Master in Management + Analytics

Diseño, implementación y evaluación de metodologías para el procesamiento automático del habla en personas con hipoacusia

Autoría: González, Joaquín
Fecha de defensa de la tesis: 2023

UNIVERSIDAD
TORCUATO DI TELLA

Master in Management + Analytics

Diseño, implementación y evaluación de metodologías para el procesamiento automático del habla en personas con hipoacusia

TESIS

Ing. Joaquin Gonzalez

Mayo 2023

Tutores:
Dr. Agustín Gravano
Dra. Carolina Andrea Gattei

Resumen

El presente trabajo se enmarca en el proyecto de investigación de la Dra. Carolina Gattei y la Mag. Analí Taboh, radicado en la Universidad Torcuato Di Tella, en el cual se realizan pruebas que ayudan a diagnosticar a personas con distintas patologías del habla y la audición. Los datos recolectados por el equipo son entrevistas a niños/as con hipoacusia y equipados con audífono(s) y/o implante coclear. La transcripción y análisis de estas pruebas es un proceso manual y costoso en términos de tiempo y recursos.

Actualmente, el estado del arte de los principales modelos de reconocimiento automático del habla (ASR por sus siglas en inglés) son entrenados para trabajar en lenguas de amplio alcance (como el español o el inglés) y para personas sin dificultades en el habla. Sin embargo, cuando se trata de hablantes con patologías del habla y la audición, como parte de los resultados de este trabajo se ha corroborado que estos modelos tienen una baja tasa de aciertos dada la escasa, o nula, representación que tiene esta población en los datos de entrenamiento.

En el presente trabajo se argumenta que, haciendo uso de técnicas de apredizaje automático y transferencia de conocimiento, puede sacarse provecho de modelos de ASR pre entrenados para lograr adaptarlos a una población con modelos acústicos poco representados actualmente.

El objetivo general de esta tesis es explorar la factibilidad de implementar una metodología que permita la construcción y evaluación de sistemas de procesamiento automático del habla que generen transcripciones enriquecidas sobre los audios producidos por los/as niños/as con dificultades del habla y la audición. Esto requiere de ensamblar distintos modelos especializados en tareas de reconocimiento de voz y elaborar un mecanismo que permita evaluar y comparar estos sistemas.

Como resultado, se logró conformar un *dataset* para tareas de entrenamiento supervisado inédito de una población poco respresentada. Por otro lado, se desarrolló un sistema que implementa la metodología propuesta y que permite automatizar el proceso de transcripción y evaluación y el de *fine tuning* de los modelos utilizados, pudiéndose extender para soportar otros de ser necesario. Además, se pudo confirmar que el proceso de optimización de los modelos muestra un alto grado de adaptación a los patrones del habla sobre la población de estudio.

Con poco trabajo adicional, el equipo de la Dra. Carolina Gattei podría comenzar a utilizar la aplicación elaborada en el marco de esta tesis para acelerar los tiempos de transcripción y puntuación, que actualmente se realizan de forma manual. Además, considerando los resultados obtenidos en la tareas de transferencia de conocimiento y de contar con más horas de audio en el *dataset*, podrían utilizarse las métricas elaboradas por el sistema de transcripción (o integrarse otras que el equipo considere) como un mecanismo para interpretar la inteligibilidad de los/as niños/as y otras variables individuales (como edad de equipamiento con el implante coclear, tipo de implante, etiología de la sordera del niño/a, por ejemplo). En este caso, la métrica se utilizaría para diagnosticar a la población de estudio, en vez de al modelo.

Abstract

This study is part of the research project led by Dr. Carolina Gattei and Mag. Analí Taboh, based at Universidad Torcuato Di Tella. The project focuses on conducting tests to diagnose individuals with various speech and hearing pathologies. The data collected by the team consists of interviews with children with hearing loss who are equipped with hearing aids and/or cochlear implants. The transcription and analysis of these tests are manual and time-consuming processes that require significant time and resources.

Currently, the state-of-the-art automatic speech recognition (ASR) models are primarily trained to work with widely spoken languages such as Spanish or English and with individuals without speech difficulties. However, as part of the findings of this study, it has been corroborated that when it comes to speakers with speech and hearing pathologies, these models have a low accuracy rate due to the scarce or non-existent representation of this population in the training data.

This study argues that by leveraging machine learning techniques and knowledge transfer, pre-trained ASR models can be effectively adapted to a population with currently underrepresented acoustic models.

The overall objective of this thesis is to explore the feasibility of implementing a methodology that allows the construction and evaluation of automatic speech processing systems capable of generating enriched transcriptions of audio produced by children with speech and hearing difficulties. This requires assembling different models specialized in speech recognition tasks and developing a mechanism to evaluate and compare these systems.

As a result, a new dataset for supervised training tasks of an underrepresented population was compiled. Additionally, a system was developed to implement the proposed methodology, automating the transcription, evaluation, and fine-tuning processes of the used models. This system can be extended to support other models if needed. Furthermore, it was confirmed that the model optimization process exhibits a high degree of adaptation to speech patterns within the study population.

With minimal additional work, Dr. Carolina Gattei's team could start utilizing the application developed within the scope of this thesis to accelerate the transcription and scoring processes, which are currently done manually. Moreover, considering the results obtained in the knowledge transfer tasks and the availability of more audio hours in the dataset, the metrics generated by the transcription system (or other metrics integrated by the team) could be used as a mechanism to interpret the intelligibility of the children and other individual variables (such as age of cochlear implant equipment, type of implant, etiology of the child's hearing loss, for example). In this case, the metrics would be used to diagnose the study population, rather than evaluating the model.

A mis directores, Agustín Gravano y Carolina Gattei. Por hacer fácil un proceso difícil y por su generosidad y dedicación durante todos estos meses.

A Mora y Julia, mis compañeras de aventuras.

A María Claudia Abeledo, por la confianza y el apoyo de siempre.

Índice

Índice de tablas

Índice de figuras

1. Introducción

1.1. Contexto

El presente trabajo se enmarca en el proyecto de investigación de la Dra. Carolina Gattei y la Mag. Analí Taboh, radicado en la Universidad Torcuato Di Tella, en el cual se realizan pruebas que ayudan a diagnosticar a personas con distintas patologías del habla y la audición (Taboh et al., 2022 y Taboh et al., 2021). En estas pruebas, niños/as con hipoacusia y equipados con audífono(s) y/o implante coclear repiten una secuencia de 15-20 frases producidas primero por una profesional. La secuencia es la misma en todos los casos, un subset de la batería CELF 4 Semel y Secord, 2006 (ver Anexo A), una prueba estandarizada diseñada para detectar impedimentos en el lenguaje en niños/as. Las oraciones varían en longitud y complejidad sintáctica. Buscan evaluar la memoria de corto plazo de los niños/as, viendo si pueden repetir las oraciones sin cambiar el significado, la forma de las palabras y la estructura sintáctica como por ejemplo, "El conejito café se comió todas las zanahorias en el jardín".

El resultado de estas pruebas son audios en los que registran la producción oral de las oraciones: aquellas leídas en voz alta por el/la profesional y aquellas repetidas por los/as niños/as. El análisis de estos materiales permite al equipo de investigación indagar sobre las características de los casos así como contribuir a la generación de evidencia para diagnósticos oportunos.

La transcripción y análisis de estas pruebas es un proceso manual y costoso en términos de tiempo y recursos, lo cual pone restricciones en la realización de las mismas.

1.2. Problema

En este contexto, se hizo necesario poder contar con herramientas que produzcan resultados lo suficientemente confiables que faciliten la transcripción y el posterior análisis de audios producidos por personas con patologías en el habla que contribuya a la elaboración de los test que lleva adelante el equipo de investigación de la Dra. Carolina Gattei y que sean de utilidad a futuro para planificar eventuales sistemas automáticos de medición y enseñanza para la población de estudio.

Actualmente, el estado del arte de los principales modelos de ASR[1] son entrenados para trabajar en lenguas de amplio alcance (español, inglés, etc.) y para personas sin dificultades en el habla. Por otro lado, el acceso a *datasets* que representen a nuestra población objetivo es escaso y de difícil acceso (Kafle et al., 2019). Esto ha motivado iniciativas como el proyecto Euphonia[2], en donde se busca incorporar de manera masiva a *datasets* de entrenamiento información de personas con patalogías en el habla.

Dada la escasa, o nula, representación que tienen estos hablantes en los *datasets* de entrenamiento de los modelos de ASR, se espera en una primera instancia, que modelos pre entrenados no mantengan el mismo desempeño sobre los audios producidos por niños/as con patología del habla y la audición como los que se utilizaron en este trabajo como se

[1] Automatic Speech Recognition

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

observa en los resultados obtenidos por Gutz et al., 2022 y Glasser et al., 2017. Es por estas razones que se tuvieron en cuenta distintas consideraciones para analizar y mejorar su desempeño (Jeong et al., 2021 y Shor et al., 2019).

1.3. Objetivo

Esta tesis busca explorar la factibilidad de implementar una metodología que permita la construcción y evaluación de sistemas de procesamiento automático del habla que generen transcripciones enriquecidas sobre los audios producidos por los/as niños/as con dificultades del habla y la audición.
Esto se logró, en una primera instancia, diseñando una metodología para realizar las distintas tareas involucradas en un sistema de procesamiento automático del habla que, a su vez, permitan la evaluación del desempeño de este sistema recolectando distintas métricas.

Por otro lado, se desarrolló una aplicación que implementa la metodología propuesta a través de la integración de distintas tecnologías y modelos de estado del arte especializados en tareas de reconocimiento de voz.

Algunas de las tareas más relevantes para llevar adelante el desarrollo metodológico y de la aplicación propuesta se listan a continuación:

- Tomando como punto de partida las entrevistas brindadas por el equipo de la Dra. Carolina Gattei, construir un *dataset* que sirva de *ground truth* para las tareas de evaluación y entrenamiento.
- Integrar a la aplicación distintos modelos de ASR para realizar la transcripción de los audios y evaluar su desempeño en la realización de estas tareas para la población de estudio. El objetivo es evaluar y comparar distintos modelos para el caso de uso tratado en esta tesis.
- Integrar a la aplicación modelos de diarización[3] para identificar hablantes en los audios y evaluar su desempeño en la realización de estas tareas para la población de estudio.
- Desarrollar herramientas que permitan el análisis y la evaluación del sistema implementado.
- Evaluar la utilidad de los resultados para el proyecto de investigación, proponiendo posibles líneas de acción a futuro a partir de los resultados obtenidos.
- Cuando sea posible, aplicar técnicas de *transfer learning* y *data augmentation* para evaluar tanto a nivel de conjunto como a nivel individual para cada niño/a, los desempeños de los modelos utilizados.

Las tareas de ASR se realizaron con dos de los sistemas de estado del arte disponibles, uno de código abierto y otro comercial:

- Open AI Whisper (Radford et al., 2022).

[3]Diarización es la tarea de añadir etiquetas de interlocutor a una transcripción o responder a la pregunta ≪¿quién habló cuándo?≫.

- Google Cloud Speech-to-Text API[4].

Las preguntas de investigación de las cuales parte esta tesis, son:

1. ¿Los modelos de estado del arte conservan un desempeño similar sobre poblaciones con patologías en el habla para tareas de ASR?

2. ¿Un *dataset* con solo algunas horas de audio con etiquetas supervisadas, es suficiente para aplicar transferencia de conocimiento a modelos de ASR pre entrenados?

3. ¿Aplicar *data augmentation* ayuda a mejorar el desempeño cuando no hay grandes cantidades de información supervisada en tareas de ASR sobre la población de estudio?

4. ¿Aplicar filtros para reducir niveles de ruido antes de aplicar tareas de inferencia mejora el desempeño de los modelos?

A la luz de estas preguntas y los resultados obtenidos, los modelos propuestos en esta tesis tienen como objetivo final integrarse al proyecto **DECILE**, un software diseñado en el marco del equipo de investigación de la Dra. Carolina Gattei. Dicho software tiene como objetivo evaluar integralmente el desarrollo del lenguaje con el fin de proporcionar a los profesionales de la salud, educadores y legisladores información de diagnóstico adicional, permitiéndoles ofrecer recomendaciones apropiadas y oportunas a los padres/madres cuyos hijos/as han recibido un diagnóstico de sordera.

Actualmente, el equipo de investigación a cargo del proyecto se está preparando para el lanzamiento de **DECILE** en las escuelas.

La aplicación desarrollada en el marco de esta tesis se integrará al proyecto **DECILE** para estimar el desempeño de niños y niñas con patologías en el habla y la audición sobre diferentes tareas lingüísticas, permitiendo la transcripción automática de los audios introducidos en la aplicación y generando métricas que serán utilizadas por tareas de evaluación de la comprensión de vocabulario oral y la comprensión y producción morfosintáctica. La calidad de estos modelos será evaluada por la aplicación propuesta y por especialistas con conocimiento de dominio. Una vez integrados y evaluados estos modelos, se podrá analizar el impacto relativo de las diferentes variables de salud y sociológicas en el rendimiento lingüístico. Estos hallazgos tienen el potencial de proporcionar información valiosa para el diseño de programas de intervención específica.

[4]https://cloud.google.com/speech-to-text

2. Datos

Esta sección describe, en primer lugar, el corpus de datos de entrevistas empleado en esta tesis junto con los criterios utilizados para la creación del *ground truth*. En segundo lugar, se describe el *dataset* creado a partir del corpus para realizar los distintos experimentos.

2.1. Corpus de entrevistas

El corpus inicial de entrevistas contiene 65 sesiones en formato comprimido MP3, que dan un total de 8 horas y 59 minutos de audio. Las entrevistas fueron registradas con un grabador portátil y están anonimizadas para resguardar la identidad de las personas entrevistadas. Las grabaciones tienen mucha variabilidad en cuanto a la calidad del audio, presentando distintos niveles de ruidos, reverberaciones y saturaciones. Esto constituye un desafío adicional para su procesamiento efectivo dado que no todos los modelos son pre entrenados con datos que tengan estas distorsiones.

Debido a restricciones de tiempo y a la cantidad de entrevistas disponibles, la transcripción de los audios fue realizada en colaboración por un anotador pago en el marco del equipo de investigación, especialmente entrenado para la tarea y según los criterios de transcripción que se detallan a continuación. Por otro lado, de las 65 sesiones disponibles, en 160 horas de trabajo se llegaron a terminar 47 entrevistas, es decir, un 72.3 % del corpus inicial.

El conjunto de archivos de audio de las entrevistas y las transcripciones realizadas por el anotador componen el *ground truth*. Es decir, la fuente de verdad que será utilizada a lo largo de este trabajo para comparar las inferencias de los distintos modelos aplicados con información depurada de manera supervisada y que se toma por correcta. Los criterios definidos para realizar las transcripciones se detallan en el Anexo B.

2.2. Metainformación

Como parte de las tareas involucradas en la construcción del *dataset*, no solo se necesita identificar y alinear temporalmente las palabras que se dicen en la entrevista sino, también, identificar quién se encuentra hablando y a qué pregunta del test CELF 4 pertenece ese segmento. Por lo tanto, para ordenar la información de manera estandarizada y que permita análisis posteriores, cada archivo que represente el *ground truth* de una entrevista contará con tres capas de información que se detallan a continuación.

