
sh run
```

- **GoP**: En esta carpeta podemos encontrar los scripts referentes a la parte de Calidad del Habla. En ella podemos encontrar los scripts correspondientes al repositorio https://github.com/juice500ml/dysarthria-gop, que es el repositorio se ha empleado para esta sección. Además de dichos scripts podemos encontrar uno nuevo denominado *leerDatos.py*, cuya función es obtener los datos y las representaciones usadas en la parte de análisis del estudio realizado.

  - *dataset.py*: Este script se usa para obtener los ficheros *.csv* con el formato correcto de los corpora que deseemos emplear.
  - *gop.py*: En este script se encuentran las diferentes implementaciones de GoP que se utilizan.
  - *model.py*: Este script se encarga de cargar el modelo y hacerle las modificaciones necesarias para su uso.
  - *loss.py*: Aquí se postulan las diferentes funciones de perdida que se pueden usar en el entrenamiento.
  - *loss.py*: Aquí se postulan las diferentes funciones de perdida que se pueden usar en el entrenamiento.
  - *temperature_scaling.py*: Este script se encarga de preparar todo lo necesario para poder modificar la temperatura del modelo.
  - *train_phone_recognizer.py*: Este es el script principal que usa todos los anteriores para entrenadar el modelo y obtener así un reconocedor de fonos que se emplee para evaluar el habla.
  - *leerDatos.py*: Este script analiza los resultados de la evaluación de una serie de audios para obtener así un conjunto de representaciones y métricas que sirven para profundizar más en los resultados de la evaluación.

  Para ejecutar uno de estos script debemos, primero decidir que argumentos queremos usar e ir al archivo "run" y escribir allí los argumentos que queramos. Una vez ya hemos definido el experimento tan solo debemos ejecutar:

```
sh run
```

Apéndice C

Artículo aceptado en el SEPLN

Adaptación de ASR al habla de personas con síndrome de Down

ASR model adaptation to the speech of people with Down syndrome

David Fernández-García,[1] **Valentín Cardeñoso-Payo,**[1]
César González-Ferreras,[1] **David Escudero-Mancebo**[1]
[1]Grupo de Investigación ECA-SIMM, Universidad de Valladolid, España
david.fernandez@estudiantes.uva.es,
{valentin.cardenoso, cesargf, escuderomancebo.david}@uva.es

Resumen: El habla de las personas con discapacidad intelectual (DI) plantea enormes retos a los sistemas de reconocimiento automático del habla (ASR), dificultando con ello el acceso de una población especialmente sensible a los servicios de información. En este trabajo se estudian las dificultades de los sistemas ASR para reconocer habla de personas DI y se muestra cómo esta limitación puede ser combatida con estrategias de ajuste fino de modelos. Se mide el rendimiento de ASR basado en *whisper* (v2 y v3) con un corpus de referencia de habla típica y habla DI, comprobando que hay diferencias importantes y significativas. Aplicando técnicas de *fine-tuning*, el rendimiento para hablantes DI mejora en al menos 30 puntos porcentuales. Nuestros resultados muestran que la inclusión de voz de personas DI en los corpus de entrenamiento es fundamental para mejorar la eficacia de los ASR.
Palabras clave: ASR, Habla anómala, *whisper*, Aumento de datos.

Abstract: The speech of people with intellectual disabilities (ID) poses enormous challenges to automatic speech recognition (ASR) systems, making it difficult for a particularly sensitive population to access information services. This work studies the difficulties of ASR systems in recognizing the speech of ID people and shows how this limitation can be combated with model fine-tuning strategies. The performance of ASR based on *whisper* (v2 and v3) is measured with a reference corpus of typical speech and DI speech, verifying that there are important and significant differences. By applying *fine-tuning* techniques, performance for DI speakers improves by at least 30 percentage points. Our results show that the inclusion of the voice of ID people in the training corpora is essential to improve the effectiveness of ASRs.
Keywords: ASR, Pathologic Speech, *whisper*, Data Augmentation.

1 Introducción

La calidad de los sistemas de reconocimiento automático de habla ha aumentado significativamente en los últimos años, lo que ha mejorado sustancialmente la accesibilidad de las aplicaciones por medio de interfaces habladas. Sin embargo, es ya conocido que su eficacia y precisión pueden variar significativamente entre diferentes grupos demográficos y condiciones de habla (Lea et al., 2023; Cibrian et al., 2024; De Russis y Corno, 2019). En concreto, su aplicación a poblaciones específicas como las personas con síndrome de Down, presenta una serie de dificultades, asociadas tanto a las limitaciones físicas como cognitivas de este tipo de personas (Hu et al., 2013).

El síndrome de Down (SD), según el nuevo sistema de clasificación de diagnóstico DSM-5 (American Psychiatric Association, 2013), es un subtipo de trastorno del desarrollo intelectual caracterizado por importantes limitaciones tanto en el funcionamiento intelectual (con un coeficiente intelectual igual o inferior a 70, con dificultades cognitivas) como en la conducta adaptativa. Las personas con SD muestran dificultades en las habilidades adaptativas conceptuales, sociales y prácticas. Aunque existe una gran heterogeneidad dentro de este colectivo, generalmente presentan también importantes dificultades en sus habilidades lingüísticas.

Aunque todas las áreas del lenguaje están afectadas en distinto grado, las habilidades de producción de lenguaje suelen estar más deterioradas que las habilidades de comprensión (Martin et al., 2009). Las personas con SD pueden mostrar problemas de articulación, y en algunos casos, su habla es casi ininteligible. La inteligibilidad del habla se ve seriamente afectada por la presencia de errores en la producción de algunos fonemas, la pérdida de consonantes y la simplificación de sílabas (Laws y Bishop, 2004; Wong et al., 2015).

Las personas con SD a menudo afrontan importantes limitaciones en sus relaciones sociales debido a un manejo deficiente de la comunicación oral (Cleland et al., 2010; Martin et al., 2009; Chapman, 1997). Es fundamental comprender estas dificultades para proporcionar un apoyo adecuado y fomentar la inclusión en la sociedad, especialmente considerando que las tecnologías de la información y la comunicación (TIC) son parte integral de nuestras actividades diarias, incluyendo a las personas con discapacidad intelectual (Tanis et al., 2012; Feng et al., 2010). Por ejemplo, las redes sociales, una de las herramientas de TIC más utilizadas, también son frecuentemente utilizadas por personas con discapacidad intelectual (Caton y Chapman, 2016).

Aunque estudios relativamente recientes muestran que menos del 4 % de los usuarios con SD usan sistemas de entrada vocal como modo de interacción con los dispositivos digitales (Feng et al., 2010), el reconocimiento automático del habla tiene el potencial de hacer que la tecnología sea más accesible para los usuarios, especialmente para los que, como éstos, pueden tener problemas motores que merman la destreza a la hora de manejar el teclado o el ratón (Hu et al., 2013).

Sin embargo, los sistemas de reconocimiento del habla actuales no proporcionan resultados de la misma calidad para personas con SD, en comparación con aquellas con desarrollo típico (Cibrian et al., 2024). Por ello, y en línea con trabajos anteriores (Shor et al., 2019; Green et al., 2021), la solución adoptada en este trabajo es la adaptación de los sistemas de reconocimiento de habla independientes de locutor de última generación para garantizar una mayor precisión del reconocedor y, con ella, una mejora de la accesibilidad.

En este trabajo analizaremos las estrategias de adaptación, inspirados en el trabajo de Tobin y Tomanek (2022), que maneja también datasets de tamaño reducido para realizar *fine-tuning* de modelos de reconocimiento, y cómo aquellas pueden mejorar la tasa de reconocimiento para este grupo de usuarios con SD.

El artículo comienza revisando el estado del arte sobre el reconocimiento de habla de personas con SD. En la sección tres se describen los corpus utilizados para el entrenamiento y la evaluación del modelo, tanto de habla Down como de habla típica. En la sección cuatro describimos la metodología empleada, incluyendo las técnicas de aumento de datos y de adaptación de los modelos *whisper* (Radford et al., 2023), que se han tomado como referencia al ser los de mejor rendimiento conocido en habla típica en el momento de escribir este artículo (Cibrian et al., 2024). En la sección cinco se analizan los resultados obtenidos, comparando el rendimiento del modelo en ambos tipos de habla, y se evalúa el impacto de las técnicas de adaptación y aumento de datos. El artículo finaliza con un resumen de las principales conclusiones y se proponen líneas de trabajo futuro para mejorar el ASR en personas con SD.

2 Estado del arte

El reconocimiento automático del habla para el habla patológica es un área de investigación en crecimiento. Varios estudios han resaltado la importancia del ASR para ayudar a individuos con trastornos del habla (Rosen y Yampolsky, 2000; Kitzing, Maier, y Åhlander, 2009; Schultz et al., 2021). Uno de los trastornos del habla más estudiados es la disartria (Almadhor et al., 2023; Janbakhshi, Kodrasi, y Bourlard, 2021; Jiao et al., 2018; Shahamiri, 2021; Bhat y Strik, 2020). Los estudios han explorado el uso de sistemas ASR para reconocer patrones de habla disártrica, con un enfoque centrado en el análisis de los factores que afectan el rendimiento del sistema. El estado actual del ASR para el habla patológica, en particular la disartria, está avanzando rápidamente, ofreciendo soluciones innovadoras para ayudar a individuos con trastornos del habla en sus procesos de evaluación y terapia. La integración de la tecnología ASR tiene un gran potencial para mejorar las vidas de aquellos afectados por condiciones de habla patológica.

El habla de los individuos con SD no es necesariamente disártrica (Kumin, 2012), pe-

ro en general no es un habla típica, debido a problemas relacionados con el tono muscular bajo, el tamaño grande de la lengua y las frecuentes infecciones del oído. Estos individuos pueden experimentar habitualmente retrasos en el habla y dificultades en la articulación cabe destacar que cada persona con SD es única, y los problemas de habla pueden variar ampliamente entre individuos (Kumin, 2012).

La disponibilidad de corpus de entrenamiento en ocasiones es un obstáculo para la investigación y el desarrollo de estos sistemas, porque recopilar este tipo de corpus es un proceso costoso al que hay que dedicar gran cantidad de recursos. En el marco del proyecto Euphonia,[1]desarrollado por Google Research, se ha recopilado un corpus de habla de personas con trastornos del habla en inglés. Se trata del corpus más grande de este tipo de hablantes grabado hasta la fecha, puesto que han participado más de 1000 personas y se han grabado más de 1 millón de locuciones, lo que supone más de 1300 horas. El corpus incluye grabaciones de 105 personas con SD, lo que supone el 18,1 % del corpus (MacDonald et al., 2021). Empleando este corpus se han realizado diversos experimentos usando diferentes técnicas de deep learning para clasificar la inteligibilidad del habla de 661 locutores con diversas patologías, incluyendo SD (Venugopalan et al., 2021). Por otro lado, para mejorar el rendimiento de los sistemas de reconocimiento automático del habla en hablantes con trastornos del habla, se ha propuesto la personalización de modelos (Tomanek et al., 2021; Green et al., 2021), como vía para evitar la degradación significativa del rendimiento en habla patológica que se produce en los actuales sistemas de reconocimiento automático de habla.

Otro problema que suele afectar a este tipo de habla son las disfluencias y las variaciones en la pronunciación del habla, que pueden degradar gravemente el rendimiento del reconocimiento de habla, ya que los sistemas actuales de ASR se entrenan principalmente con habla fluida de hablantes típicos. Un enfoque sencillo para mejorar el rendimiento es ajustar los parámetros de decodificación en un sistema de reconocimiento de habla existente, lo que puede mejorar la tasa de errores de palabras (WER) para personas con trastornos de fluidez (Mitra et al., 2021).

[1]https://sites.research.google/euphonia/about/

3 Corpora

Para el desarrollo del trabajo se han usado tres corpora: FLEURS (Conneau et al., 2023) como corpus de referencia de habla típica para ajustar los hiperparámetros de *fine-tuning*, PRAUTOCAL Down (Escudero-Mancebo et al., 2022) como corpus de referencia de habla Down, para uso en *fine-tuning* y evaluación, y PRAUTOCAL Típico y VoxPopuli (Wang et al., 2021) como corpora de evaluación de habla típica. Las principales características de estos corpus se pueden ver en la Tabla 1.

3.1 Corpus de habla Down

El corpus PRAUTOCAL (Escudero-Mancebo et al., 2022) es un corpus de hablantes de español con SD del norte/centro peninsular que permite el análisis de aspectos específicos del habla de las personas con SD. También incluye grabaciones comparables de usuarios con desarrollo típico (DT) que sirven de referencia. El corpus se ha construido grabando interacciones con un videojuego para entrenar las competencias orales de las personas con síndrome de Down. El corpus se recopiló en seis campañas de grabación y contiene 90 locutores, con 4175 ficheros de audio distribuidos en 40 actividades diferentes asociadas al desarrollo de un videojuego educativo supervisado por terapeutas.

Para garantizar una representación homogénea del texto tanto en las referencias como en las predicciones de los modelos, ha sido necesario realizar un pre-procesamiento del corpus PRAUTOCAL Down para obtener una representación común de las transcripciones.

El etiquetado del corpus PRAUTOCAL Down contiene ciertas marcas indicativas de disfluencias que deben ser eliminadas para obtener una referencia limpia y precisa. Las marcas que nos podemos encontrar en las transcripciones de las frases del corpus son:

- Términos que han sido marcados entre los símbolos < y > indican que se ha producido una disfluencia.
- Palabras precedidas por el símbolo #, lo cual indica que dicha palabra es un *filler*, es decir, una palabra de relleno.
- Signos de puntuación (puntos y comas) que pierden su significado ortográfico y pasan a referirse a pausas que el hablante ha hecho durante la locución.

Corpus	Tipo Habla	Nº Hablantes	Nº Horas	Procedencia
FLEURS	Típica	3	12	Art. de Wikipedia
VoxPopuli	Típica	305	166	I. Parlamentarias
PRAUTOCAL Típico	Típica	30	2	Int. con Videojuego
PRAUTOCAL Down	Anómala	42	2	Int. con Videojuego

Tabla 1: Tabla resumen de las principales características de cada corpus utilizado. Los datos de todos los corpus son de la partición en español.

Además de los anteriormente citados, también se ha eliminado cualquier otro signo de puntuación, todas las tildes (debido a que el corpus no estaba correctamente etiquetado en ese sentido) y todos los signos ortográficos, como guiones, corchetes, ..., que pudiesen aparecer. Esta misma normalización ha sido también aplicada a las predicciones de *whisper*, en la evaluación con este corpus, para poder realizar una comparación justa y precisa.

Se han eliminado 67 ficheros de audio del corpus PRAUTOCAL Down que se determinó que contienen pseudo-habla (ruidos sin sentido).

Por último, se ha decidido añadir un segundo de silencio al comienzo y al de cada fichero de audio del corpus, lo que se comprobó que mejoraba el rendimiento de los reconocedores en esas frases, como ya se detectó en el trabajo de (Prananta et al., 2022).

3.2 Corpus de habla típica

En lo que respecta a los corpus FLEURS y VoxPopuli, simplemente se les ha aplicado el pre-procesamiento básico que el propio modelo *whisper* proporciona. Este procesamiento esta enfocado al trabajo plurilingüe y se basa en eliminar todo tipo de signos ortográficos y tildes, manteniendo la *ñ*.

Para el caso especial de PRAUTOCAL Típico, se le ha aplicado la misma normalización que a PRAUTOCAL Down, debido a que, aunque carece de anotaciones, incluye numerosos errores en la ubicación de los signos de puntuación (interrogaciones y exclamaciones) y de las tildes.

3.3 Eliminación de frases en bucle

Como bien se cita en Radford et al. (2023), el modelo *whisper* tiene un defecto que hace que, en ocasiones, el proceso de predicción entre en un bucle infinito (que solo termina al alcanzar el número máximo de caracteres permitidos en la generación) donde solo escribe una y otra vez la misma palabra, letra o conjunto de palabras. El propio artículo documenta que se espera que dicho comportamiento desaparezca al hacer un *fine-tuning* del modelo. En toda la experimentación que hemos realizado nos hemos topado con este problema, tanto con el modelo base, como en la gran mayoría de los *fine-tuning* que hemos realizado. El problema con estas frases es que normalmente son muy influyentes en el WER del modelo, debido a que generan una gran cantidad de inserciones que disparan su número de errores, y por tanto, el WER general. En resumen, se ha procedido a eliminar las que denominaremos "frases en bucle".

Para cada experimento de los realizados se ha creado un nuevo corpus denominado PRAUTOCAL Down Clean, a partir del PRAUTOCAL Down, pero que no incluya los ficheros de audio que generan predicciones en bucle, que pueden ser distintos para cada modelo generado. Mantener dos versiones distintas del corpus PRAUTOCAL Down por modelo, y no del resto de corpora, se debe a que este fenómeno se ve muy intensificado en este corpus, generando variaciones de WER de hasta un 20 %. En el resto de corpora el fenómeno sigue apareciendo pero en menor medida, por eso se ha decidido experimentar sólo con una versión en estos casos, que no contiene las "frases en bucle" para cada modelo.

El fenómeno de que un modelo *whisper* repita indefinidamente palabras o fragmentos de texto ante determinadas entradas de voz puede atribuirse a varios factores: limitaciones del modelo, complejidad del contexto, problemas de entrenamiento o bucles de atención. En última instancia, el fenómeno de repetición indefinida parece provenir de la complejidad inherente de la generación de lenguaje natural y las limitaciones actuales de los modelos de inteligencia artificial en este ámbito.