1. A qué pregunta del test CELF 4 se encuentra asociada la ventana de tiempo marcada en el archivo de audio. Esta capa de información se agrega con el objetivo de realizar análisis comparativos detallados en etapas posteriores. Por ejemplo, evaluar la performance por dificultad de la pregunta. Las etiquetas definidas son:
    - tX: Donde t corresponde a preguntas de prueba del test CELF 4 que se realizan antes de comenzar y X indica el número de pregunta que se está realizando.
    - qX: Donde q corresponde a preguntas del test CELF 4 y X indica el número de pregunta que se está realizando.
    - ignore: Sección que se omite del análisis

2. Diarización de interlocutor/a. En esta capa se indicará qué interlocutor se encuentra hablando en la ventana de tiempo marcada. Las etiquetas definidas son:

    - speaker1: Se encuentra hablando la entrevistadora
    - speaker2: Se encuentra hablando el/la niño/a
    - ignore: Sección que se omite del análisis

3. Transcripción a nivel de palabra. En esta capa se indican las palabras que cada interlocutor dice en el momento que se encuentra hablando. Las etiquetas definidas son:

    - Vocablo válido del idioma español
    - ?: No se puede asociar lo escuchado con un vocablo válido del idioma español
    - <Vocablo válido del idioma español> / ¡Vocablo válido del idioma español¿: Cuando no se puede distinguir entre dos palabras válidas
    - <Vocablo válido del idioma español> + ?: Cuando se transcribe una palabra pero no es clara.
    - ignore: Se omite del análisis. Silencios, ruido u otros casos que no pueden ser utilizados para el proceso de evaluación o reconstrucción de la entrevista

Con estas tres capas de información, se puede desarrollar un sistema de evaluación de modelos de diarización y ASR respectivamente ya que se podrán comparar las inferencias de los modelos con las capas generadas.

La transcripción de los audios se realizó con el software Praat[5] por ser una herramienta de gran aceptación en ámbitos académicos.

En la Figura 1 se ilustra la interfaz de Praat y las capas de información detalladas.

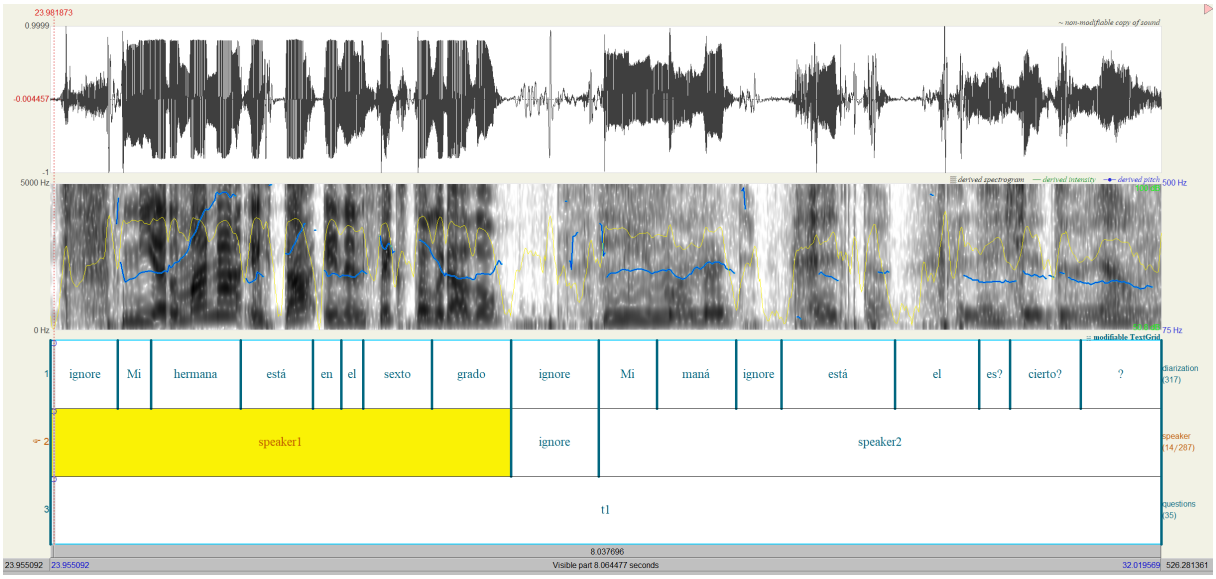

Figura 1. Capas de información que se agregan al audio de cada entrevista para la pregunta número 1 de tipo test del CELF 4.

[5]https://www.fon.hum.uva.nl/praat/

El resultado de esta tarea se guarda en formato TextGrid; dentro de los datos disponibles se encuentran las marcas de tiempo (*timestamps*) de cada recorte realizado en cada una de las capas.

En resumen, el conjunto de audios y archivos TextGrid componen el corpus de datos crudos a utilizar. A continuación, se detalla un ejemplo de la información en el formato mencionado.

```
    item []:
            item [1]:
                class = "IntervalTier"
                name = "RO_H23"
                xmin = 0
                xmax = 332.1850113378685
                intervals: size = 274
                intervals [1]:
                    xmin = 0
                    xmax = 12.110446116448076
                    text = "ignore"
                intervals [2]:
                    xmin = 12.110446116448076
                    xmax = 12.311797120107716
                    text = "Mi"
                intervals [3]:
                    xmin = 12.311797120107716
                    xmax = 12.926143834431336
                    text = "hermana"
                intervals [4]:
                    xmin = 12.926143834431336
                    xmax = 13.428691353696912
                    text = "está"
                intervals [5]:
                    xmin = 13.428691353696912
                    xmax = 13.636208130375207
                    text = "en"
    ...
```

Figura 2. Formato de *ground truth* utilizado por Praat.

Se describen algunos atributos importantes del archivo de *ground truth* que se muestra en la Figura 2.

- name: indica la capa de diarización. En este caso, es la correspondiente a la transcripción de palabras.
- xmin: timestamp que indica el comienzo de la capa
- xmax: timestamp que indica el final de la capa
- intervals: Cuantos intervalos tiene la capa
- intervals [i]: Listado de todos los intervalos marcados en la capa con su respectiva información
  - xmin: timestamp que indica el comienzo del intervalo
  - xmax: timestamp que indica el final del intervalo
  - text: La palabra que se transcribe para el intervalo

2.3. Dataset

En la sección anterior se describieron detalles y criterios de la información necesaria para producir un *ground truth* con todos los atributos necesarios para el análisis propuesto. Sin embargo, esa información no se encuentra procesada y es insuficiente para el objetivo de evaluar y/u optimizar modelos de ASR. A continuación, se describen los pasos realizados para conformar el *dataset* final utilizado a base del corpus de información compuesto por los archivos de audio de las entrevistas y los archivos de *ground truth* generados con Praat.

El objetivo, entonces, es crear un *dataset* en base a recortes de las entrevistas donde haya audio aislado por pregunta del CELF 4 y por interlocutor junto con un archivo de metainformación que establezca una relación entre esos recortes y una transcripción supervisada. Un ejemplo de este archivo de relación se puede observar en la Figura 3.

```
{
   "RO_H23_Entrevistadora_6.wav": "Mi hermana está en el sexto grado",
   "RO_H23_Alumne_7.wav": "Mí mamá está en sexto grado",
   "RO_H23_Entrevistadora_10.wav": "Enseña lectura el señor Lopez",
   "RO_H23_Alumne_11.wav": "Enseña lectura el señor pez",
   "RO_H23_Entrevistadora_15.wav": "No terminaron los niños la prueba",
   "RO_H23_Alumne_16.wav": "No terminaron los niños de la prueba",
   "RO_H23_Entrevistadora_19.wav": "La bebita de Carmen jugó con la muñeca",
   "RO_H23_Alumne_20.wav": "La cada me/vez es jugo está con la muñeca",
    ...
}
```

Figura 3. Archivo que establece una relación entre transcripción supervisada de preguntas del test CELF 4 y archivos de audio.

Este formato es ideal tanto para el proceso de *fine tuning* de modelos como para el de evaluación, ya que se tiene el recorte del audio con su etiqueta (transcripción) correspondiente.

Dado que las tareas de optimización de modelos se realizarán con librerías del ecosistema HuggingFace[6], el dataset se guarda utilizando herramientas del mismo ecosistema.

El proceso antes mencionado se resume en la Figura 4. En una primera instancia, se leen y se procesan los archivos de las entrevistas y los archivos de ground truth en formato TextGrid. Luego, se realizan las siguientes tareas de procesamiento.

Loader

- Se recortan los archivos de audio en segmentos por entrevista, interlocutor y pregunta
- Normalización
  - Se utiliza solo un canal de audio
  - Se realiza downsampling a 16KHz ya que es la frecuencia de los archivos con los que fueron entrenados los modelos de ASR utilizados
- Se construye el archivo de relación entre audio y transcripciones

Builder

- Se aplican filtros sobre las transcripciones (remoción de caracteres especiales /, ?, ignore, etc.)
- Conversión de archivos generados en etapa anterior a formato columnar con Apache Arrow.
- Se guarda el *dataset* para su uso y eventual distribución.

Figura 4. Flujo de procesamiento para creación del *dataset*.

Los atributos del *dataset* que se observa en la Figura 5 se detallan a continuación

- filename: Nombre del archivo de audio. Este archivo es un segmento de unos pocos segundos donde se enuncia una oración del test CELF4

[6]https://huggingface.co/

- audio: El audio en formato vector numpy
- sentence: Pregunta del test CELF4 que hace de ground truth
- speaker: Quien se encuentra hablando en el audio. Entrevistadora (speaker1) o entrevistado/a (speaker2)
- duration: Duración del segmento de audio en segundos
- filesize: Tamaño del archivo en MB
- channels: Cantidad de canales que tiene el archivo de audio
- sample_rate: Tasa de muestreo del archivo de audio
- bitrate: Tasa de bit del archivo de audio
- word_count: Cantidad de palabras que se dicen en el archivo de audio

Dataset Preview

filename (string)	audio (audio)	sentence (string)	speaker (string)	duration (float64)	filesize (float64)	channels (int64)	sample_rate (int64)	bitrate (int64)	word_count (int64)
"data/dataset /RO_H15_Entrevistadora_24.wav"	0:00 / 0:00	"La bebita de Carmen jugó con la muñeca"	"speaker1"	3.212993	0.2703	1	44,100	705,600	8
"data/dataset/RO_H21_Alumne_94.wav"	0:00 / 0:00	"La ropa lo vea no fue cae sábana en los niño"	"speaker2"	6.220998	0.523315	1	44,100	705,600	13
"data/dataset /RO_H21_Entrevistadora_105.wav"	0:00 / 0:00	"Porque los niños están cansados se van a…	"speaker1"	4.299002	0.361649	1	44,100	705,600	10
"data/dataset /RO_H20_Entrevistadora_96.wav"	0:00 / 0:00	"El señor que trae el correo a mi casa es mi…	"speaker1"	4.199002	0.353237	1	44,100	705,600	12
"data/dataset /RO_H02_Entrevistadora_59.wav"	0:00 / 0:00	"No terminaron los niños la prueba"	"speaker1"	2.369002	0.199308	1	44,100	705,600	6
"data/dataset /RO_H23_Entrevistadora_57.wav"	0:00 / 0:00	"María preparó la cena y luego lavó los…	"speaker1"	3.747007	0.315218	1	44,100	705,600	9
"data/dataset/RO_H01_Alumne_6.wav"	0:00 / 0:00	"Mi enana está se "	"speaker2"	2.219002	0.186691	1	44,100	705,600	5
"data/dataset/RO_H01_Alumne_11.wav"	0:00 / 0:00	"es y no puedo"	"speaker2"	3.242993	0.272823	1	44,100	705,600	5
"data/dataset/RO_H35_Alumne_14.wav"	0:00 / 0:00	"No terminaron los niños la prueba"	"speaker2"	2.669002	0.224543	1	44,100	705,600	6

Figura 5. *Dataset* disponible para uso en tareas de inferencia y evaluación.

2.4. Análisis exploratorio

De las 47 entrevistas procesadas que componen el *ground truth*, se obtiene un total de 6 horas y 21 minutos de audio, al extraer el contenido que no corresponde a preguntas del CELF 4, el tiempo neto de evaluación y entrenamiento es aún menor.

En la Tabla 1 se puede observar la comparación entre el corpus inicial (completo) de los materiales registrados por el equipo de investigación, el corpus procesado y la conformación del *dataset* sobre el cual se basa el análisis aquí presentado.

Corpus			
	Completo	**Procesado**	**Dataset**
Sesiones	65	47	47
Audio	8 horas y 59 minutos	6 horas y 21 minutos	2 horas y 19 minutos
Tamaño	2.7GB	1.9GB	704MB
Detalles	NA	NA	1793 segmentos de audio con sus respectivas etiquetas supervisadas Speaker1: 902 Speaker2: 891

Tabla 1. Corpus de entrevistas

En las Figuras 6 y 7 se observa la duración de cada entrevista del *dataset* expresada en minutos y su distribución.

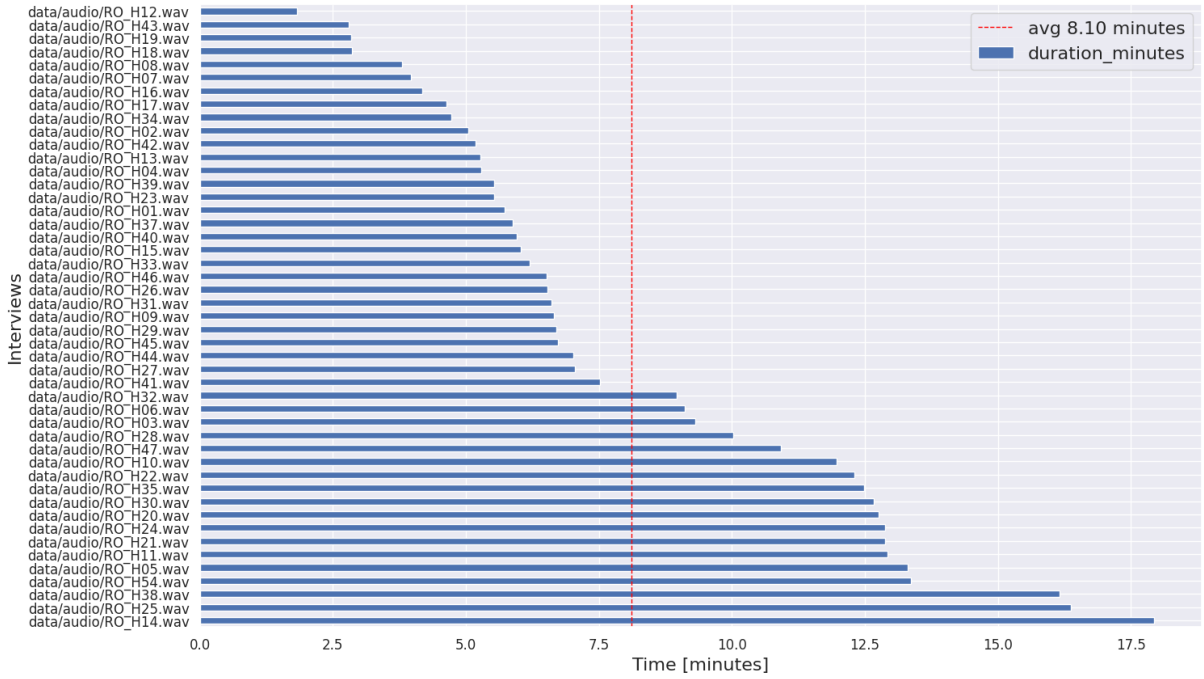

Figura 6. Tiempo de duración de entrevistas.