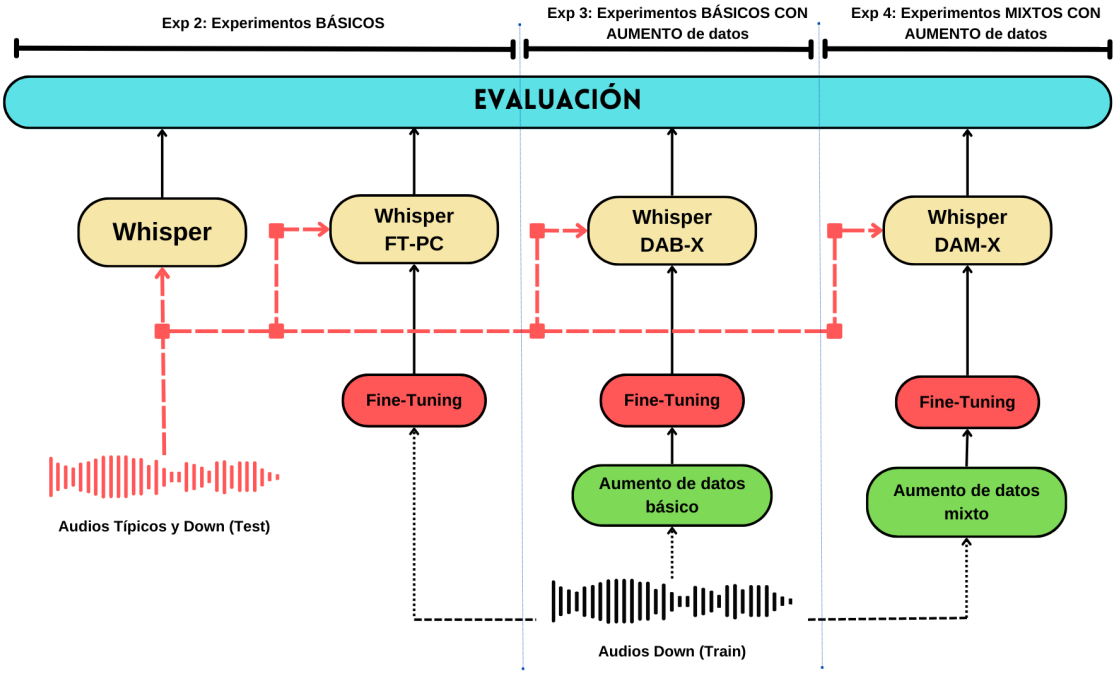


Figura 1: Diagrama explicativo de la metodología seguida en el artículo.

4 *Metodología*

En este artículo se va a usar el modelo *whisper* para realizar la experimentación, empleando el modelo de tamaño *large*. La elección de dicho modelo se debe a los buenos resultados que ofrece para español de habla típica, ya que es un modelo que ya se ha contrastado en varias ocasiones en tareas de habla anómala (Cibrian et al., 2024).

El flujo conceptual de los experimentos realizados puede verse en la Figura 1.

4.1 Adaptación de modelos

Previo a la realización de los diversos *fine-tuning*, se ha hecho un ajuste de los hiper-parámetros. Para ello, se ha utilizado el corpus FLEURS, ya que el corpus de habla Down del que se dispone es demasiado pequeño como para ser empleado en esta tarea y se podrían propiciar situaciones de *overfitting*. La búsqueda de hiper-parámetros se ha realizado siguiendo la metodología estándar: establecer un valor para cada hiper-parámetro a optimizar, posteriormente realizar un *fine-tuning* con dichos valores y con la partición *train* y evaluar el resultado con la partición *validation*. Para cada parámetro se han probado valores dentro de un intervalo inicialmente muy amplio que posteriormente se ha ido estrechando hasta alcanzar los valores óptimos: *learning rate: 2.155e-5*, *weight decay: 3.168e-4*, *epochs: 1*, usando siempre formato de coma flotante de precisión alta. La métrica utilizada para decidir que valores de los hiper-parámetros eran mejores ha sido la métrica WER.

4.1.1 Partición de datos

Debido principalmente al tamaño reducido del corpus PRAUTOCAL Down, se ha optado por un esquema de partición 60/40: 60 % de los ficheros de audio para *train* y 40 % para *test*, obviando la partición de validación. El porcentaje de muestras de *train* se ha elegido teniendo en cuenta que se aplicarán técnicas de aumento de datos sobre el corpus que harán que el conjunto de entrenamiento crezca considerablemente, lo que podría generar que la partición *test* tuviera un tamaño muy reducido en comparación a la de *train* si se optase por otros esquemas más tradicionales. Siguiendo el esquema anterior de partición, se han realizado dos divisiones diferentes, que se describen a continuación.

La **división por actividad** consiste en separar en *train* y *test* las diferentes actividades que contiene el corpus PRAUTOCAL. Todas las locuciones del 60 % de las 40 actividades diferentes que contiene el corpus irán al conjunto *train* y el 40 % restante al de *test*. Entendemos por actividad un tipo de frase pronunciada por cualquier locutor. Para cada actividad y locutor existen diferentes locu-

ciones. Realizando esta división conseguimos que ninguna partición tenga frases en común. De esta manera, una vez realizado el *fine-tuning*, al evaluar el modelo con el conjunto *test*, se enfrentará a frases nunca vistas, lo que nos permitirá observar cuál es su verdadero aprendizaje de las características propias del habla Down, al ser un modelo independiente de locutor y de tarea.

La **división aleatoria** consiste en distribuir directamente las locuciones entre *train* y *test* siguiendo los porcentajes 60-40 %, sin tener en cuenta la actividad. Con esta división es muy probable que algunas frases coincidan en ambos conjuntos, y por tanto, es posible que el modelo no solo aprenda las características intrínsecas del habla Down, sino también la pronunciación de esas frases particulares, resultando sólo independiente de locutor pero no de tarea.

4.1.2 Aumento de datos

La escasez de datos de habla anómala es un problema con el que hay que lidiar diariamente en la construcción de modelos de ASR. Con el fin de combatir esta limitación, una de las técnicas mas utilizadas es el aumento datos (Hermann y Magimai.-Doss, 2023), mediante técnicas como conversión de voz, modificación de audio, generación de voz vía TTS, ... Este problema también se manifiesta en nuestros experimentos y por eso vamos a aplicar técnicas de modificación de audio básicas sobre los ficheros de audio de PRAUTOCAL Down, como una primera aproximación al aumento de datos de habla Down.

- **Variación de la velocidad**: Esta técnica consiste en variar la velocidad del audio original, obteniendo así un nuevo audio un poco distinto. Conseguimos así simular cambios en el ritmo y en la velocidad del habla, generando de esta forma nuevos "pseudo-hablantes". Estos nuevos "pseudo-hablantes" pueden ayudar al modelo a generalizar mejor para nuevos ritmos y velocidades a los cuales el modelo no esta acostumbrado. La modificación no debe ser muy grande, dado que sino el modelo estaría aprendiendo ritmos y velocidades irreales con los cuales nunca se va a encontrar. Es por ello que la velocidad del audio se variará a 0.9 y 1.1 empleando la biblioteca *wave*.[2]
- **Introducción de ruido**: Esta técnica consiste en añadir ruido de diferente naturaleza al audio original. Se han considerado 2 tipos de ruido. El primero es el ruido blanco cuya principal característica es que tiene una densidad espectral de potencia constante. Esto significa que el ruido contiene todas las frecuencias y todas ellas tienen la misma potencia. El segundo tipo es el ruido de color, el cual no tiene una densidad espectral constante, sino que dependiendo de la forma que tenga la onda se clasificará como ruido de un color o de otro. Esta nueva modificación puede ayudar al modelo a trabajar con ficheros de audio cuya calidad no sea del todo perfecta o que contengan sonidos de fondo que puedan distorsionar el rendimiento del modelo. Dado la gran cantidad de tipos de ruidos que existen se ha decidido utilizar solo dos tipos, el ruido blanco y el ruido rosa. La elección del primero se debe a sus características únicas explicadas anteriormente. En cuanto al segundo, se ha decidido elegir el ruido de color rosa debido a que es el más frecuente en experimentos de este estilo. Para generar los ruidos de color rosa se ha utilizado el algoritmo propuesto en Timmer y Koenig (1995).
- **Variación del tono**: Esta técnica consiste en variar el tono (*pitch*) del audio original obteniendo así nuevos ficheros de audio ligeramente distintos a los originales. La aplicación de esta técnica genera nuevos "pseudo-hablantes" que pueden facilitar la generalización al modelo. En la implementación de esta técnica se va a subir/bajar un cierto número de semitonos al audio original. La variación no debe ser muy grande, por lo que las modificaciones que se van a realizar van a consistir en subir/bajar 2 semitonos. Se ha utilizado la función *pitch_shift()* de la biblioteca *librosa*.[3]

Las técnicas descritas se van a aplicar sobre el corpus PRAUTOCAL Down, tanto de una en una como en grupos de dos, tres y cuatro, con el objetivo de aumentar los pocos datos de los que se disponen. Al incrementar el número de ficheros de audio se espera obtener una mejoría en el rendimiento del modelo, debido a que al aumentar la cantidad de

[2] https://docs.python.org/3/library/wave.html

[3] https://librosa.org/doc/latest/index.html

datos, el modelo dispone de más muestras para aprender las características intrínsecas de las mismas, lo que facilita una generalización más efectiva.

El procedimiento de aumento de datos consistirá en generar un nuevo audio por cada instancia de cada combinación de técnicas aplicada. Así, si aplicamos la técnica de *Variación de la Velocidad* sobre el corpus, triplicaremos su tamaño, ya que tendremos los ficheros de audio originales, los ficheros de audio modificados a velocidad 0.9 y los modificados a velocidad 1.1. Cabe aclarar que el aumento de datos se aplica sólo sobre la partición *train*.

4.2 Modelos evaluados

El objetivo de este trabajo es desarrollar un modelo de ASR que mejore el rendimiento de los sistemas para personas con SD. Se llevan a cabo cuatro tipos de experimentos distintos, cada uno diseñado para evaluar el rendimiento del modelo en diferentes escenarios. Estos experimentos incluyen el uso de corpus aumentados para habla Down, así como el análisis del nuevo modelo *whisper-large-v3*. Se realizan ajustes de hiperparámetros con el corpus FLEURS y se llevan a cabo experimentos de fine-tuning.

En nuestro estudio, no sólo nos interesa ver el rendimiento del modelo para habla Down, sino que también nos interesa ver cómo evoluciona el rendimiento para habla típica en función del modelo preparado. Es por eso que todo experimento realizado será evaluado tanto para habla Down como para habla típica.

Se han realizado cuatro tipos de experimentos diferentes:

- **Experimento Base**: Se evalúa el modelo base (BASE) con diferentes corpus y se realiza un primer fine-tuning (FT-PC) con cada división del corpus PRAUTOCAL Down.
- **Experimento Base con aumento de datos**: Se realizan 4 experimentos distintos por división, utilizando corpus aumentados con una sola técnica de las descritas (DAB-X). Se busca determinar la utilidad de las técnicas de aumento de datos para ayudar al modelo a generalizar.
- **Experimentos Mixtos con aumento de datos**: Se llevan a cabo 11 experimentos distintos para cada tipo de división, combinando dos, tres o cuatro de las técnicas de aumento de datos anteriormente descritas, cada una de las cuales genera el correspondiente fichero de audio extra.
- **Experimentos con *whisper-large-v3***: Se replican 3 experimentos sobre el nuevo modelo *whisper-large-v3*, seleccionando los más representativos. Se ajustan hiperparámetros con el corpus FLEURS.

4.3 Métricas

Además de la métrica estándar WER, aplicada tanto a frases sueltas como a colecciones de frases de un mismo usuario o de una misma partición, se ha empleado el **Análisis con BERT F1 Score**.

BERTScore (Zhang et al., 2020) es una métrica de evaluación automática para la generación de texto que calcula una puntuación de similitud para cada token en la oración candidata con cada token en la oración de referencia. Aprovecha las incorporaciones contextuales previamente entrenadas de los modelos BERT y relaciona palabras en oraciones candidatas y de referencia mediante similitud de coseno.

Analizar el rendimiento de un modelo en base al número de palabras coincidentes entre la referencia y la predicción puede llegar a ser algo pobre. Pueden existir casos con WER alto pero debido solamente a fallos puntuales en ciertas palabras que no son del todo importantes para comprender el significado de la frase. Para solucionar este problema usamos el BERT F1 Score, que ha sido ya utilizada para evaluar el rendimiento de sistemas ASR (Tobin et al., 2022). Esta métrica no se fija en la similitud ortográfica de las frases, sino en su similitud semántica. Consideramos que el análisis con esta métrica, junto con los resultados expuestos anteriormente, ayudará a dar unos resultados más completos.

De todos los experimentos realizados solo se va a aplicar esta métrica con los siguientes experimentos (al igual que en la experimentación con *whisper-large-v3*): **BASE**, **FT-PC** y **DAB-2**. La métrica se va a aplicar para las particiones *test* de los corpus que se han venido utilizando, y para ambas versiones de *whisper* (v2 y v3). Los resultados se pueden ver en la Tabla 4.

Corpus	BASE	FT-PC	DAB-1	DAB-2	DAB-3	DAB-4
División Aleatoria						
VoxPopuli (ES)	8.422	15.585	15.390	16.083	16.940	17.602
PRAUTOCAL Típico (ES)	3.409	2.193	1.929	1.982	2.167	2.511
PRAUTOCAL Down (SD)	64.315	23.565	14.653	14.576	22.559	22.463
PRAUTOCAL Down Clean (SD)	35.858	17.354	14.653	**14.576**	15.747	15.021
Nº ficheros de audio (Train)	-	1208	2416	2416	3624	3624
División por Actividades						
VoxPopuli (ES)	8.422	15.941	16.662	15.409	19.100	18.742
PRAUTOCAL Típico (ES)	4.239	7.781	6.815	9.149	10.789	10.572
PRAUTOCAL Down (SD)	56.166	44.645	29.091	27.283	28.164	29.671
PRAUTOCAL Down Clean (SD)	37.668	28.376	27.355	**27.029**	28.164	29.551
Nº ficheros de audio (Train)	-	1275	2550	2550	3825	3825

Tabla 2: WER (%) para cada corpus y experimento realizado con *whisper-large-v2*. BASE: sin fine-tuning. FT-PC: Fine-tuning con PRAUTOCAL Down. DAB-X: Fine-tuning con PRAUTOCAL Down, pero aplicándole una técnica de aumento de datos (1- Ruido Blanco, 2- Ruido Rosa, 3- Variación Velocidad, 4- Variación Tono). El conjunto test contiene: 875 para la división aleatoria y 808 para la división por actividades.

Corpus	BASE	FT-PC	DAB-2
División Aleatoria			
VoxPopuli (ES)	11.214	13.116	13.657
PRAUTOCAL Típico (ES)	3.013	1.929	2.748
PRAUTOCAL Down (SD)	48.734	24.995	22.695
PRAUTOCAL Down Clean (SD)	32.232	15.908	**15.471**
División por Actividades			
VoxPopuli (ES)	11.214	12.714	12.726
PRAUTOCAL Típico (ES)	3.488	7.137	7.352
PRAUTOCAL Down (SD)	52.851	51.553	32.221
PRAUTOCAL Down Clean (SD)	32.919	25.336	**25.175**

Tabla 3: WER (%) para cada corpus y experimento realizado con *whisper-large-v3*. BASE: sin fine-tuning. FT-PC: Fine-tuning con PRAUTOCAL Down. DAB-2: Fine-tuning con PRAUTOCAL Down, pero aplicándole la técnica que mejores resultados ha dado en la experimentación con *whisper-large-v2* (Ruido Rosa).

5 Resultados

Como muestran las Tablas 2 y 4, el modelo base permite alcanzar un resultado de WER del 8.4 % y un BERT Score F1 de 0.97 para el corpus VoxPopuli (habla típica español). Los resultados mejoran cuando este modelo base se aplica al corpus PRAUTOCAL Típico (WER: 3.4 % y BERT:0.988) y empeoran mucho cuando se aplica al corpus PRAUTOCAL Down (WER:$>$ 35,9 % y BERT:$<$ 0,9).

La aplicación de *fine-tuning* hace que los resultados empeoren para VOXPOPULI entre un 8 % y un 9 %y mejoren en el caso del corpus PRAUTOCAL. Está mejora es especialmente relevante en el caso de habla Down, donde las tasas pasan de 35.858 % a un 17.354 % para el caso de eliminación de frases en bucle y división aleatoria.

Las técnicas de aumento de datos (DAB) aportan una mejora que es muy poco significativa en el dataset Clean con división por actividades (en torno a un 1 %) y de hasta 9 % para dataset sin eliminar frases en bucle y división aleatoria. Esto claramente nos indica que el modelo *whisper*, en el caso de la división aleatoria, esta aprendiendo frases en el proceso de *fine-tuning* que luego aparecen en la partición *test*. Al aumentar el tamaño del corpus, se incrementa el número de repeticiones de cada frase, facilitando así el aprendizaje de la pronunciación de frases es-

Modelo	Corpus	BASE	FT-PC	DAB-2
V2	**División Aleatoria**			
	VoxPopuli (ES)	0.975	0.953	0.951
	PRAUTOCAL Típico (ES)	0.988	0.991	0.992
	PRAUTOCAL Down Clean (SD)	0.882	0.947	0.953
	División por Actividades			
	VoxPopuli (ES)	0.975	0.950	0.951
	PRAUTOCAL Típico (ES)	0.984	0.976	0.973
	PRAUTOCAL Down Clean (SD)	0.876	0.914	0.916
V3	**División Aleatoria**			
	VoxPopuli (ES)	0.965	0.961	0.959
	PRAUTOCAL Típico (ES)	0.989	0.992	0.989
	PRAUTOCAL Down Clean (SD)	0.893	0.948	0.952
	División por Actividades			
	VoxPopuli (ES)	0.965	0.959	0.961
	PRAUTOCAL Típico (ES)	0.987	0.976	0.976
	PRAUTOCAL Down Clean (SD)	0.895	0.925	0.925

Tabla 4: Análisis con la métrica BERTScore F1 de los modelos más significativos que se han generado a lo largo de toda la experimentación.

pecificas. El escaso incremento obtenido en la división por actividades, permite concluir que sólo con el *fine-tuning* **FT-PC** ya nos acercamos mucho al máximo aprendizaje que se puede obtener del corpus PRAUTOCAL.