En la Tabla 2 se pueden ver atributos de los archivos que fueron procesados para adaptar los mismos a las tareas de inferencia y evaluación.

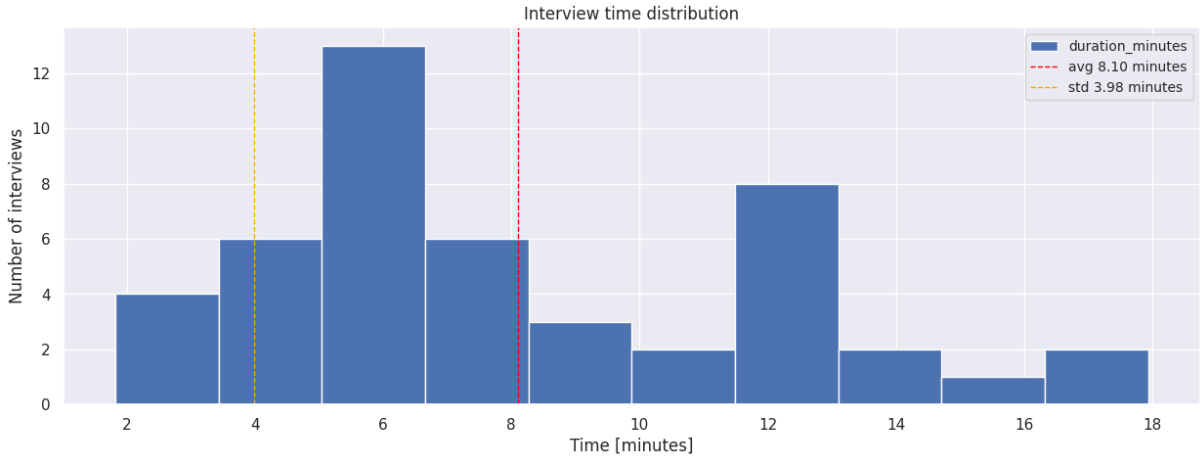

Figura 7. Distribución de duración de entrevistas.

	Entrevistas Originales	**Entrevistas Procesadas**
audio_format	MP3	WAV
bit_depth	16	16
channels	2	1
sample_rate	44100Hz	16000Hz

Tabla 2. Atributos principales de las entrevistas

Otro aspecto que conforma parte del análisis de los datos es la representación vectorial de los archivos de audio. Cada elemento del vector que los representa indica la amplitud de la señal de audio en un instante dado. El tamaño de muestras está relacionado a la longitud del archivo de audio y a la tasa de muestreo. De esta manera, se puede extraer una representación temporal del audio como se muestra en la Figura 8.

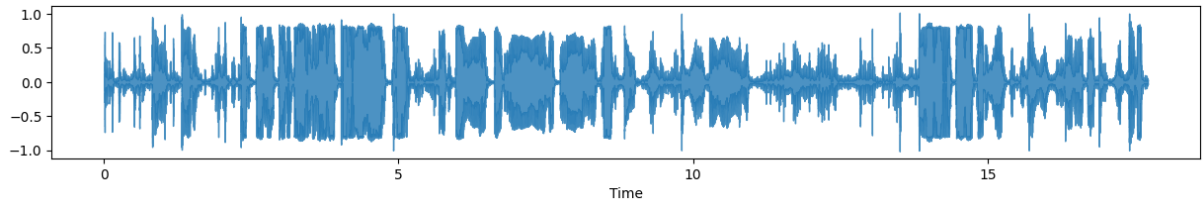

Figura 8. Forma de onda del audio correspondiente a entrevista RO_H05.wav del *dataset*.

Por otro lado, se puede extraer información en el dominio de la frecuencia de la señal de audio aplicando, por ejemplo, una transformada de Fourier discreta a las representaciones vectoriales en tiempo de los audios. Para obtener esta información a lo largo del tiempo podemos representar estas múltiples transformadas con un espectrograma log Mel como el que se muestra en la Figura 9. Este espectrograma aplica, además, la escala logarítmica de Mel para que los cambios en la frecuencia sean proporcionales a cómo los percibe un oído humano.

Estas representaciones numéricas del audio (tiempo y frecuencia) son importantes ya que son las características (*features*) que utilizan los modelos de ASR durante su entrenamiento y para realizar tareas de inferencia.

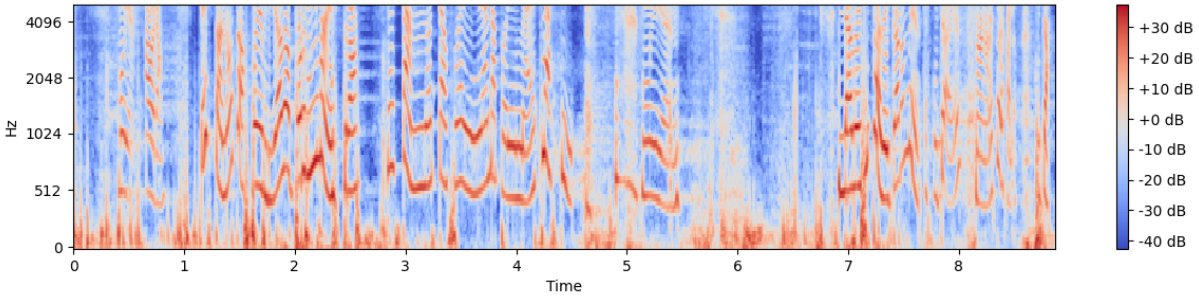

Figura 9. Espectrograma log-mel de entrevista RO_H05.wav del *dataset*.

3. Metodología

El diseño metodológico de esta tesis se puede separar en dos secciones que fueron implementadas como flujos de trabajo separados dentro de módulos de una misma aplicación.

1. Construcción y evaluación de sistema de procesamiento automático del habla
2. Fine tuning y optimización de modelos de ASR

El objetivo es construir, a partir de archivos de audio, transcripciones enriquecidas de las entrevistas como la que se observa en la Figura 10 y que incluyan para cada segmento de habla: el tiempo inicial, el tiempo final, las palabras dichas y la identidad del hablante. Además, se generarán distintas métricas que permitan evaluar cuantitativamente el desempeño del sistema.

Esta transcripción enriquecida podrá ser utilizada por el equipo de investigación con el objetivo de mejorar el proceso de trabajo así como dar más herramientas para futuras líneas de investigación. Por otro lado, utilizando las transcripciones que hacen de *ground truth*, se realizó la comparación de las salidas de los modelos de diarización y reconocimiento automático del habla para entender su desempeño sobre el dominio de este trabajo.

Dado que la tarea de reconstrucción de la entrevista requiere identificar hablantes se utilizaron, además de los modelos de ASR mencionados inicialmente, modelos especializados en la tarea de diarización.

Por último, y en los casos en que el modelo lo haya permitido, se aplicaron técnicas de transferencia de conocimiento para entender qué niveles de mejora se pueden lograr sobre la población de estudio de este trabajo.

3.1. Sistema de reconocimiento automático del habla

El sistema de reconocimiento automático del habla desarrollado integra distintos modelos y herramientas cuyos resultados tienen dependencia, es por esto que se define un *pipeline* para su implementación. A continuación, se describen en detalle cada una de las etapas que componen al *pipeline* desarrollado para generar transcripciones enriquecidas y evaluar los distintos modelos utilizados. La estructura del *pipieline* puede verse en la Figura 11. La metodología adoptada extiende el trabajo propuesto por Dernoncourt et al., 2018 para evaluar modelos de ASR. Los resultados de la ejecución de las tareas dentro de esta sección se dividen en:

```
0.818s, 5.425s - SPEAKER_00: Muy bien, ahora, esta nota fue enviada por mi
↪  maestra.
5.425s, 10.437s - SPEAKER_01: Esta nota fue enlada por mi maestra así.
10.437s, 12.985s - SPEAKER_00: Muy bien.
13.778s, 21.035s - SPEAKER_00: La bebita de Carmen... Escucha, eh. No, no,
↪  esto déjalo. Escucha. La bebita de Carmen jugó con la muñeca.
21.035s, 25.591s - SPEAKER_01: La bebida de carne en jugo con la muñeca.
25.591s, 32.915s - SPEAKER_00: ¡Muy bien! ¡Sí, Ella! Escucha, ¿eh? Mi amigo
↪  no llevó su almuerzo a la escuela.
32.915s, 38.551s - SPEAKER_01: Mi amigo no se porta almuerzo a la escuela.
38.551s, 44.508s - SPEAKER_00: ¡Bien, una genia! A ver, ¿se le olvidó al
↪  estudiante hacer su tarea?
44.508s, 49.992s - SPEAKER_01: El segundo es el estudianta hacer su tarea.
↪  Muy bien.
```

Figura 10. Reconstrucción de entrevista.

- Reconstrucción de la entrevista que se está analizando con marcas de tiempo y reconocimiento de hablantes
- Obtención de las métricas que evalúen el desempeño de los modelos.

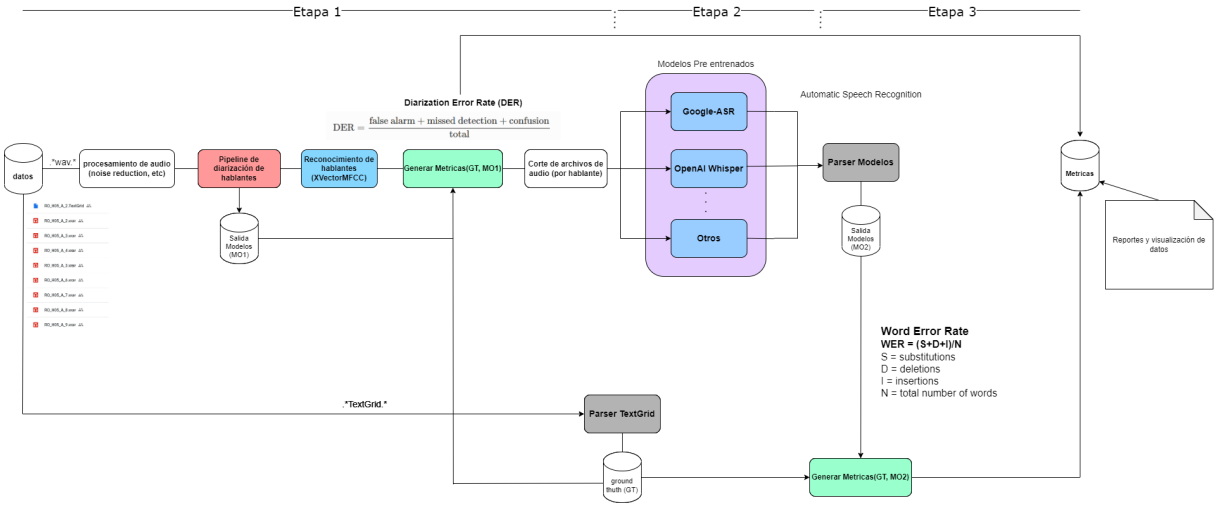

Figura 11. Diagrama de flujo del pipeline de transcripción implementado.

3.1.1. Etapa 1 - Preprocesamiento de datos y diarización

A continuación, se detallan individualmente cada una de las tareas de la etapa uno del *pipeline* de reconocimiento automático del habla.

3.1.1.1 Procesamiento de audio

Como se ha mencionado anteriormente, las grabaciones fueron realizadas en espacios públicos y con un grabador genérico, tienen saturación y ruidos de fondo. Una de las

hipótesis planteadas al comienzo de este trabajo es que estas condiciones pueden correlacionarse con una caída en el desempeño de los modelos. Por este motivo, antes de comenzar con las tareas de inferencia se aplica, opcionalmente, una etapa de filtrado de ruido en las entrevistas. De esta manera, se podrán comparar los resultados con y sin filtros para entender su impacto.

Se utiliza la librería *noisereduce* (ver Anexo G) que implementa un algoritmo de atenuación de ruido basado en la técnica spectral gating. Funciona computando el espectrograma de la señal y estimando un umbral de ruido. Luego, comparando el umbral de ruido con el espectro de la señal se obtiene una máscara de atenuación que será aplicada posteriormente a la señal original. La librería soporta aplicar el método en versión estacionaria, es decir, un solo umbral de ruido para toda la señal o no estacionaria, en donde el umbral se va recalculando dinámicamente por ventanas de tiempo. En esta etapa se utilizará el método estacionario.

En la Figura 12 se observa el diagrama de amplitud en el dominio del tiempo para un recorte de la entrevista RO_H05. En la parte superior de la figura se observa la señal tal cual fue tomada por el grabador, en la parte inferior vemos la misma señal después de pasar por el filtro de ruido, donde se ve atenuación en amplitud y que se suavizan los cambios abruptos (recorte de frecuencias altas).

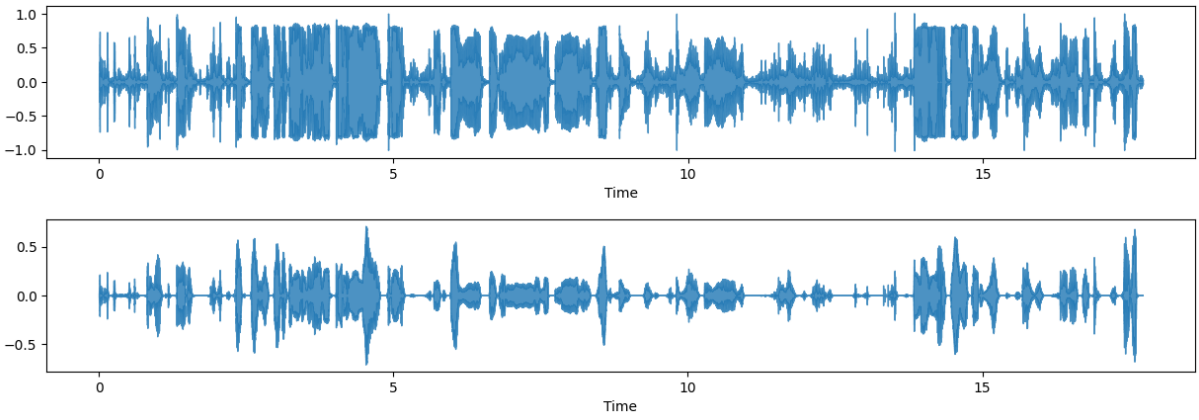

Figura 12. Señales en dominio tiempo para el mismo archivo sin y con filtro de reducción de ruido.

Por otro lado, en la Figura 13 vemos los mismos dos escenarios pero en el dominio de la frecuencia a través de un espectrograma. En este caso se ve claramente el efecto del filtro, donde no hay densidad de potencia espectral en regiones en las que actuó el filtro.