Si nos restringimos al modelo *whisper* v2, del 18.5 % de mejora en el resultado de WER que se obtiene con la división aleatoria entre **FT-PC** y **BASE**, la mitad aproximadamente (9.3 %) es atribuible a mejora de resultado WER en reconocimiento de habla Down, que es la diferencia entre el WER de **BASE** y **FT-PC** en la división por actividades.

En cuanto a los resultados con cada una de las distintas técnicas de aumento de datos, podemos ver que en el caso de la división aleatoria todas mejoran, por lo tanto, podemos concluir que existe una cierta mejora que viene directamente proporcionada por el aumento de datos y no por la técnica empleada (aproximadamente un 2 %). En cuanto a la división por actividad podemos comprobar que las técnicas de *Variación de Velocidad* y *Variación de Ruido* empeoran el rendimiento del modelo. Esto se puede deber a que la simulación de "pseudo-hablantes" no sea del todo realista, y por tanto, estén introduciendo en el corpus características no reales. Los mejores resultados, en ambas divisiones, se obtienen con la técnica de *Introducción de Ruido Rosa*. Esta mejora puede venir explicada por la naturaleza propia de las personas con SD. Estas personas cometen una gran cantidad de bloqueos, repeticiones de palabras, elongaciones y ruidos entre palabras, que aparecen cuando sufren complicaciones en la frase que desean pronunciar. Estos sonido producidos pueden parecerse al ruido que se introduce con esta técnica, y por tanto, conseguir que el modelo se adapte mejor.

En cuanto a los experimentos mixtos con aumento de datos, ninguno de los experimentos obtuvo una mejora en el rendimiento del modelo. Los resultados de estos experimentos no se incluyen en el artículo por cuestiones de espacio.

La utilización de la versión v3 del modelo *whisper*, publicado mientras se preparaba este trabajo, no conlleva apenas diferencias de comportamiento frente a la versión v2 ya comentada.

Por lo que respecta a la métrica BERTScore F1, podemos ver que, en líneas generales, el modelo *whisper* (v2 y v3) genera predicciones de texto a partir de voz que tienen una muy buena correspondencia semántica incluso cuando el WER pueda no ser bajo, tanto en habla típica como Down. Así, el peor resultado obtenido es *0.876* – tengamos en cuenta que la métrica BERTScore F1 se mueve en un rango 0(pésimo)-1(óptimo). La WER más baja, obtenida para división aleatoria y **DAB-2**, es de 14.576 %, y se corresponde con

una BERTScore F1 de 0.953.

En las Tablas 2 a 4 se puede ver que existe una correlación casi total entre los resultados de WER y de BERT F1 Score para los diferentes modelos aplicados a cada corpus analizado (r-pearson: -0.998, p-value=4.1e-20): cuando el WER mejora (baja) el BERT también (sube), y viceversa. Aun así, se puede apreciar cómo grandes cambios en el WER no tienen porqué conllevar grandes variaciones en el BERT F1 Score (por ejemplo, el salto entre el modelo BASE y el FT-PC de *whisper-large-v2* evaluado con VoxPopuli), y cómo existen casos en los que pequeñas variaciones del WER suponen cambios más notables en el BERT F1 Score (como la diferencia entre FT-PC y DAB-2 de *whisper-large-v2* evaluado en la división Aleatoria de PRAUTOCAL Down). Esta variabilidad reside en el valor semántico de las palabras erradas/acertadas por *whisper*, haciendo así que no haya una relación total entre el WER y el BERTScore, y dando así valor a los resultados obtenidos con esta métrica.

En cuanto al habla Down, obtenemos una prueba más de lo influyente que es que existan frases comunes en *train* y *test*. Los resultados muestran que en los experimentos que involucran un *fine-tuning* con PRAUTOCAL Down el resultado de la división aleatoria es siempre mejor que el de la división por actividad. Esto se debe en su totalidad a la distribución de la división.

Por último, podemos corroborar que la aplicación del *fine-tuning* con PRAUTOCAL Down es muy beneficiosa para mejorar tanto el acierto en la predicción de palabras del modelo (WER), como su comprensión semántica. En el caso del aumento de datos aplicados a la división por actividad, se ve como la diferencia es prácticamente inexistente, siguiendo la tendencia que mostraba el WER, y demostrando así que el uso de ampliación de datos no es extremadamente beneficioso para que el modelo aprenda más sobre el habla Down. Por otro lado, sí que es ciertamente beneficioso en el caso de la división aleatoria, siendo esto debido a la naturaleza de la distribución originada por esta división.

6 Conclusiones y trabajo futuro

Los resultados presentados en este trabajo muestran que los modelos base *whisper-large-v2* y *whisper-large-v3* tienen un rendimiento muy malo en el reconocimiento de habla Down española. Aun así cabe destacar que su desempeño en esta tarea, o similares, es superior a la gran mayoría de modelos ASR actuales (Cibrian et al., 2024). Por otro lado, se ha demostrado que aplicar la técnica de *fine-tuning* con un corpus Down sobre estos modelos, proporciona grandes mejoras, tanto en la capacidad de adaptación del modelo al habla Down (división por actividad), como en la adaptación específica a la tarea del corpus PRAUTOCAL (división aleatoria). La mejora anteriormente citada, siempre lleva consigo un decremento en el rendimiento del modelo para habla típica, por lo que podemos concluir que el *fine-tuning* del modelo *whisper* tiene una capacidad de aprendizaje limitada. En lo que respecta a la aplicación de técnicas de aumento de datos, se observa que su eficacia es limitada cuando nos centramos en obtener mejoras en características generales del habla Down, pero sí resultan útiles para mejorar el rendimiento en escenarios dependientes de tarea, en el corpus en cuestión. Finalmente, en relación a la experimentación realizada, podemos concluir que la versión v3 del modelo *whisper* no es esencialmente mejor que la versión v2, en lo que a reconocimiento de habla Down española respecta.

Como trabajo futuro queda la exploración de técnicas de aumento de datos más complejas, como puede ser conversión de voz, o la aplicación de un TTS para simular la voz Down y aumentar la variedad léxica del corpus PRAUTOCAL. También queda como trabajo pendiente realizar un estudio más en profundidad de los resultados obtenidos en este artículo, centrando la atención en explicar porque se han obtenido estos resultados y su relación con las características propias de los hablantes Down. Además, como trabajo futuro, planteamos analizar más a fondo el fenómeno por el que en ocasiones el proceso de predicción entra en un bucle infinito. Finalmente, queda pendiente realizar un estudio y una experimentación más profunda y completa sobre el nuevo modelo *whisper-large-v3*.

Agradecimientos

Este trabajo ha sido realizado en el marco del proyecto **PID2021-126315OB-I00** que ha sido financiado por **MCIN / AEI / 10.13039/501100011033 / FEDER, EU**.

Bibliografía

Almadhor, A., R. Irfan, J. Gao, N. Saleem, H. Tayyab Rauf, y S. Kadry. 2023. E2e-dasr: End-to-end deep learning-based dysarthric automatic speech recognition. *Expert Systems with Applications*, 222:119797.

American Psychiatric Association. 2013. *Diagnostic and Statistical Manual of Mental Disorders, Fifth Edition (DSM-5)*. American Psychiatric Publishing, Arlington, VA.

Bhat, C. y H. Strik. 2020. Automatic assessment of sentence-level dysarthria intelligibility using blstm. *IEEE Journal of Selected Topics in Signal Processing*, 14(2):322–330.

Caton, S. y M. Chapman. 2016. The use of social media and people with intellectual disability: A systematic review and thematic analysis. *Journal of intellectual and developmental disability*, 41(2):125–139.

Chapman, R. S. 1997. Language development in children and adolescents with Down syndrome. *Mental Retardation and Developmental Disabilities Research Reviews*, 3(4):307–312.

Cibrian, F. L., K. Anderson, C. M. Abrahamsson, V. G. Motti, y others. 2024. Limitations in speech recognition for young adults with Down syndrome. *Research Square (Preprint Version 1)*.

Cleland, J., S. Wood, W. Hardcastle, J. Wishart, y C. Timmins. 2010. Relationship between speech, oromotor, language and cognitive abilities in children with Down's syndrome. *International journal of language & communication disorders*, 45(1):83–95.

Conneau, A., M. Ma, S. Khanuja, Y. Zhang, V. Axelrod, S. Dalmia, J. Riesa, C. Rivera, y A. Bapna. 2023. Fleurs: Few-shot learning evaluation of universal representations of speech. En *2022 IEEE Spoken Language Technology Workshop (SLT)*, páginas 798–805.

De Russis, L. y F. Corno. 2019. On the impact of dysarthric speech on contemporary asr cloud platforms. *Journal of Reliable Intelligent Environments*, 5:163–172.

Escudero-Mancebo, D., M. Corrales-Astorgano, V. Cardeñoso-Payo, L. Aguilar, C. González-Ferreras, P. Martínez-Castilla, y V. Flores-Lucas. 2022. PRAUTOCAL corpus: a corpus for the study of Down syndrome prosodic aspects. *Language Resources and Evaluation*, 56:191–224, Mayo.