Esta tarea se puede habilitar o deshabilitar antes de comenzar con la ejecución del *pipeline*.

3.1.1.2 Modelos y pipelines de diarización

La diarización consiste en identificar **quién** dice **algo** y **cuándo** a lo largo de una conversación. Teniendo como objetivo final la reconstrucción de las entrevistas en un formato que contenga toda la información que se muestra en la Figura 10, antes de poder aplicar un modelo especializado en tareas de reconocimiento automático del habla para extraer las palabras que se dicen en el audio, es necesario realizar dos acciones. Por un lado, identificar regiones donde hay actividad y, por el otro, generar algún mecanismo

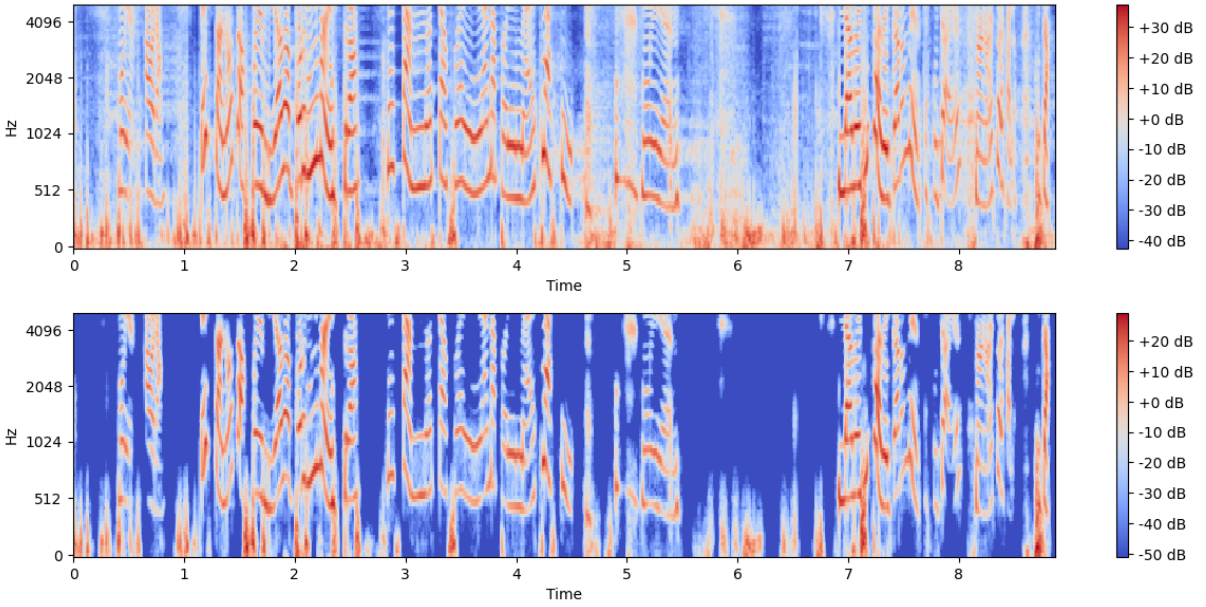

Figura 13. Señales en el dominio de la frecuencia para el mismo archivo sin y con filtro de reducción de ruido.

para identificar la firma de los distintos hablantes a lo largo de la conversación de modo que sea posible asignar cada palabra a uno de ellos.

Para resolver estas necesidades, se utiliza el *toolkit* open source pyannote.audio (Bredin et al., 2019) que implementa *pipelines* de diarización de hablantes con modelos pre entrenados y con resultados de estado del arte en estas tareas. Además, provee un framework de evaluación con una interfaz que permite utilizar métricas que son estándar de facto para la evaluación de la tarea (Bredin, 2017) como el Diarization Error Rate (DER).

La tarea de diarización requiere de la implementación de un *pipeline* (Figura 14) que, a su vez, implementa tareas y modelos especializados como:

- feature extraction: Extracción de características representativas del audio que serán utilizadas para el procesamiento e inferencia en tareas posteriores. Generalmente se utilizan MFCC (mel frequency cepstral coefficients)
- voice activity detection: tarea de clasificación binaria que establece si hay o no actividad en la ventana de tiempo analizada
- segmentation: Identificar probabilidad de que uno o más hablantes se encuentren activos en una ventana de tiempo
- speaker embeddings: Vectores extraídos a partir de features del audio que permiten identificar a un hablante en particular
- clustering: Normalizar etiquetas de hablantes a nivel global para toda la entrevista
- resegmentation: Asignar nuevamente etiquetas a los hablantes utilizando información de embeddings y clustering

Estas tareas encadenadas dan como resultado la diarización de la entrevista. Como puede observarse en el diagrama de segmentación de la Figura 15, si bien en la ventana

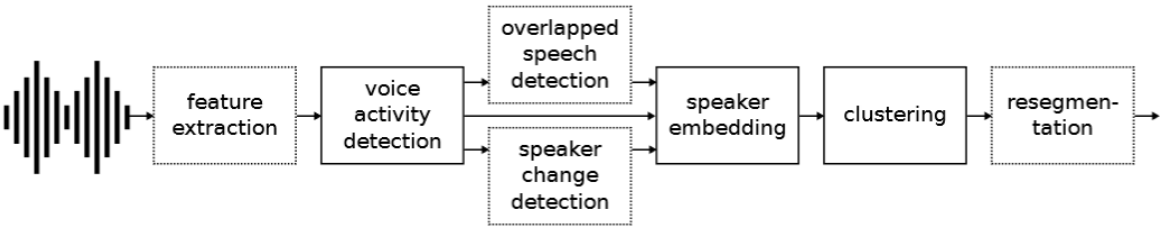

Figura 14. Pasos del *pipeline* del modelo de diarización. (Fuente: Bredin, 2017)

de tiempo entre los 4 y 8 segundos hay un cambio en el hablante, no lo hay en el color que representa la segmentación (azul). Esto se debe a que el análisis se realiza en ventanas de 5 segundos y la segmentación es local a esa ventana, por lo que si bien se tiene una misma etiqueta en distintos intervalos de tiempo podrían referirse a hablantes distintos (Bredin y Laurent, 2021). Es por este motivo que luego se ejecutan tareas de *clustering* y resegmentación, que permiten corregir este problema y obtener una asignación coherente de las etiquetas.

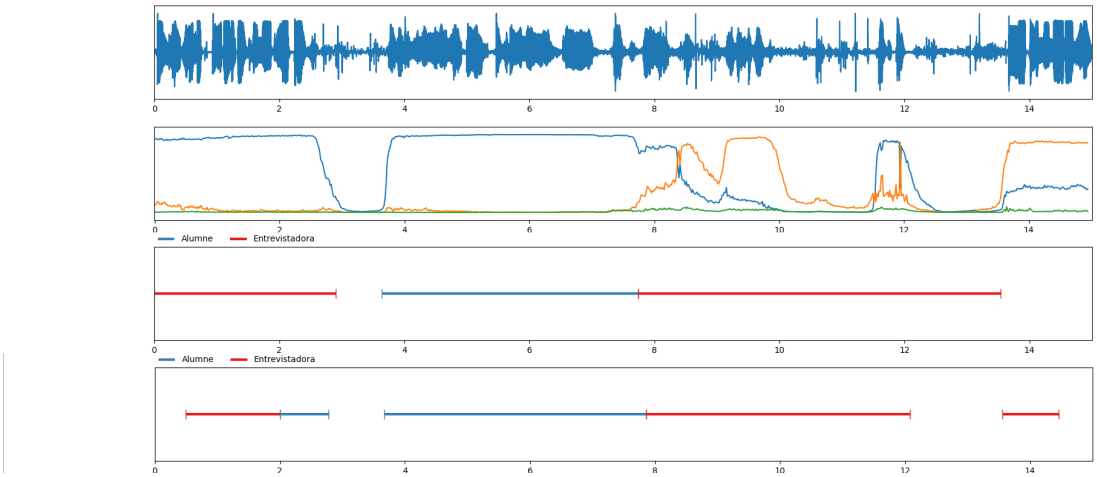

Figura 15. Diagramas de un extracto de la entrevista RO_H23. Señal en tiempo, segmentación, diarización según *ground truth*, diarización según *pipeline* respectivamente.

Este pipeline de diarización de hablantes es aplicado sobre las entrevistas que salen de la etapa de procesamiento anterior. La salida de esta tarea consiste en:

- Diagramas de segmentación y diarización
- Segmentos con actividad e información de speaker que serán utilizados en tareas posteriores
- Métricas de desempeño del pipeline

3.1.1.3 Métricas y evaluación

La métrica considerada un estándar de facto para evaluar modelos de diarización de hablantes es el Diarization Error Rate (DER) (Liu y Yu, 2022)

$$DER = \frac{false\,alarm + missed\,detection + confusion}{total} \quad (1)$$

Donde *false alarm* es el tiempo de regiones sin actividad que fueron clasificadas como con actividad, *missed detection* es el tiempo de regiones con actividad que fueron clasificadas como sin actividad, *confusion* es el tiempo de regiones donde se clasificó un *speaker* incorrecto y *total* es la duración de la entrevista.

En el caso de nuestro *pipeline* de trascripción, se toman los datos del *ground truth* de la capa 2 para la entrevista a procesar, se toma la salida del modelo de diarización y se computa el DER aplicando un método de *optimal mapping* en donde se establece una relación entre las etiquetas de ambas entradas que minimice el error.

Estas métricas serán guardadas en una base de datos para cada ejecución del *pipeline* con el objetivo de ser procesadas y analizadas en etapas posteriores.

3.1.1.4 Reconocimiento de hablantes basado en *embeddings*

El pipeline de diarización utilizado, si bien cuenta con la capacidad de identificar de manera consistente distintos interlocutores en un audio, no tiene memoria de lo aprendido cuando analizamos un nuevo archivo ya que el proceso de inferencia comienza de nuevo, así como el cálculo de *embeddings*, *features*, etc. De esta manera, la etiqueta que asigna dinámicamente el proceso de inferencia para una entrevista, podría cambiar para un mismo hablante en otra. Por ejemplo, en la entrevista RO_H05, el modelo podría asignar el label SPEAKER_00 a la entrevistadora, y en la entrevista RO_H06 podría asignarle el label SPEAKER_01 como se observa en la Tabla 3. Esta situación podría representar un problema para el análisis ya que si intercambiamos la etiqueta que identifica a cada parte a lo largo de los archivos se llegaría a resultados incorrectos o no representativos para la población de estudio.

Entrevista	**Label Entrevistadora**	**Label Entrevistada/o**
RO_H05	SPEAKER_00	SPEAKER_01
RO_H06	SPEAKER_01	SPEAKER_00

Tabla 3. Asignación dinámica de etiquetas en proceso de diarización

Para resolver este problema, se utiliza el modelo *pyannote/embedding*[7], una red neuronal (Tripathi et al., 2020 y Snyder et al., 2018) pre entrenada que utiliza características MFCC[8] para obtener representaciones de la voz en un espacio vectorial de longitud fija (x-vectors) a los que se les puede aplicar una métrica de disimilitud como distancia coseno. La distancia coseno es inversa a la proyección de un vector sobre el otro como se observa en la Figura 16. Es decir, cuando la distancia aumenta, los vectores son más distintos entre sí (mayor ángulo de separación entre ellos).

[7] https://huggingface.co/pyannote/embedding
[8] https://es.wikipedia.org/wiki/MFCC

$$cosine\,similarity = Sc(a,b) = \cos\theta = \frac{a \cdot b}{||a|| \cdot ||b||} = \frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{i=1}^{n} a_i^2}\sqrt{\sum_{i=1}^{n} b_i^2}} \tag{2}$$

$$cosine\,distance = 1 - Sc(a,b) \tag{3}$$

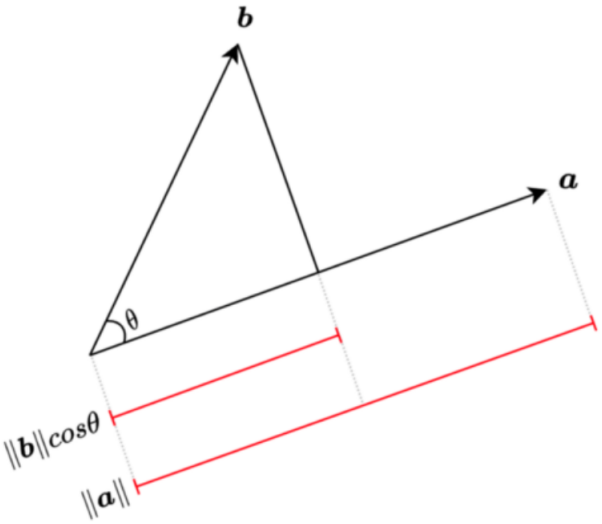

Figura 16. Proyección de vectores en 2 dimensiones.

Al poder vectorizar características (*features*) de los *speakers* en regiones de la entrevista, y confiando en que la generación de estos vectores es lo suficientemente representativa del hablante, se puede pre calcular el *embedding* para segmentos de audio de la entrevistadora y, luego, utilizar estos valores para calcular distancias coseno que se obtienen de la salida de los modelos de diarización para identificar y corregir las etiquetas en cada ejecución del *pipeline* sobre las entrevistas. El proceso implementado sigue el código que se muestra en 17.

En este punto, ya contamos con la diarización de la entrevista, y una asignación consistente de las etiquetas.

```
# calcular embedding para entrevistadora
reference_embedding = Embedding('entrevistadora.wav')

diarization = Diarization('interview.wav')

# calcular embeddings para cada segmento con actividad
for speaker, segment in diarization:
        data[speaker]['embeddings'].append(Embedding(segment))

speaker_a.distance(reference_embedding, data[speaker_a]) # calcula
↪  distancia promedio a referencia
speaker_b.distance(reference_embedding, data[speaker_b]) # calcula
↪  distancia promedio a referencia

speaker_a.reference_similarity(speaker=speaker_b) # Encuentra menor
↪  distancia promedio a referencia

update_labels(speaker_a, speaker_b) # actualiza labels
```

Figura 17. Código para reconocimiento de hablante utilizando *embeddings*.

3.1.2. Etapa 2 - Inferencia y modelos ASR

3.1.2.1 Inferencia

En esta instancia el *pipeline* de trascripción toma los segmentos con actividad e información de los *speakers* producto de la etapa anterior y, en base al modelo seleccionado, se corren tareas de inferencia por cada segmento y se agrupan en una estructura de datos que permita reconstruir la conversación y aplicar tareas de evaluación como se observa en el siguiente código 18.

```
for segment, speaker in diarization:
    translation = asr(audio=audio[segment], model=asr_model)

    annotations[filename]['data'].append(
         {
             "label":labels[speaker],
             "segment": (f'{segment.start}', f'{segment.end}'),
             "translation": '' if translation is None else translation,
         }
       )
```

Figura 18. Código inferencia en tarea de ASR.

La transcripción, producto de la inferencia, se realiza para los distintos modelos a evaluar y que son parametrizados al momento de ejecución del *pipeline*. En el contexto de esta tesis, se comparan distintas versiones del modelo OpenAI Whisper[9] y el servicio

[9]https://openai.com/research/whisper

Speech-to-Text de Google Cloud Platform[10] ya que se consideran herramientas que representan el estado del arte en términos de efectividad para tareas de ASR, una OpenSource y otra comercial.

3.1.2.2 Modelos de Automatic Speech Recognition

OpenAI Whisper

Whisper[11] es una red neuronal Seq2Seq con arquitectura Transformer Encoder-Decoder (Radford et al., 2022) pre entrenada y publicada como Open Source por OpenAI[12] que se especializa en tareas de ASR y Multilingual ASR.

Modelos anteriores a Whisper con resultados de estado del arte, como Wav2Vec 2.0 (Baevski et al., 2020) pre entrenan los modelos con miles de horas de audio sin etiquetas con el objetivo de tener un mapa de la estructura del lenguaje y, luego, optimizan el modelo con pocas horas de datos con etiquetas. Si bien este abordaje genera buenos resultados, el proceso de entrenamiento sobreajusta rápido y no generaliza bien con otros *datasets* (Geirhos et al., 2020). Esto se debe a que el aprendizaje supervisado se realiza sobre pocos datos y el no supervisado sobre *datasets* grandes (dado su bajo costo de adquisición). El objetivo de Whisper es solventar estas limitaciones y generar un modelo que generalice bien para cualquier escenario de manera *out of the box*, es decir, sin requerir *fine tuning*.

La limitante para lograr modelos robustos que se comporten bien *out of distribution*, es que se requieren mejores y más grandes *datasets*. La diferencia suele ser, entre datasets de aprendizaje supervisado versus no supervisado, de varios órdenes de magnitud (1 millón de horas versus 1000 horas por ejemplo).

El abordaje *weakly supervised speech recognition* de Whisper propone, entre otras cosas, enfocar esfuerzos en crear mejores datasets supervisados relajando el estándar de transcripciones 100 % validadas por humanos (Chen et al., 2021 y Galvez et al., 2021) haciendo uso de *pipelines* automatizados que permitan escalar a mayores cantidades de datos pero más ruidosos.

Para reducir esta brecha entre *datasets*, se estresa este abordaje para llegar a las 680K horas de audio supervisado. Esta metodología saca principalmente su información haciendo *web scraping* en internet de audios que tengan transcripción. Si bien la diversidad en la calidad del audio ayuda a que el modelo sea más robusto, no sucede lo mismo con varianza en la calidad de las transcripciones. Más aún, muchas transcripciones en internet no son generadas por humanos sino por otros sistemas de ASR. Recientes trabajos muestran que el entrenamiento usando *datasets* mixtos (humanos y sistemas de ASR) puede perjudicar la efectividad del modelo. Para resolver este problema se crean heurísticas para remover transcripciones generadas por modelos de ASR (Ghorbani et al., 2021).

Hay 5 versiones de Whisper publicadas con diferentes cantidades de parámetros y efectividad.

[10] https://cloud.google.com/speech-to-text
[11] Web-scale Supervised Pretraining for Speech Recognition
[12] https://openai.com/

Tamaño	Parámetros	English-Only	Multilingual	VRAM	Speed
tiny	39M	tiny.en	tiny	1GB	32x
base	74M	base.en	base	1GB	16x
small	244M	small.en	small	2GB	6x
medium	769M	medium.en	medium	5GB	2x
large	1550M	N/A	large	10GB	1x

Tabla 4. Versiones pre entrenadas de Whisper

Para concluir, Whisper incorpora una metodología para construir datasets supervisados varios órdenes de magnitud más grandes de los que había antes de su publicación. Esto genera un modelo más robusto y más generalizable que reporta un Word Error Rate (WER 3.1.3.1) de 3 % para español en promedio para los datasets evaluados.

Google Speech-to-Text

Es un servicio comercial desarrollado por Google que soporta tareas de ASR y Speaker Diarization en más de 125 idiomas. El servicio se expone a través de una REST API en la nube pública Google Cloud Platform (GCE). Al ser un servicio cerrado, no hay información respecto a arquitecturas o tecnologías utilizadas para su implementación.

3.1.3. Etapa 3 - Evaluación de modelos y análisis de resultados

3.1.3.1 Métricas utilizadas

Para evaluar los modelos elegidos y los modelos producto de etapas de optimización con respecto a las tareas de ASR, se utilizó la métrica WER. El WER es un estándar de facto en cuanto a métrica para la evaluación de sistemas de ASR (Morris et al., 2004) que deriva de la distancia de Levenshtein, pero en vez de centrarse en la distancia a nivel caracter, lo hace a nivel de palabra.

El WER se obtiene alineando las oraciones o secuencias de palabras a comparar y calculando la siguiente expresión.

$$WER = \frac{S + D + I}{S + D + C} = \frac{S + D + I}{N} \tag{4}$$

Donde

- S (Substitutions): Cantidad de palabras en una misma posición dentro de la oración que deben reemplazarse por otra
- D (Deletions): Cantidad de palabras que deben eliminarse de la oración
- I (Insertions): Cantidad de palabras que deben agregarse
- N: Cantidad de palabras en la oración de referencia

Las operaciones de sustitución, borrado e inserción, se computan considerando cómo llegar desde la referencia, es decir, el *ground truth* hasta la hipótesis (salida del modelo

que estamos analizando). Esto se muestra en el ejemplo de la Tabla 5:

Referencia	El señor López
Hipótesis	Es señal Lopez
Referencia normalizada	[['el', 'señor', 'lopez']]
Hipótesis normalizada	[['es', 'señal', 'lopez']]

Tabla 5. Comparación entre hipótesis y referencia para cómputo de WER

En este caso, para llegar desde la referencia 'el señor lopez' hasta la hipotesis 'es señal lopez', debemos realizar 2 operaciones de sustitución 'el'/'es' y 'señor'/'señal'. Por lo tanto, el WER es:

$$WER = \frac{S}{S+C} = \frac{2}{2+1} = 0.\hat{6} \tag{5}$$

Uno de los problemas que presenta esta métrica, es que no da información sobre la fuente de error en la tarea de transcripción. Por lo que un análisis en detalle debe realizarse para entender la fuente de un eventual problema. Otra consideración interesante respecto a esta métrica, es que puede ser arbitrariamente mayor a 1, dado que la cantidad de inserciones o eliminaciones puede ser mayor a la cantidad de palabras que hay en la oración de referencia. Esto puede pasar particularmente en sistemas de ASR que padecen el problema de alucinaciones. Será de especial importancia identificar estas situaciones ya que WER >> 1 pueden actuar de *outlier* y dificultar una evaluación justa de los modelos.

Hay otras métricas que se utilizan para evaluar la eficacia de los sistemas de ASR como el Word Information Lost (WIL) (Morris et al., 2004) que se basan en la proporción de información a comunicar y que son calculadas pero no analizadas en el contexto de este trabajo dado que las publicaciones que permiten análisis comparativos siempre refieren al WER.

3.1.3.2 Implementación de sistema de validación

Como se menciona en la sección de datos, la evaluación de los modelos se realizará sobre las porciones de audio que estén alineadas con preguntas del test CELF 4, ya que es donde se ha trabajado en la transcripción de las 3 capas de información durante la construcción del *ground truth*. La estructura que contiene la información que se utilizará durante las tareas de evaluación se observa en 19.

Teniendo la asociación entre entrevista, segmento de audio, pregunta del test CELF4 y la transcripción, se puede iterar sobre esta estructura y ejecutar el proceso de inferencia para cada uno de ellos.

Luego de realizar la transcripción con el modelo definido, deben realizarse tareas que normalicen la salida del modelo y la del *ground truth* con el objetivo de no generar falsos errores durante el cómputo del WER. Las transformaciones que se realizan apuntan a remover signos de puntuación, acentos y otros caracteres que no deberían ser tenidos en cuenta en el cómputo de errores. Estas transformaciones se detallan en la Tabla 6.

```
{
. . .
    "data/dataset/RO_H40_Entrevistadora_5.wav": "Mi hermana está en el
    ↪  sexto grado",  # Trial 1
    "data/dataset/RO_H40_Alumne_6.wav": "Mi hermana está sexto el grado",
    ↪  # Trial 1
    "data/dataset/RO_H40_Entrevistadora_8.wav": "Enseña lectura el señor
    ↪  López",  # Trial 2
    "data/dataset/RO_H40_Alumne_9.wav": "h Enseña les  López",  # Trial 2
    "data/dataset/RO_H40_Entrevistadora_11.wav": "No terminaron los niños
    ↪  la prueba",  # Question 1
    "data/dataset/RO_H40_Alumne_12.wav": "No terminaron la prueba los
    ↪  niño",  # Question 1
    "data/dataset/RO_H40_Entrevistadora_16.wav": "Fue puesta la carta por
    ↪  correo",  # Question 2
    "data/dataset/RO_H40_Alumne_17.wav": "Que la puerta de correo",  #
    ↪  Question 2
    "data/dataset/RO_H40_Entrevistadora_20.wav": "Esta nota fue enviada por
    ↪  mi maestra",  # Question 3
    "data/dataset/RO_H40_Alumne_21.wav": "Mareada estoy maestra mareada",
    ↪  # Question 3
    "data/dataset/RO_H40_Entrevistadora_24.wav": "La bebita de Carmen
    ↪  jugó con la muñeca",  # Question 4
    "data/dataset/RO_H40_Alumne_25.wav": "La carne está jugando con la
    ↪  muñeca",  # Question 4
    "data/dataset/RO_H40_Entrevistadora_28.wav": "Mi amigo no llevó su
    ↪  almuerzo a la escuela",  # Question 5
    "data/dataset/RO_H40_Alumne_29.wav": "Mi amigo no llegó a las cueva",
    ↪  # Question 5
. . .
}
```

Figura 19. Preguntas del test CELF 4 dentro del *dataset* para la entrevista RO_H40.

Transformación	**Descripción**
RemoveSpecificWords([“ignore”, “eh”, “h-”, “s-”, “e-”, “f-”, “i-”])	Remover palabras o códigos especiales que fueron definidos en la guía de transcripción. En casos como ignore, son porciones de audio que deben ser ignoradas pues no pudieron ser correctamente identificadas por la persona que realizó la transcripción. Otras, como h- son fragmentos que no se espera que el modelo pueda transcribir como se ha definido, y por ende no se los contempla a la hora de computar su eficacia.
ToLowerCase()	Se convierten todas las palabras a minúscula
RemoveMultipleSpaces()	Se remueven múltiples espacios en blanco y se reemplazan por solo uno
RemovePunctuation()	Remueve caracteres de puntuación (caracteres UNICODE sección P)
ReduceToListOfListOfWords	Se convierte la oración a una lista de palabras
Remover acentos	Se remueven acentos de vocales

Tabla 6. Transformaciones aplicadas sobre oraciones, tanto del *ground truth* como de las generadas por el modelo para poder computar métrica de evaluación.

Teniendo una estructura conveniente en los datos del *ground truth*, las transformaciones necesarias para normalizar las oraciones a comparar que permitan computar la métrica de evaluación correctamente, y las integraciones necesarias para realizar inferencia con los modelos a analizar, se implementa el siguiente código para computar el WER.

```
wer = []

for audio, sentence in dataset:
    hypothesis = asr(audio)
    reference = sentence
    wer.append(metric.wer(transform(reference), transform(hypothesis)))
```

Figura 20. Cómputo de WER para evaluación de modelos.

En la Tabla 7, se describe paso a paso el proceso de evaluación para una de las preguntas del test CELF 4.

Descripción	Resultado
Pregunta test CELF4	El gato que se sube a la mesa, es el más travieso.
Transcripción *ground truth* para respuesta de entrevistada	El gato sub que sube a la mesa es el mas travieso
Transcripción modelo	El gato sube sube a la mesa es el mas travieso
Transformación ground truth	[['el', 'gato', 'sub', 'que', 'sube', 'a', 'la', 'mesa', 'es', 'el', 'mas', 'travieso']]
Transformación modelo	[['el', 'gato', 'sube', 'sube', 'a', 'la', 'mesa', 'es', 'el', 'mas', 'travieso']]
WER	0.16666666666666666
hits	10
insertions	0
substitutions	1
deletions	1

Tabla 7. Pasos para el cómputo de WER para oración 6 del test CELF4 de la entrevista RO_H22.

3.1.3.3 Alucinaciones

El caso puntual de OpenAI Whisper, presenta ocasionalmente un comportamiento no deseado que se describe como alucinaciones (Radford et al., 2022). Este comportamiento tiende a agregar palabras u oraciones completas sin relación alguna a lo que se dice en el audio. Se le atribuye a datos de mala calidad durante el proceso de entrenamiento ya que el abordaje de Whisper se centra en relajar la calidad de esta información, a una mala respuesta del modelo a sonidos prolongados, situación que suele haber especialmente en la población de estudio, y a cortes abruptos en el audio, situación que puede presentarse en el caso de tener un alto DER en etapas iniciales del *pipeline*. El desafío adicional que presenta este problema, es que al tener un alto número de inserciones de nuevas palabras, el WER tiende a crecer a valores mucho mayores que 1. Estos valores se presentan como *outliers* y pueden desplazar mucho la performance media del modelo. En la Tabla 8 se describe un ejemplo de alucinación del modelo.

Pregunta	q1
Archivo	data/audio/RO_H38.wav
Hablante	speaker2 (entrevistado/a)
Inserciones	42
Oración	No terminaron los niños la prueba
Alucinación	no terminan lo fu Damiracto Nu ha guards parentsorden si ha problemos porque uno imp restoration y o sea lo que aun no tomen No Hawaii no usa ¿.....? No la tanta No se s devotee practice,o gosh keywords y hanyaulative technical 5 6 4 2 ipt DOT,e,re, fras, iptitä,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,Yo,,,,
WER	6.7142

Tabla 8. Descripción de alucinación producto del proceso de inferencia de modelo Whisper.

Debido a lo mencionado anteriormente, se propone una heurística sencilla para detectar estas alucinaciones del modelo. Dada la naturaleza de la métrica utilizada, las alucinaciones se pueden detectar haciendo foco en la cantidad de inserciones de palabras y definiendo un umbral que esté relacionado a la longitud de la oración de referencia.

$$threshold = len(reference.split('')) \cdot k \tag{6}$$

El valor k>1 hace referencia a qué proporción de palabras de más con respecto a la oración de referencia deben insertarse para determinar si se trata de una alucinación o no. Su valor es empírico y está sujeto a ser corregido. De esta manera, se computará el WER con y sin la heurística de alucinaciones y se calculará cuántas hay por cada uno de los hablantes para realizar análisis posteriores. En este trabajo se utilizó un k=1.5. Es decir, se tolera un 50 % de inserciones erróneas respecto al tamaño de la oración de referencia.

3.1.3.4 Visualización de datos y reportes

La última etapa del *pipeline* de reconocimiento automático del habla toma los datos y métricas computadas en las etapas anteriores y guarda todos los valores en una base de datos relacional para poder analizar posteriormente. Lo mismo sucede con el proceso de entrenamiento que se detalla en secciones posteriores como se observa en la Figura 21.