Feng, J., J. Lazar, L. Kumin, y A. Ozok. 2010. Computer usage by children with Down syndrome: Challenges and future research. *ACM Transactions on Accessible Computing (TACCESS)*, 2(3):1–44.

Green, J. R., R. L. MacDonald, P.-P. Jiang, J. Cattiau, R. Heywood, R. Cave, K. Seaver, M. A. Ladewig, J. Tobin, M. P. Brenner, P. C. Nelson, y K. Tomanek. 2021. Automatic Speech Recognition of Disordered Speech: Personalized Models Outperforming Human Listeners on Short Phrases. En *Proc. Interspeech 2021*, páginas 4778–4782.

Hermann, E. y M. Magimai.-Doss. 2023. Few-shot Dysarthric Speech Recognition with Text-to-Speech Data Augmentation. En *Proc. INTERSPEECH 2023*, páginas 156–160.

Hu, R., J. Feng, J. Lazar, y L. Kumin. 2013. Investigating input technologies for children and young adults with Down syndrome. *Universal access in the information society*, 12:89–104.

Janbakhshi, P., I. Kodrasi, y H. Bourlard. 2021. Automatic dysarthric speech detection exploiting pairwise distance-based convolutional neural networks. En *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, páginas 7328–7332. IEEE.

Jiao, Y., M. Tu, V. Berisha, y J. Liss. 2018. Simulating dysarthric speech for training data augmentation in clinical speech applications. En *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, páginas 6009–6013. IEEE.

Kitzing, P., A. Maier, y V. L. Åhlander. 2009. Automatic speech recognition (asr) and its use as a tool for assessment or therapy of voice, speech, and language disorders. *Logopedics Phoniatrics Vocology*, 34(2):91–96.

Kumin, L. 2012. *Early communication skills for children with Down syndrome: A guide for parents and professionals*. Woodbine House, 3ª edición.

Laws, G. y D. V. Bishop. 2004. Verbal deficits in Down's syndrome and specific language impairment: a comparison. *International Journal of Language & Communication Disorders*, 39(4):423–451.

Lea, C., Z. Huang, J. Narain, L. Tooley, D. Yee, D. T. Tran, P. Georgiou, J. P. Bigham, y L. Findlater. 2023. From user perceptions to technical improvement: Enabling people who stutter to better use speech recognition. En *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, páginas 1–16.

MacDonald, R. L., P.-P. Jiang, J. Cattiau, R. Heywood, R. Cave, K. Seaver, M. A. Ladewig, J. Tobin, M. P. Brenner, P. C. Nelson, J. R. Green, y K. Tomanek. 2021. Disordered Speech Data Collection: Lessons Learned at 1 Million Utterances from Project Euphonia. En *Interspeech 2021*, páginas 4833–4837.

Martin, G. E., J. Klusek, B. Estigarribia, y J. E. Roberts. 2009. Language characteristics of individuals with Down syndrome. *Topics in language disorders*, 29(2):112–132.

Mitra, V., Z. Huang, C. Lea, L. Tooley, S. Wu, D. Botten, A. Palekar, S. Thelapurath, P. Georgiou, S. Kajarekar, y J. Bigham. 2021. Analysis and Tuning of a Voice Assistant System for Dysfluent Speech. En *Proc. Interspeech 2021*, páginas 4848–4852.

Prananta, L., B. Halpern, S. Feng, y O. Scharenborg. 2022. The Effectiveness of Time Stretching for Enhancing Dysarthric Speech for Improved Dysarthric Speech Recognition. En *Proc. Interspeech 2022*, páginas 36–40.

Radford, A., J. W. Kim, T. Xu, G. Brockman, C. McLeavey, y I. Sutskever. 2023. Robust speech recognition via large-scale weak supervision. En *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

Rosen, K. y S. Yampolsky. 2000. Automatic speech recognition and a review of its functioning with dysarthric speech. *Augmentative and Alternative Communication*, 16(1):48–60.

Schultz, B. G., V. S. A. Tarigoppula, G. Noffs, S. Rojas, A. van der Walt, D. B. Grayden, y A. P. Vogel. 2021. Automatic speech recognition in neurodegenerative disease. *International Journal of Speech Technology*, 24(3):771–779.

Shahamiri, S. R. 2021. Speech vision: An end-to-end deep learning-based dysarthric automatic speech recognition system. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 29:852–861.

Shor, J., D. Emanuel, O. Lang, O. Tuval, M. Brenner, J. Cattiau, F. Vieira, M. McNally, T. Charbonneau, M. Nollstadt, A. Hassidim, y Y. Matias. 2019. Personalizing asr for dysarthric and accented speech with limited data. En *Interspeech 2019*, interspeech2019. ISCA, Septiembre.

Tanis, E. S., S. Palmer, M. Wehmeyer, D. K. Davies, S. E. Stock, K. Lobb, y B. Bishop. 2012. Self-report computer-based survey of technology use by people with intellectual and developmental disabilities. *Intellectual and developmental disabilities*, 50(1):53–68.

Timmer, J. y M. Koenig. 1995. On generating power law noise. *Astronomy and Astrophysics, v. 300, p. 707*, 300:707.

Tobin, J., Q. Li, S. Venugopalan, K. Seaver, R. Cave, y K. Tomanek. 2022. Assessing ASR Model Quality on Disordered Speech using BERTScore. En *Proc. 1st Workshop on Speech for Social Good (S4SG)*, páginas 26–30.

Tobin, J. y K. Tomanek. 2022. Personalized automatic speech recognition trained on small disordered speech datasets. En *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, páginas 6637–6641.

Tomanek, K., F. Beaufays, J. Cattiau, A. Chandorkar, y K. C. Sim. 2021. On-device personalization of automatic speech recognition models for disordered speech. *arXiv:2106.10259*.

Venugopalan, S., J. Shor, M. Plakal, J. Tobin, K. Tomanek, J. R. Green, y M. P. Brenner. 2021. Comparing Supervised Models and Learned Speech Representations for Classifying Intelligibility of Disordered Speech on Selected Phrases. En *Interspeech 2021*, páginas 4843–4847.

Wang, C., M. Riviere, A. Lee, A. Wu, C. Talnikar, D. Haziza, M. Williamson, J. Pino, y E. Dupoux. 2021. VoxPopuli: A large-scale multilingual speech corpus for representation learning, semi-supervised learning and interpretation. En C. Zong F. Xia W. Li, y R. Navigli, editores, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, páginas 993–1003, Online, Agosto. Association for Computational Linguistics.

Wong, B., C. Brebner, P. McCormack, y A. Butcher. 2015. Word production inconsistency of Singaporean-English-speaking adolescents with Down Syndrome. *International journal of language & communication disorders*, 50(5):629–645.

Zhang, T., V. Kishore, F. Wu, K. Q. Weinberger, y Y. Artzi. 2020. Bertscore: Evaluating text generation with bert. En *International Conference on Learning Representations*.

Apéndice D

Análisis GoP por frases

RESULTADOS DE GOP EJECUCIÓN EXPERIMENTO EXP-TOT-T (FRASE POR FRASE)

UN CASO DE ESTUDIO PARA HABLA ANÓMALA EN ESPAÑOL

TFG
GRUPO DE INVESTIGACIÓN ECA-SIMM.