Dado que este trabajo fue concebido en el contexto de un proyecto de investigación cuyo alcance excede al de esta tesis, se elaboran una serie de tableros que proveen métricas sobre cada ejecución, tanto de las tareas de inferencia como de las tareas del proceso de entrenamiento, que podrán ser utilizados a futuro para continuar extendiendo y/o mejorando lo hecho hasta ahora. Es por esto que se puede observar en los esquemas de las tablas propuestas que existen atributos que no son utilizados.

Por otra parte, este abordaje permite recibir feedback inmediato ante la ejecución de los *pipelines*, lo que orienta más rápidamente el análisis en tiempo de desarrollo y conlleva a realizar ajustes de manera más precisa.

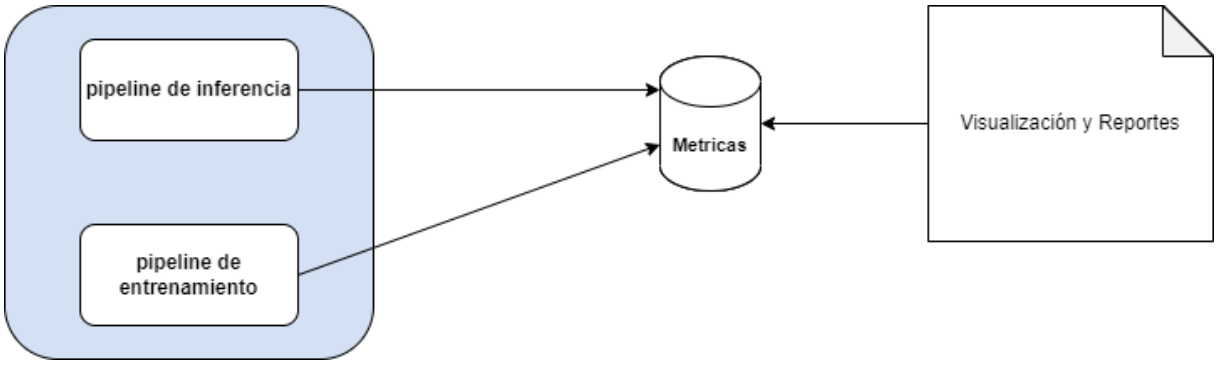

Figura 21. Flujo de datos y reportes.

En la Figura 22 se observa una vista de los tableros creados que muestra el desempeño promedio (WER) de cada modelo a nivel entrevista. Se disponen filtros que permiten aplicar cálculos para distintos modelos, métricas o situaciones que se consideran oportunas para el análisis. Por ejemplo, cómo obtener la misma métrica para ejecuciones del pipeline con y sin filtro de ruido o contemplando o no la heurística de alucinaciones.

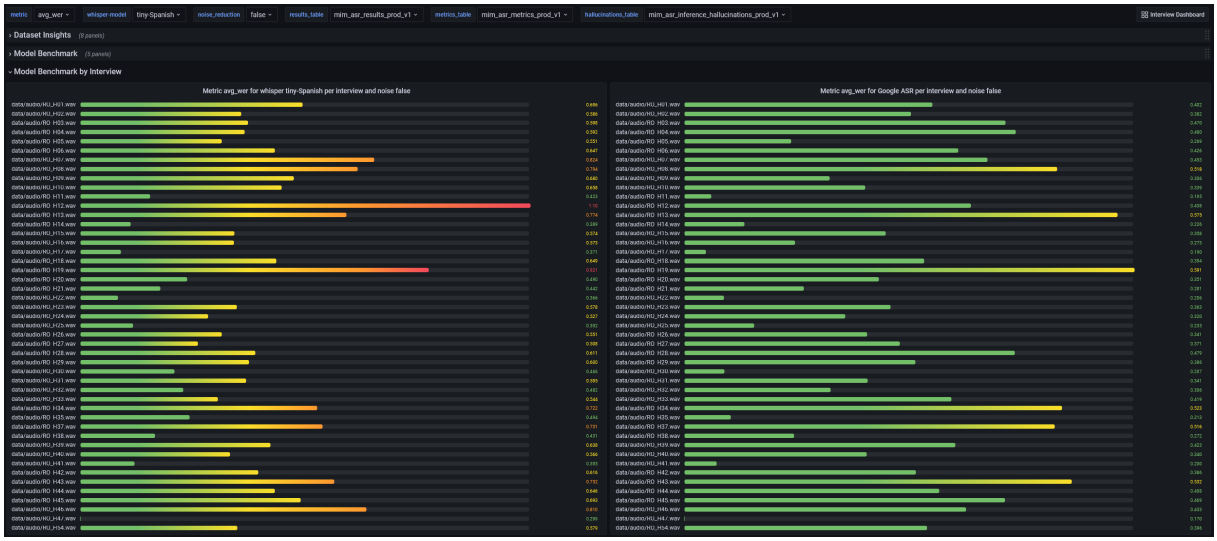

Figura 22. Dashboard para análisis de modelos de ASR.

La información que genera el *pipeline* de inferencia se modela en 3 tablas distintas que se detallan en el Anexo C.

3.2. Entrenamiento y Optimización de modelos

De los modelos elegidos para realizar este trabajo, el que permite aplicar procesos de optimización es OpenAI Whisper ya que la metodología y el modelo pre entrenado se han hecho públicos como código abierto. No obstante, para los procesos de optimización no se utilizarán los modelos publicados por OpenAI, sino las réplicas pre entrenadas disponibles en el hub de inteligencia artificial HuggingFace. Estas variantes, que fueron publicadas por

OpenAI, replican la arquitectura de red neuronal Transformer Seq2Seq Encoder-Decoder de Whisper pero en su implementación utilizan una librería que ofrece una interfaz que facilita su manipulación y entrenamiento. La librería en cuestión es Transformers (ver Anexo G) y es muy utilizada en la comunidad de modelos Open Source. A su vez, OpenAI Whisper utiliza partes de la librería Transformer dentro de su modelo, en particular el módulo generador de Tokens[13][14].

3.2.1. Entrenamiento

Antes de describir el proceso de entrenamiento del modelo, se procede a detallar cómo se realiza la inferencia haciendo foco en los distintos componentes de la arquitectura utilizada.

Para realizar la inferencia, dado que el modelo fue entrenado con archivos de audio muestreados a una tasa de 16KHz y ventanas de tiempo de 30 segundos se realizan las transformaciones acordes. En los casos en los que el archivo dure menos de 30 segundos, se rellenará el vector con ceros (padding).

Como se observa en la Figura 23, luego de procesar el audio, se extraen los Mel Frequency Cepstral Coefficients (MFCC) utilizando el módulo WhisperFeatureExtractor. Estos coeficientes son las características que utiliza el modelo como entrada por ser un buen modelo acústico de la voz humana.

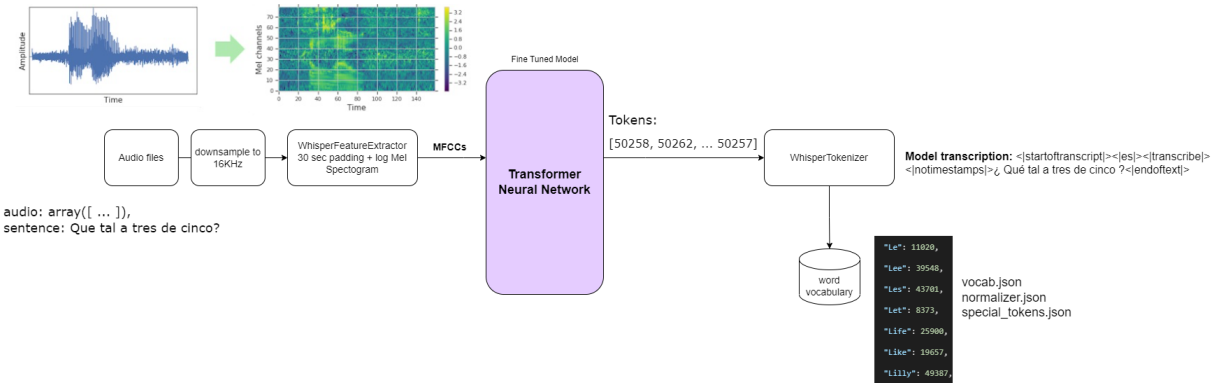

Figura 23. Diagrama que representa el proceso de inferencia de Whisper.

Luego, estos coeficientes ingresan a la red neuronal y, como producto a la salida, se genera un vector de tokens numéricos por cada una de las palabras inferidas y que tendrá la misma dimensión que el vocabulario del modelo. El vocabulario es un mapa con pares clave-valor que asocia palabras con tokens numéricos y hay muchas estrategias para construirlo. El caso particular de Whisper utiliza un generador de tipo Byte Pair Encoding (Sennrich et al., 2016) que se basa en la frecuencia de aparición de palabras/sub-palabras en los datos de entrenamiento.

A este vector se le aplicará una función Softmax para extraer las probabilidades de cada token del vocabulario para el segmento de audio procesado. De esta manera, se elige el token con mayor probabilidad y es el que la red seleccionará como salida. Este proceso se repite hasta finalizar el procesamiento del audio completo.

Por último, los tokens seleccionados se procesan con el módulo WhisperTokenizer, que busca dentro del mapa con pares clave-valor para hacer la conversión inversa de token a

[13]https://pypi.org/project/openai-whisper/
[14]https://github.com/openai/whisper/blob/v20230117/whisper/tokenizer.py

palabra. Además, el vocabulario tiene códigos especiales que se usan como meta información de la transcripción. Como por ejemplo, para indicar el inicio de una nueva oración se utiliza el código <|*startoftranscript*|>, para indicar el lenguaje <|*es*|>, el final de una transcripción <|*endoftext*|> y cada uno de estos códigos tendrá un token numérico asociado en el vocabulario. En la Tabla 9 se muestra un ejemplo del módulo WhisperTokenizer con la metainformación de transcripción.

Oración del dataset (ground truth)	El señor que trae el correo a mi casa es mi vecino
Tokens ground truth	[17356, 22188, 631, 220, 17227, 68, 806, 29731, 78, 257, 2752, 9022, 785, 2752, 42021, 2982]
Tokens inferencia	[**50258, 50262, 50359, 50363**, 17356, 22188, 631, 220, 17227, 68, 806, 29731, 78, 257, 2752, 9022, 785, 2752, 42021, 2982, **50257**]
Transcripción del modelo con códigos de transcripción	<|**startoftranscript**|> <|**es**|> <|**transcribe**|> <|**notimestamps**|> El señor que trae el correo a mi casa es mi vecino.<|**endoftext**|>
Transcripción del modelo	El señor que trae el correo a mi casa es mi vecino.
Transcripción normalizada	[['el', 'señor', 'que', 'trae', 'el', 'correo', 'a', 'mi', 'casa', 'es', 'mi', 'vecino']]
Oración del dataset normalizada	[['el', 'señor', 'que', 'trae', 'el', 'correo', 'a', 'mi', 'casa', 'es', 'mi', 'vecino']]
WER	0.0

Tabla 9. Transformaciones aplicadas a lo largo del proceso de inferencia.

Durante el proceso de entrenamiento se replica lo hecho en la inferencia, pero se comparan los tokens a la salida del modelo con la los tokens de la oración que representa el *ground truth*. Para cada lote de datos procesados (*batch*), se actualizarán los pesos del modelo usando AdamW (Loshchilov y Hutter, 2019) como mecanismo de optimización y entropía cruzada como función de pérdida al tratarse de un modelo que procesa secuencias de longitud variable tanto a la entrada como a la salida. Por otra parte, se utiliza un *scheduler linear* para la tasa de aprendizaje (*learning rate*) del optimizador con un valor inicial de 1e-5. Estos detalles se pueden ver en el Anexo D.

Por último, cada 1000 *steps* se ejecuta el proceso de evaluación y cómputo de la métrica WER sobre los datos de test así como la actualización de hiperparámetros como la tasa de aprendizaje. Un *step* representa el procesamiento de un *batch* y posterior actualización de pesos.

Cada uno de estos estados o checkpoints producto del proceso de evaluación será guardado en un repositorio de manera tal de poder utilizar cualquiera de ellos para tareas de inferencia. Además, el proceso de entrenamiento seleccionará el mejor de todos estos checkpoints según la métrica WER más baja.

Como la selección del modelo se realiza en base al mejor WER, el sobreajuste generado al utilizar una cantidad de *steps* tan alta solo afectará en términos del tiempo que requiere el entrenamiento, ya que si el WER que mejor performa en test sucedió a los 2000 *steps*, entonces ese será el checkpoint seleccionado.

3.2.2. Optimización de modelos

Dada la poca cantidad de información que se cuenta para la etapa de optimización, con respecto a la cantidad utilizada para el entrenamiento inicial y al tamaño del modelo, se espera que este se ajuste rápidamente a los datos de entrenamiento. No obstante, el objetivo es entender que tanto el proceso de entrenamiento puede mejorar o ajustarse aún con poca información a patrones del habla en la población de estudio. En la Figura 24 se observa cómo se dividió el *dataset* en conjuntos de *training*, validación y test respectivamente.

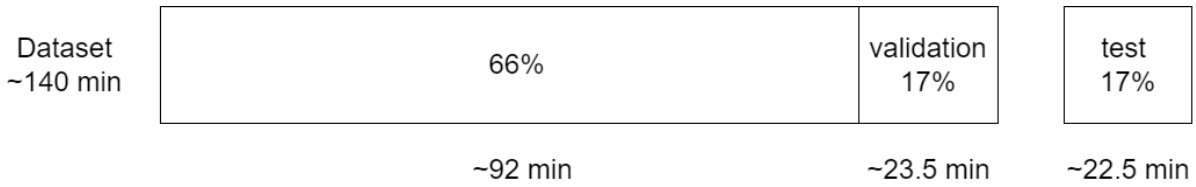

Figura 24. División de *dataset* para proceso de entrenamiento.

El conjunto de test (holdout set) es separado en otro *dataset* sobre el que se ejecutarán tareas de validación para entender cómo generaliza el modelo. Es importante aclarar que, al tratarse de fragmentos de las entrevistas, habrá audios del mismo hablante tanto en *training* como en test. Si bien no habrá *data leakage* en términos estrictos, el modelo conocerá al hablante, por lo cual los resultados obtenidos serán optimistas.

Otra pregunta que esta situación propone, si bien fuera del alcance de esta tesis, es cuánto tiempo de entrenamiento necesita un modelo para generalizar bien o entender el modelo acústico de una persona en particular que pertenezca a la población de estudio. Se espera que esta metodología pueda extenderse para ayudar a contestar esta y otras preguntas.

Además de especializar al modelo con el *dataset* generado a partir de las entrevistas, se han entrenado otras versiones realizando transferencia de conocimiento utilizando el *dataset* en español Common Voice 11[15]. El objetivo es entender si especializando aún más al modelo en la lengua objetivo se obtienen mejores resultados. Por último, sobre este modelo optimizado con el *dataset* Common Voice 11, se hace una optimización más esta vez utilizando el *dataset* generado con las entrevistas tomando como referencia las experiencias de Jeong et al., 2021.

Los detalles de los modelos entrenados se muestran en la Tabla 10.

[15]https://commonvoice.mozilla.org/en/datasets

ID	Modelo Base	Dataset	Descripción	Nombre
1	whisper-tiny	asr-interviews-trainer-full	Dataset de entrevistas	custom_tiny_no_augmented
2	whisper-tiny	asr-interviews-trainer-full	Dataset de entrevistas utilizando data augmentation	custom_tiny_augmented
3	whisper-tiny	mozilla-foundation/ common_voice_11_0	Dataset Common Voice	custom_tiny_cv11
4	whisper-tiny	mozilla-foundation/ common_voice_11_0 asr-interviews-trainer-full	Dataset de entrevistas utilizando data augmentation sobre modelo pre entrenado con Common Voice	custom_tiny_cv11_augmented
5	whisper-small	asr-interviews-trainer-full	Dataset de entrevistas	custom_small_no_augmented
6	whisper-small	asr-interviews-trainer-full	Dataset de entrevistas utilizando data augmentation	custom_small_augmented
7	whisper-small	mozilla-foundation/ common_voice_11_0	Dataset Common Voice	custom_small_cv11
8	whisper-small	mozilla-foundation/ common_voice_11_0 asr-interviews-trainer-full	Dataset de entrevistas utilizando data augmentation sobre modelo pre entrenado con Common Voice	custom_small_cv11_augmented

Tabla 10. Variantes de modelos producto del proceso de *fine tuning*

3.2.3. Data Augmentation

Otra variante que se prueba para entender el impacto en la eficacia de estos modelos sobre la población de estudio, es la implementación de *data augmentation*. Algunos modelos se entrenarán extendiendo los datos de entrenamiento realizando distintos tipos

de transformaciones sobre los audios de manera de hacerlo más robusto sobre estas distorsiones y para compensar la poca cantidad de datos disponibles. Esto se implementa con la librería *audiomentations* (ver Anexo G). Las transformaciones que se detallan en la Tabla 11 se aplican aleatoriamente y en composición sobre los archivos del *dataset*, permitiendo extenderlo de un dataset con 1204 fragmentos y casi 93 minutos de audio a otro con 19264 fragmentos y casi 25 horas de audio como se observa en la Tabla 12. La operación de *augmentation* solo se hace sobre la partición de entrenamiento.

Transformación	**Función**	**Parámetros**
Agrega ruido aditivo blanco gaussiano al audio	AddGaussianNoise	min_amplitude=0.005 max_amplitud=0.015 p=0.8
Aplica un filtro pasa bajos eliminando/atenuando frecuencias por fuera de la función transferencia	LowPassFilter	min_cutoff_freq=3000 max_cutoff_freq=3001 p=0.7
Aplica ganancia en dB al audio	Gain	min_gain_in_db=-3 max_gain_in_db=3 p=0.4
Modifica el pitch del audio en semitonos	PitchShift	min_semitones=-4 max_semitones=4 p=0.3

Tabla 11. Parámetros y características de las transformaciones realizadas para implementar data augmentation.

Atributos		**Dataset**			**Dataset c/ Augmentation**		
Divisiones		**Total**	**speaker1**	**speaker2**	**Total**	**speaker1**	**speaker2**
Fragmentos audio	Train	1024	614	590	19264	9824	9440
	Validation	301	151	150	301	151	150
Horas audio	Train	92.15min	NA	NA	24.57hrs	NA	NA
	Validation	23.53min	NA	NA	23.53min	NA	NA
Cantidad palabras	Train	10650	NA	NA	170400	NA	NA
	Validation	2630	NA	NA	2630	NA	NA

Tabla 12. Comparación de dataset con y sin *augmentation*.

Como se mencionó anteriormente, las transformaciones pueden ser aplicadas al mismo tiempo sobre un archivo con probabilidad p. La probabilidad fue definida intuitivamente dándole más peso a las transformaciones que son más representativas de características que pudieran tener las entrevistas, como el caso del ruido blanco gaussiano como se observa en la Tabla 11.

3.2.4. Monitoreo de entrenamientos

Como parte de la propuesta metodológica, se propone implementar un mecanismo que permita monitorear, comparar y dar seguimiento a los modelos optimizados de manera automática y embebida en la tarea misma de entrenar. Esto es de especial importancia

en este tipo de modelos dado su tamaño y los recursos de cómputo y tiempo necesarios para su entrenamiento. Por un lado, se almacenan todos los hiperparámetros y atributos de configuración necesarios para reproducir el proceso, así como el *dataset* utilizado, la performance del modelo a lo largo del entrenamiento, el checkpoint seleccionado como óptimo en base al Word Error Rate y el repositorio donde será guardado.

Adicionalmente, se utiliza la librería *codecarbon* (ver Anexo G) para tener valores estimados de la energía eléctrica y de las emisiones de carbono utilizadas durante el proceso de entrenamiento. Si bien estos valores pueden ser estimados utilizando distintas metodologías u otras librerías, el valor de la propuesta está orientado a habilitar el seguimiento de la huella energética y de emisiones que el cómputo que soporta este tipo de modelos genera y que esta sea una variable más a la hora de analizar el *trade-off* de sus beneficios.

El modelo de datos utilizado para almacenar la información que genera cada entrenamiento realizado se muestra en la Figura 26. La aplicación, al terminar de realizar el entrenamiento, procede a calcular las métricas y guardarlas en la base de datos correspondiente con el objetivo de permitir trazabilidad y análisis como se observa en la Figura 25.

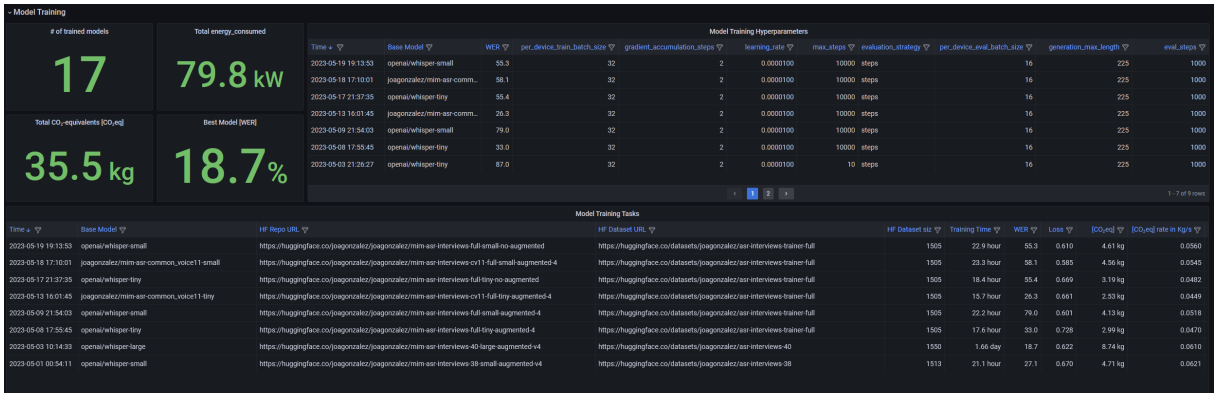

Figura 25. Visualización de métricas del proceso de entrenamiento.

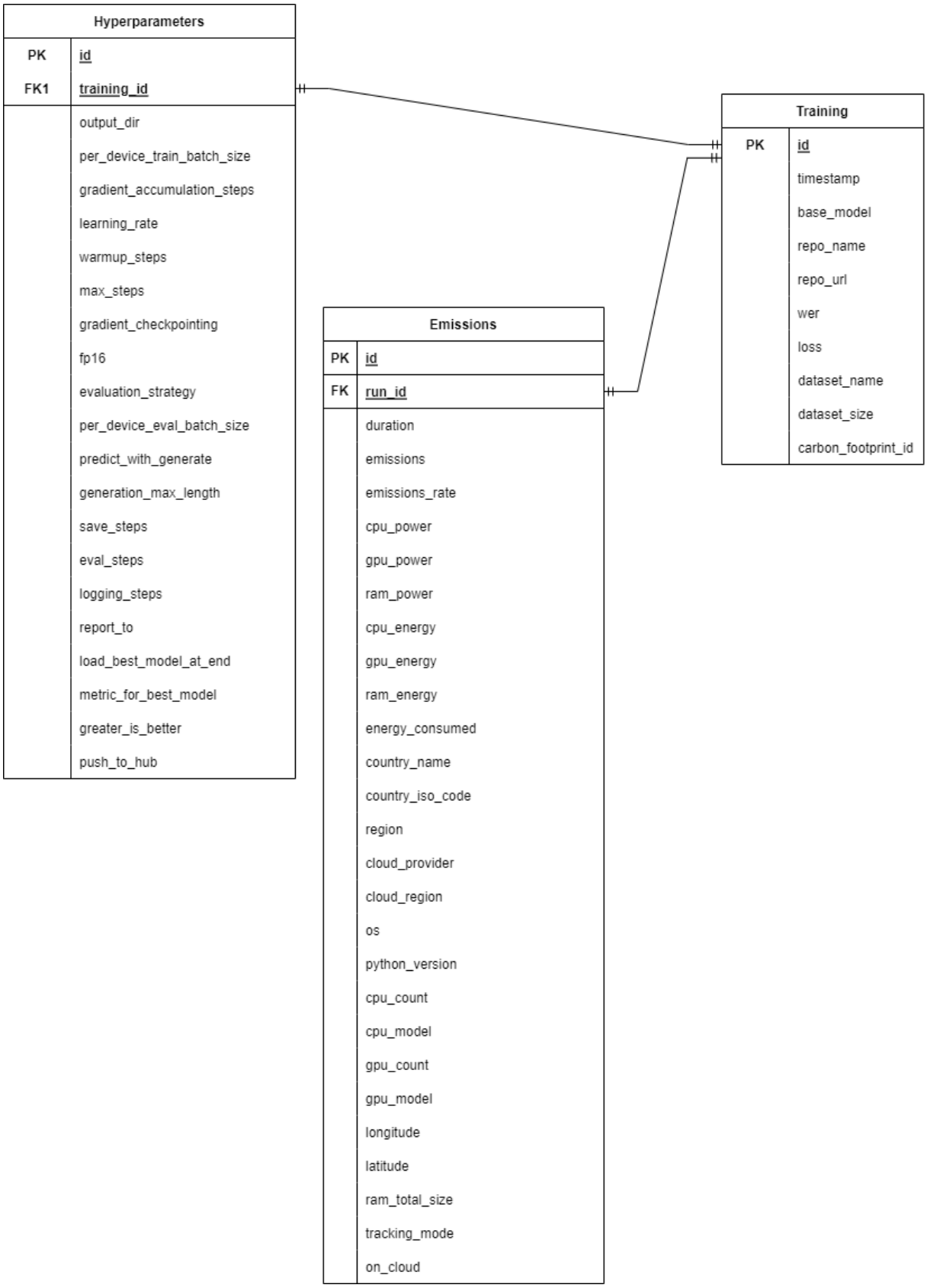

Figura 26. Esquema de tablas que modela métricas para trazabilidad del proceso de entrenamiento.

4. Resultados y Análisis

En esta sección se detallan los resultados obtenidos. En una primera instancia, en la sección 4.1, se describen los resultados de los modelos que fueron sometidos a tareas de *fine tuning*. Aquí, se detallan los resultados de performance de los modelos obtenidos durante el proceso de entrenamiento contra el *dataset* de evaluación (siempre se publica la performance del mejor checkpoint) y, además, la performance obtenida de cada modelo contra la partición de test (ver sección 3.2.2).

Luego, en la sección 4.2, se detalla el desempeño que estos modelos tuvieron al realizar el *pipeline* de transcripción y se los compara con las variantes originales. Esto permite evaluar a los modelos contra un conjunto más amplio de datos y validar el efecto que distintas tareas de preprocesamiento tienen sobre su performance.

Por último, se describen dos productos generados durante la realización de esta tesis: por un lado, un corpus de datos inéditos para objetivos académicos que permita aumentar la base de futuras tareas de transferencia de conocimiento en modelos de ASR especializados y, por el otro, un sistema de reconocimiento automático del habla con capacidad de integrarse con distintos modelos.

4.1. Entrenamiento y fine tuning

En la Tabla 13 se listan los modelos optimizados junto con su desempeño en *datasets* de evaluación y test.

Cada uno de los modelos reportados es producto de un proceso de optimización, en donde se parte de un modelo base pre entrenado, con excepción de los casos 9 y 10, en donde se muestra el desempeño de los modelos base sobre el mismo conjunto de test para poder comparar.

Se trabaja sobre cuatro variantes de las versiones *tiny* y *small* de OpenAI Whisper según se detalla en la sección de metodología 4 dando un total de ocho modelos optimizados distintos. Las variantes contemplan realizar transferencia de conocimiento de la siguiente manera:

1. Utilizar el dataset de entrevistas sin modificaciones: *custom_<modelo>_no_augmented*
2. Utilizar el dataset de entrevistas y aplicar data augmentation: *custom_<modelo>_augmented*
3. Utilizar el dataset Common Voice 11 sin modificaciones: *custom_<modelo>_cv11*
4. Utilizar modelo previamente entrenado con Common Voice 11 con dataset de entrevistas y aplicar data augmentation: *custom_<modelo>_cv11_augmented*

ID	Modelo Base	Nombre	WER Eval/Test [%]		Emisiones CO2[Kg]	Dur. [Hr]
1	whisaper-tiny	custom_tiny_no_augmented	55.35	26.50	3.19	18.4
2	whisper-tiny	custom_tiny_augmented	33.01	25.00	2.99	17.6
3	whisper-tiny	custom_tiny_cv11	30.70	45.80	2.95	16.8
4	whisper-tiny	custom_tiny_cv11_augmented	30.48	24.30	2.53	17.9
5	whisper-small	custom_small_no_augmented	55.28	21.60	4.61	22.9
6	whisper-small	custom_small_augmented	51.29	20.40	4.13	22.2
7	whisper-small	custom_small_cv11	48.35	26.50	4.02	21.6
8	whisper-small	**custom_small_cv11_augmented**	58.12	**20.00**	4.56	23.3
9	whisper-tiny	N/A	N/A	46.60	N/A	N/A
10	whisper-small	N/A	N/A	32.90	N/A	N/A

Tabla 13. Modelos OpenAI Whisper optimizados.

El proceso de entrenamiento y optimización utilizando los *datasets* mencionados se detalla en la Figura 27.