UNIVERSIDAD DE VALLADOLID

David Fernández García (david.fernandez@estudiantes.uva.es)

1 de febrero de 2024

FD13036D0220R03

FRASE: <has>has sido muy amable Juan, muchas gracias

TRANSCRIPCIÓN FONÉTICA: s a s i ð o m u i a m a v l e x w a n m u t ʃ a s r a θ j a s

CLASE: 1

GOP MEDIO FRASE: 10.61

GOP ACUMULADO FRASE: 318.39

GOP POR FONEMA:

's': 33.8713

'a': 26.7256

's': 37.0216

'i': 26.0021

'ð': -6.2178

'o': 22.9910

'm': -21.2274

'u': -1.7149

'i': 39.5913

'a': 14.7262

'm': 7.1593

'a': 33.1931

'v': -0.2273

'l': 1.6886

'e': 10.0744

'x': -8.1629

'w': -9.2978

'a': 24.4032

'n': -4.8363

'm': -25.7347

'u': 6.7307

't∫': -4.4649

'a': 2.4093

's': 9.0825

'r': 36.3943

'a': 37.8744

'θ': -5.5298

'j': -5.6921

'a': 14.4427

's': 27.1162

FD13036D0320R03

FRASE: Si, a ver, si, la necesito, cuanto vale

TRANSCRIPCIÓN FONÉTICA: i a b e r s i l a n e θ e s i t o k w a n t o b a l e

CLASE: 1

GOP MEDIO FRASE: 11.81

GOP ACUMULADO FRASE: 318.91

GOP POR FONEMA:

'i': 26.5758

'a': -0.2556

'b': -6.1539

'e': 12.8346

'r': 3.6641

's': 24.3332

'i': 23.8071

'l': 1.8183

'a': 5.2052

'n': -6.5910

'e': 21.5771

'θ': -6.0542

'e': 10.5223

's': 5.7998

'i': 9.9386

't': 21.2754

'o': 27.6340

'k': 5.4459

'w': -5.5498

'a': 24.2203

'n': 3.4459

't': 16.8717

'o': 27.0965

'b': -4.4416

'a': 31.6577

'l': 18.5624

'e': 25.6739

FD13038D0420R01

FRASE: Hola tio Pau, ya vuelvo a casa

TRANSCRIPCIÓN FONÉTICA: o l a t i o p a u a b w e l v o a k a s a

CLASE: 1

GOP MEDIO FRASE: 14.64

GOP ACUMULADO FRASE: 307.5

GOP POR FONEMA:

'o': -5.8778

'l': 7.9896

'a': 37.4086

't': 9.8852

'i': 14.3334

'o': 26.8908

'p': 5.5722

'a': 32.6414

'u': -6.9313

'a': 14.1756

'b': -6.0384

'w': -4.7928

'e': 14.9582

'l': 0.4258

'v': 17.3111

'o': 29.5869

'a': 29.8796

'k': 11.6200

'a': 34.1780

's': 14.0068

'a': 30.2755

FD13039D0120R01

FRASE: Hasta luego, tio Pau

TRANSCRIPCIÓN FONÉTICA: a s t a l w e ɣ o t i o p a u

CLASE: 1

GOP MEDIO FRASE: 14.48

GOP ACUMULADO FRASE: 217.17

GOP POR FONEMA:

'a': 6.3222

's': 26.7613

't': 12.6243

'a': 35.5991

'l': 9.9991

'w': -6.2704

'e': 0.4751

'γ': -6.3583

'o': 29.4265

't': 12.7362

'i': 30.0937

'o': 34.0313

'p': 6.6259

'a': 25.0841

'u': 0.0220

FD13039D0310R02

FRASE: Hola, tienes lupas, queria comprar una

TRANSCRIPCIÓN FONÉTICA: o l a t j e n e s l u p a s k e r i a k o m p r a r u n a

CLASE: 1

GOP MEDIO FRASE: 16.01

GOP ACUMULADO FRASE: 464.39

GOP POR FONEMA:

'o': 18.5774

'l': 7.2857

'a': 37.8175

't': 0.7887

'j': -4.8447

'e': 28.7618

'n': 31.5014

'e': 22.9198

's': 5.5794

'l': 12.2834

'u': 20.3568

'p': 8.4107

'a': 23.9508

's': 23.5591

'k': -3.4542

'e': 18.0509

'r': -6.2389

'i': 25.5629

'a': 26.7575

'k': 14.6725

'o': 29.2802

'm': 21.4848

'p': 1.0693

'r': 13.7435

'a': 20.5341

'r': 11.4827

'u': 6.6493

'n': 13.9245

'a': 33.9243

FD13039D0320R01

FRASE: Si, <l>la necesito, cuanto vale

TRANSCRIPCIÓN FONÉTICA: s i e l e l a n e θ e s i t o k w a n t o b a l e

CLASE: 1

GOP MEDIO FRASE: 15.67

GOP ACUMULADO FRASE: 391.79

GOP POR FONEMA:

's': 3.2413

'i': 32.9340

'e': 32.1393

'l': 3.4224

'e': 24.0611

'l': 9.7600

'a': 39.8029

'n': 23.8640

'e': 12.8922

'θ': -6.1453

'e': 12.1159

's': 14.5809

'i': 27.3306

't': 24.7599

'o': 21.1432

'k': 9.8467

'w': -8.1523

'a': 15.7160

'n': 7.4601

't': -12.1666

'o': 44.5631

'b': -4.8440

'a': 34.8818

'l': 13.3763

'e': 15.2097

FD13039D0510R02

FRASE: Hola, tio Pau, sabes donde vive la señora Luna

TRANSCRIPCIÓN FONÉTICA: o l a t i o p a u s a v e s d o n d e b i v e l a s e n o r a l u n a

CLASE: 1

GOP MEDIO FRASE: 13.46

GOP ACUMULADO FRASE: 471.08

GOP POR FONEMA:

'o': 4.8521

'l': 20.0479

'a': 38.1792

't': 6.9392

'i': 28.3392

'o': 15.2685

'p': 4.5994

'a': 37.2040

'u': -11.1094

's': 9.3795

'a': 27.2162

'v': -8.4525

'e': 24.7332

's': 0.6203

'd': 3.1162

'o': 18.0491

'n': 3.9904

'd': -27.4167

'e': 17.0205

'b': -5.8798

'i': 10.4336

'v': -24.2956

'e': 31.7976

'l': 21.6188

'a': 30.2753

's': 29.5514

'e': 21.2880

'n': 15.2068

'o': 13.1527

'r': -12.9222

'a': 40.3195

'l': 20.8103

'u': 7.0805

'n': 25.4761

'a': 34.5884

FD23027D0820R01

FRASE: <ehh>si, no, si no tardo si por favor

TRANSCRIPCIÓN FONÉTICA: e si no sinotarðosiporfavor

CLASE: 1

GOP MEDIO FRASE: 17.1

GOP ACUMULADO FRASE: 410.4

GOP POR FONEMA:

'e': 32.0995

's': 26.1565

'i': 37.8320

'n': -12.1756

'o': 19.9531

's': 27.3115

'i': 27.9465

'n': 17.8234

'o': 9.6827

't': 11.6966

'a': 22.8694

'r': 5.9236

'ð': -6.2062

'o': 29.0427

's': 25.3374

'i': 41.3172

'p': -1.2740

'o': 14.0352

'r': 17.0395

'f': 1.7944

'a': 20.2547

'v': 5.3954

'o': 22.7650

'r': 13.7828

FD23027D0930R01

FRASE: <eh>que tengo que decir ahora, <ah><mmm>no no la de la cuerda, si

TRANSCRIPCIÓN FONÉTICA: e k e t e ŋ g o k e d e θ i r a o r a a m n o n o l a d e l a k w e r ð a s i

CLASE: 1

GOP MEDIO FRASE: 11.17

GOP ACUMULADO FRASE: 435.55

GOP POR FONEMA:

'e': 27.9879

'k': 5.9693

'e': 11.7621

't': 23.8406

'e': 5.9046

'ŋ': -5.1940

'g': -4.9171

'o': 4.4558

'k': 27.3792

'e': 7.4544

'd': -13.3787

'e': 2.5866

'θ': -8.2600

'i': 27.5887

'r': 1.7681

'a': 30.8327

Bibliografía

[1] Ahmad Almadhor et al. «E2E-DASR: End-to-end deep learning-based dysarthric automatic speech recognition». En: *Expert Systems with Applications* 222 (2023), pág. 119797. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2023.119797. URL: https://www.sciencedirect.com/science/article/pii/S0957417423002981.

[2] Alexei Baevski et al. *wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations*. 2020. arXiv: 2006.11477 [cs.CL].

[3] Chitralekha Bhat y Helmer Strik. «Automatic Assessment of Sentence-Level Dysarthria Intelligibility Using BLSTM». En: *IEEE Journal of Selected Topics in Signal Processing* 14.2 (2020), págs. 322-330. DOI: 10.1109/JSTSP.2020.2967652.

[4] Franceli L Cibrian et al. «Limitations in Speech Recognition for Young Adults with Down Syndrome». En: *Research Square (Preprint Version 1)* (2024). DOI: https://doi.org/10.21203/rs.3.rs-3974634/v1.

[5] Franceli L. Cibrian et al. «Limitations in Speech Recognition for Young Adults with Down Syndrome». En: (feb. de 2024). DOI: 10.21203/rs.3.rs-3974634/v1. URL: http://dx.doi.org/10.21203/rs.3.rs-3974634/v1.

[6] Alexis Conneau et al. *FLEURS: Few-shot Learning Evaluation of Universal Representations of Speech*. 2022. arXiv: 2205.12446 [cs.CL].

[7] Luigi De Russis y Fulvio Corno. «On the impact of dysarthric speech on contemporary ASR cloud platforms». En: *Journal of Reliable Intelligent Environments* 5 (2019), págs. 163-172.

[8] Jacob Devlin et al. «BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding». En: *North American Chapter of the Association for Computational Linguistics*. 2019. URL: https://api.semanticscholar.org/CorpusID:52967399.

[9] David Escudero-Mancebo et al. «PRAUTOCAL corpus: a corpus for the study of Down syndrome prosodic aspects». En: *Language Resources and Evaluation* 56.1 (mayo de 2021), págs. 191-224. DOI: 10.1007/s10579-021-09542-8. URL: https://doi.org/10.1007/s10579-021-09542-8.

[10] Jonas Gehring et al. *Convolutional Sequence to Sequence Learning*. 2017. arXiv: 1705.03122 [cs.CL].

[11] Behrooz Ghorbani et al. *Scaling Laws for Neural Machine Translation*. 2021. arXiv: 2109.07740 [cs.LG].

[12] Naman Goyal et al. *The FLORES-101 Evaluation Benchmark for Low-Resource and Multilingual Machine Translation*. 2021. arXiv: 2106.03193 [cs.CL].

[13] Jordan R. Green et al. «Automatic Speech Recognition of Disordered Speech: Personalized Models Outperforming Human Listeners on Short Phrases». En: *Proc. Interspeech 2021*. 2021, págs. 4778-4782. DOI: 10.21437/Interspeech.2021-1384.

[14] Kaiming He et al. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385 [cs.CV].

[15] Enno Hermann y Mathew Magimai.-Doss. «Few-shot Dysarthric Speech Recognition with Text-to-Speech Data Augmentation». En: *Proc. INTERSPEECH 2023*. 2023, págs. 156-160. DOI: 10.21437/Interspeech.2023-2481.

[16] Ruimin Hu et al. «Investigating input technologies for children and young adults with Down syndrome». En: *Universal access in the information society* 12 (2013), págs. 89-104.

[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 (nov. de 2017), págs. 3165-3177. DOI: 10.1121/1.5011159.

[18] Parvaneh Janbakhshi, Ina Kodrasi y Hervé Bourlard. «Automatic dysarthric speech detection exploiting pairwise distance-based convolutional neural networks». En: *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2021, págs. 7328-7332.

[19] Farhad Javanmardi, Sudarsana Reddy Kadiri y Paavo Alku. «Exploring the Impact of Fine-Tuning the Wav2vec2 Model in Database-Independent Detection of Dysarthric Speech». En: *IEEE Journal of Biomedical and Health Informatics* (2024), págs. 1-12. DOI: 10.1109/JBHI.2024.3392829.

[20] Yishan Jiao et al. «Simulating dysarthric speech for training data augmentation in clinical speech applications». En: *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE. 2018, págs. 6009-6013.

[21] Kak Soky et al. *Domain and Language Adaptation of Large-scale Pretrained Model for Speech Recognition of Low-resource Language*. Nov. de 2022.

[22] Peter Kitzing, Andreas Maier y Viveka Lyberg Åhlander. «Automatic speech recognition (ASR) and its use as a tool for assessment or therapy of voice, speech, and language disorders». En: *Logopedics Phoniatrics Vocology* 34.2 (2009), págs. 91-96.

[23] Philipp Klumpp et al. «Common Phone: A Multilingual Dataset for Robust Acoustic Modelling». En: *Proceedings of the Thirteenth Language Resources and Evaluation Conference*. Ed. por Nicoletta Calzolari et al. Marseille, France: European Language Resources Association, jun. de 2022, págs. 763-768. URL: https://aclanthology.org/2022.lrec-1.81.

[24] Libby Kumin. *Early communication skills for children with Down syndrome: A guide for parents and professionals*. 3ª edición. Woodbine House, 2012.

[25] Colin Lea et al. «From user perceptions to technical improvement: Enabling people who stutter to better use speech recognition». En: *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. 2023, págs. 1-16.

[26] Robert L. MacDonald et al. «Disordered Speech Data Collection: Lessons Learned at 1 Million Utterances from Project Euphonia». En: *Interspeech 2021*. 2021, págs. 4833-4837. DOI: 10.21437/Interspeech.2021-697.

[27] A Mehrabian y S R Ferris. «Inference of attitudes from nonverbal communication in two channels». en. En: *J. Consult. Psychol.* 31.3 (jun. de 1967), págs. 248-252.

[28] A Mehrabian y M Wiener. «Decoding of inconsistent communications». en. En: *J. Pers. Soc. Psychol.* 6.1 (mayo de 1967), págs. 109-114.

[29] Vikramjit Mitra et al. «Analysis and Tuning of a Voice Assistant System for Dysfluent Speech». En: *Proc. Interspeech 2021*. 2021, págs. 4848-4852. DOI: 10.21437/Interspeech.2021-2006.

[30] Asunción Moreno et al. «Albayzin speech database: Design of the phonetic corpus». En: vol. 1. Sep. de 1993. DOI: 10.21437/Eurospeech.1993-66.

[31] Luke Prananta et al. *The Effectiveness of Time Stretching for Enhancing Dysarthric Speech for Improved Dysarthric Speech Recognition*. 2022. arXiv: 2201.04908 [cs.SD].

[32] Jinzi Qi y Hugo Van hamme. *Parameter-efficient dysarthric speech recognition using adapter fusion and householder transformation*. 2023.

[33] Jinzi Qi y hugo Van hamme. *Speech disorder classification using extended factorized hierarchical variational auto-encoders*. eng. 2021.

[34] Zhaopeng Qian, Kejing Xiao y Chongchong Yu. «A survey of technologies for automatic Dysarthric speech recognition». en. En: *EURASIP J. Audio Speech Music Process.* 2023.1 (nov. de 2023).

[35] Alec Radford et al. *Robust Speech Recognition via Large-Scale Weak Supervision*. 2022. arXiv: 2212.04356 [eess.AS].

[36] Kristin Rosen y Sasha Yampolsky. «Automatic speech recognition and a review of its functioning with dysarthric speech». En: *Augmentative and Alternative Communication* 16.1 (2000), págs. 48-60. DOI: 10.1080/07434610012331278904.

[37] Hyuksu Ryu y Minhwa Chung. «Mispronunciation Diagnosis of L2 English at Articulatory Level Using Articulatory Goodness-Of-Pronunciation Features». En: ago. de 2017, págs. 65-70. DOI: 10.21437/SLaTE.2017-12.

[38] Robin M. Schmidt. *Recurrent Neural Networks (RNNs): A gentle Introduction and Overview*. 2019. arXiv: 1912.05911 [cs.LG].

[39] Benjamin G Schultz et al. «Automatic speech recognition in neurodegenerative disease». En: *International Journal of Speech Technology* 24.3 (2021), págs. 771-779.

[40] Seyed Reza Shahamiri. «Speech vision: An end-to-end deep learning-based dysarthric automatic speech recognition system». En: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 29 (2021), págs. 852-861.

[41] Helmer Strik et al. «Comparing different approaches for automatic pronunciation error detection». En: *Speech Communication* 51.10 (2009). Spoken Language Technology for Education, págs. 845-852. ISSN: 0167-6393. DOI: https://doi.org/10.1016/j.specom.2009.05.007. URL: https://www.sciencedirect.com/science/article/pii/S0167639309000715.

[42] Sweekar Sudhakara et al. «An Improved Goodness of Pronunciation (GoP) Measure for Pronunciation Evaluation with DNN-HMM System Considering HMM Transition Probabilities». En: *Proc. Interspeech 2019*. 2019, págs. 954-958. DOI: 10.21437/Interspeech.2019-2363.

[43] J. Timmer y M. Koenig. «On generating power law noise.» En: *Astronomy and Astrophysics* 300 (ago. de 1995), pág. 707.

[44] Jimmy Tobin et al. «Assessing ASR Model Quality on Disordered Speech using BERTScore». En: *Proc. 1st Workshop on Speech for Social Good (S4SG)*. 2022, págs. 26-30. DOI: 10.21437/S4SG.2022-6.

[45] Katrin Tomanek et al. «On-Device Personalization of Automatic Speech Recognition Models for Disordered Speech». En: *arXiv:2106.10259* (2021). arXiv: 2106.10259 [eess.AS].

[46] Rong Tong et al. «Goodness of tone (GOT) for non-native Mandarin tone recognition». En: *Proc. Interspeech 2015*. 2015, págs. 801-805. DOI: 10.21437/Interspeech.2015-254.

[47] Ashish Vaswani et al. *Attention Is All You Need*. 2023. arXiv: 1706.03762 [cs.CL].

[48] Subhashini Venugopalan et al. «Comparing Supervised Models and Learned Speech Representations for Classifying Intelligibility of Disordered Speech on Selected Phrases». En: *Interspeech 2021*. 2021, págs. 4843-4847. DOI: 10.21437/Interspeech.2021-1913.

[49] Changhan Wang et al. *VoxPopuli: A Large-Scale Multilingual Speech Corpus for Representation Learning, Semi-Supervised Learning and Interpretation*. 2021. arXiv: 2101.00390 [cs.CL].

[50] Huimeng Wang et al. *Enhancing Pre-trained ASR System Fine-tuning for Dysarthric Speech Recognition using Adversarial Data Augmentation*. 2024. arXiv: 2401.00662 [cs.SD].

[51] Eunjung Yeo et al. «Speech Intelligibility Assessment of Dysarthric Speech by using Goodness of Pronunciation with Uncertainty Quantification». En: ago. de 2023, págs. 166-170. DOI: 10.21437/Interspeech.2023-173.

[52] Tianyi Zhang* et al. «BERTScore: Evaluating Text Generation with BERT». En: *International Conference on Learning Representations*. 2020. URL: https://openreview.net/forum?id=SkeHuCVFDr.