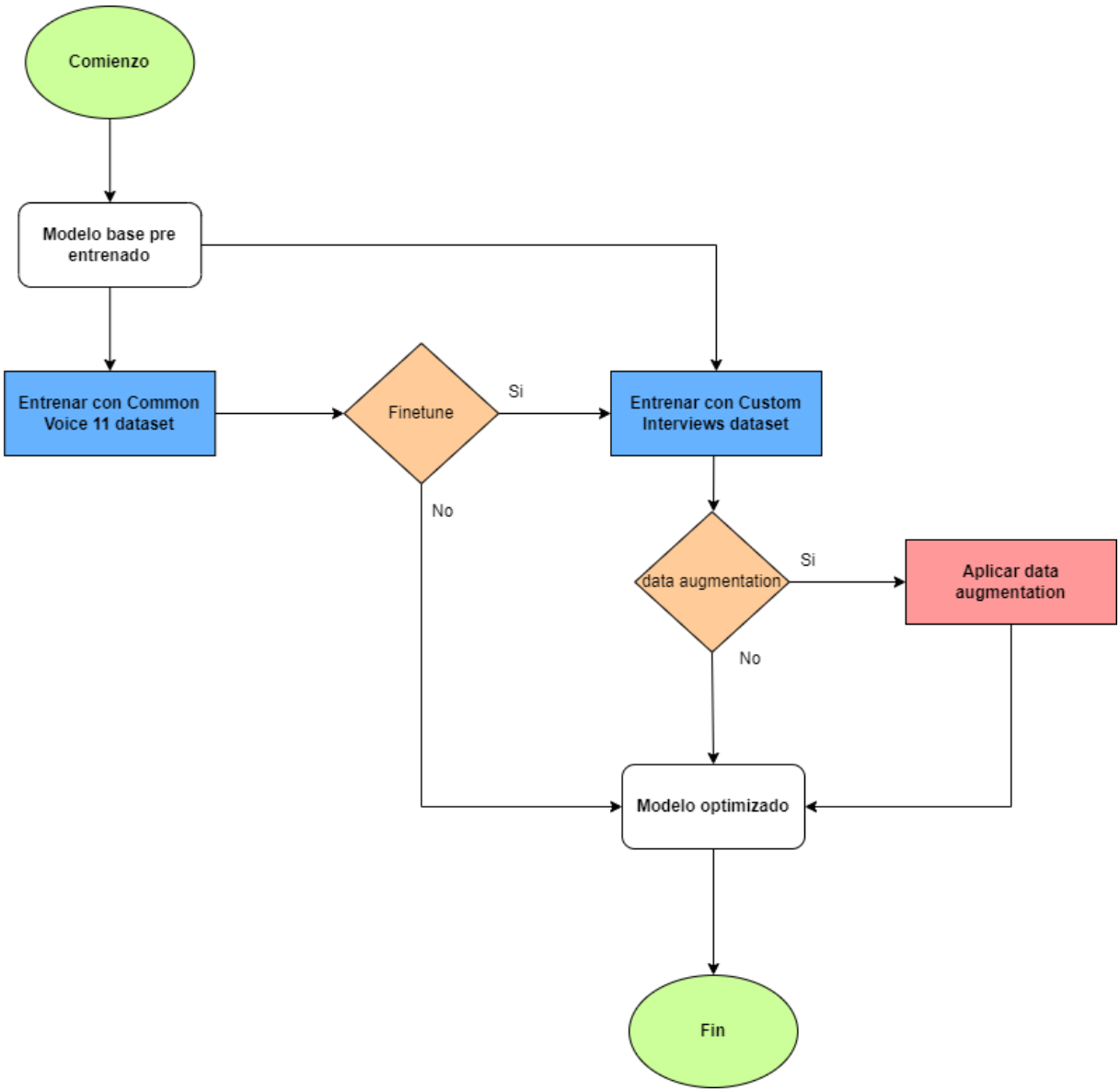

Figura 27. Flujo de entrenamiento y optimización de modelos.

Se observa que en todos los casos en donde se realiza transferencia de conocimiento con el *dataset* de entrevistas el desempeño mejora significativamente en test, siendo el mejor modelo el número 8. Este modelo alcanza un 26.6 % de mejora en el WER respecto al modelo base **openai/whisper-tiny** y un 12.9 % respecto al modelo **openai/whisper-small**.

Es importante aclarar los resultados obtenidos de Word Error Rate de los modelos optimizados para los conjuntos de evaluación y test (*held-out dataset*) respectivamente en donde se observa una peor performance en el primero de ellos. Esto se puede explicar en un principio debido a que, como se detalla en la sección de datos, tanto test como evaluación son conjuntos que tienen muy poca información (aproximadamente 20 minutos de audio). A pesar de que los segmentos fueron asignados de manera aleatoria a cada conjunto, podría haber una mayor cantidad de segmentos en test que tengan mayor claridad o que tengan, en proporción, mayor similitud con los segmentos de entrenamiento. Como hay alto sobreajuste, esto podría llevar al resultado obtenido.

En la Figura 28 se observan distintas métricas del proceso de entrenamiento del modelo 2 de la Tabla 13. En este gráfico se puede ver la evolución de la tasa de aprendizaje (*train/learning rate*), el reporte de la función de pérdida en datos de entrenamiento (*train/loss*), en datos de evaluación (*eval/loss*) y el cómputo del Word Error Rate (WER) que se realiza cada 1000 pasos (*eval/wer*).

La primera característica que se observa es la rápida tendencia al sobreajuste para todos los modelos entrenados con el *dataset* de entrevistas, en donde la función de pérdida para el conjunto de entrenamiento se adapta casi por completo a los datos luego de 3000 pasos mientras que en evaluación tiene un comportamiento creciente (*eval/loss*). Esto se podría explicar debido a que se ajusta demasiado a los patrones observados en entrenamiento y no puede generalizar correctamente.

Se muestra solamente el caso del modelo 2 de la Tabla 13 ya que el mismo patrón se repite en el resto de los modelos entrenados con este *dataset* que, como se detalló anteriormente, cuenta con aproximadamente 2hrs de audio.

Por otra parte, cuando se utilizó el *dataset* Common Voice 11, que cuenta con 413 horas[16] de audio, se generan resultados más robustos y que permiten generalizar mejor en el conjunto de evaluación. Las métricas de entrenamiento para este escenario se observan en la Figura 29, que refleja los resultados para el modelo 3 de la Tabla 13. A diferencia del caso anterior, no se observa sobreajuste en la función de pérdida en entrenamiento (*train/loss*). Finalmente, tanto resultados de WER como de la función de pérdida en evaluación (*eval/loss*) evolucionan de manera tal que se refleja un aprendizaje por parte del modelo que puede extenderse más allá del conjunto de datos de entrenamiento.

[16] https://commonvoice.mozilla.org/en/datasets

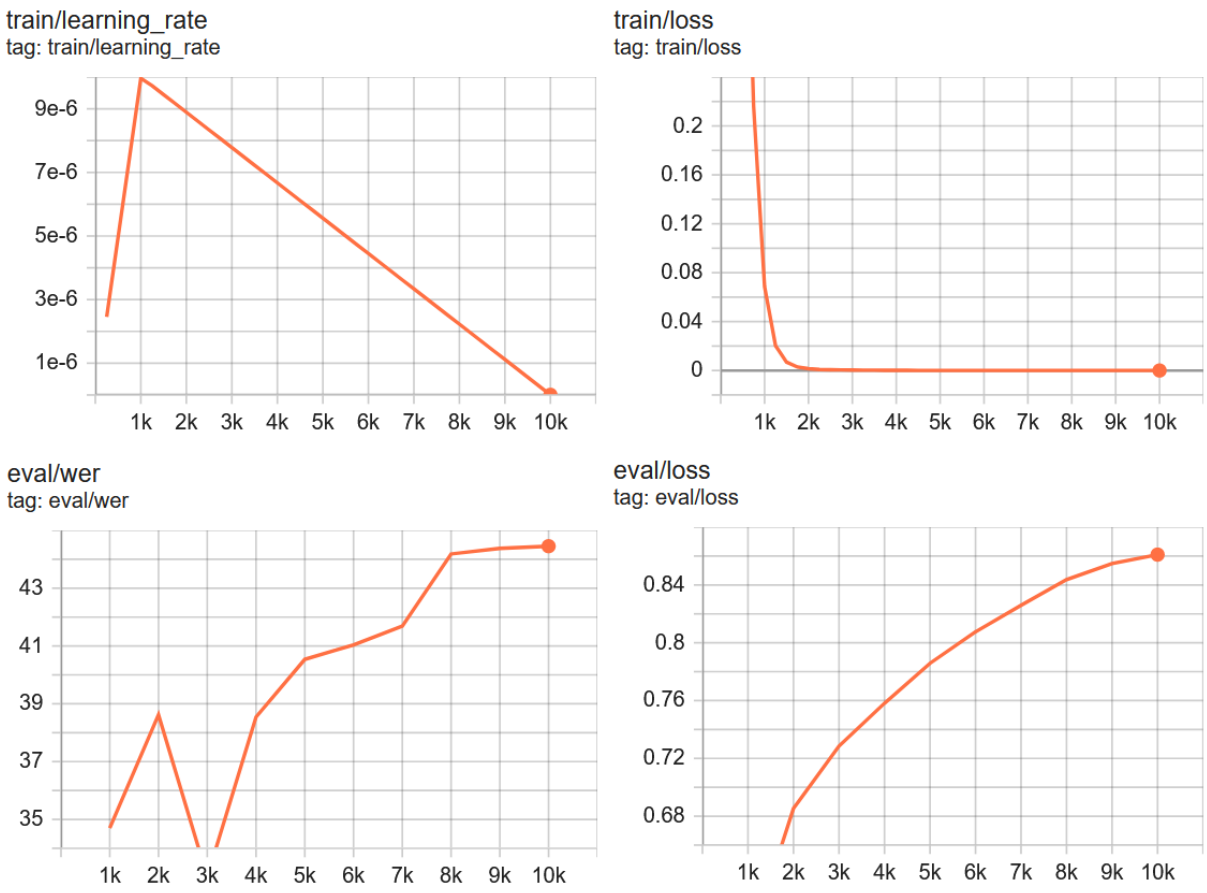

Figura 28. Métricas del proceso de entrenamiento del modelo 2 de la tabla 13 utilizando dataset de entrevistas.

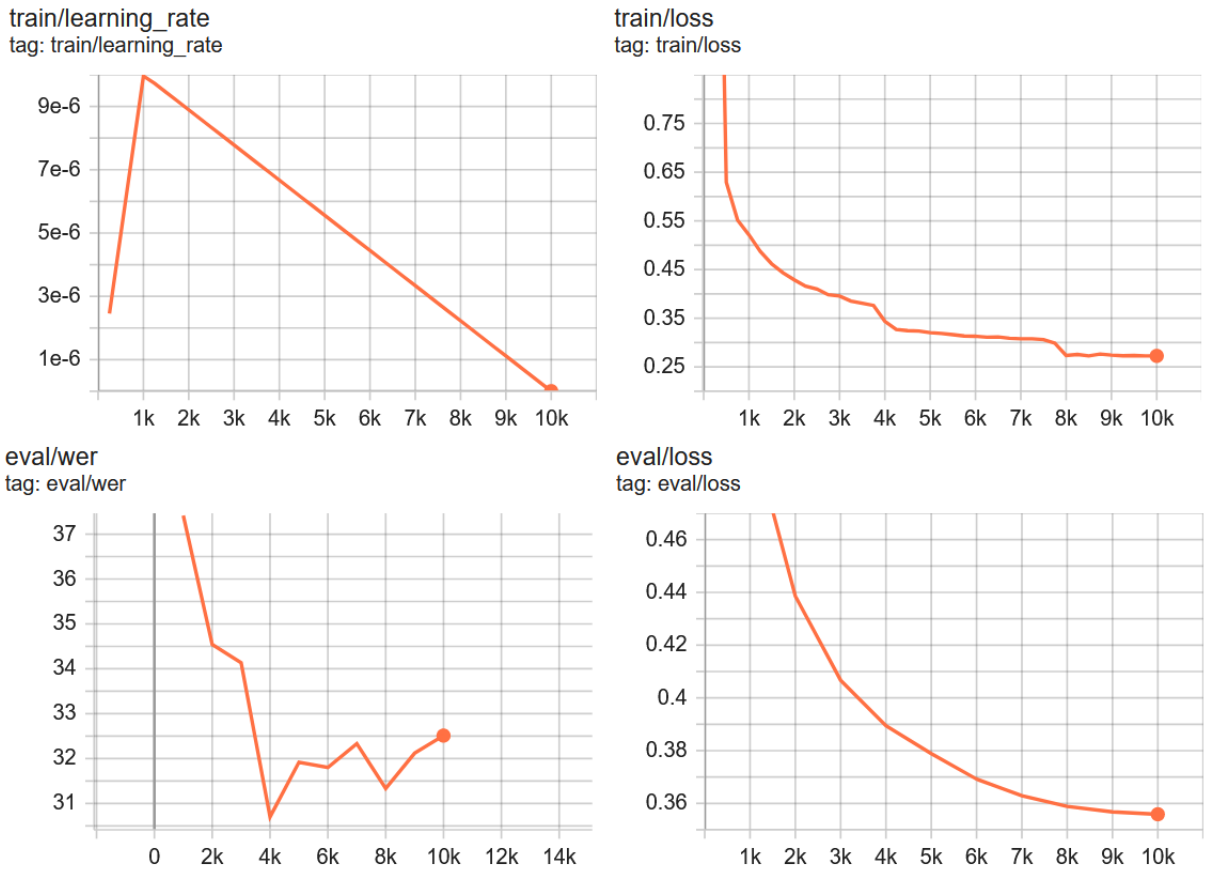

Figura 29. Métricas del proceso de optimización del modelo 3 de la Tabla 13 utilizando dataset Common Voice 11.

4.2. Evaluación y comparación de modelos

En esta sección se realiza un análisis comparativo sobre el desempeño que tienen las distintas variantes de los modelos optimizados mostrados en la Tabla 13 con respecto a sus versiones originales mostrados en la Tabla 4. Además, se considera el impacto que generan sobre ellas las tareas de pre y post procesamiento, tales como el filtro de ruido o la heurística de alucinaciones.

Con respecto a los modelos de OpenAI Whisper[17], se evaluarán tanto las versiones publicadas por OpenAI como las implementaciones de HuggingFace[18] que, si bien coinciden en arquitectura y cantidad de parámetros, utilizan distintas implementaciones. Es por este motivo que habrá un total de 10 modelos Whisper, dos implementaciones por cada una de las 5 variantes (Tabla 4).

Los valores obtenidos al realizar este análisis se obtienen de computar un total de **1735** ejecuciones del *pipeline* de transcripción para distintos modelos y tareas de procesamiento del audio.

4.2.1. Diarización

En la Figura 30 se detallan los resultados de la métrica Diarization Error Rate (DER) para el modelo de diarización. Para cada archivo de audio del corpus de entrevistas se pueden ver dos valores, uno que se corresponde con el valor producto de hacer inferencia sobre el archivo de audio original sin ningún tipo de modificación o pre procesamiento, mientras que el segundo valor es producto de computar la métrica aplicando, previo a la tarea de inferencia, un filtro de ruido al archivo. Por otra parte, en la Figura 31 se muestra la distribución de la métrica DER a través de las distintas ejecuciones del pipeline.

Además, se agregan dos marcadores en ambas figuras donde se observa el promedio de la métrica DER para los dos casos antes mencionados.

En la gran mayoría de los casos se observa que el filtro de ruido no ayuda al desempeño del modelo. Por otro lado, no se realizan tareas de *fine tuning* por lo que hay margen de mejora respecto de los resultados obtenidos. El valor de DER coincide con el reportado para el modelo utilizado en datos *out of distribution* (31.3 %) como se observa en Bredin, 2022.

4.2.2. Heurística para detección de alucinaciones

Este análisis se realiza sobre el WER de los modelos sin aplicar ningún tipo de pre procesamiento a los audios de las entrevistas con el objetivo de no superponer efectos y poder entender cómo impacta la heurística en esta métrica.

En la Figura 32 se muestra la cantidad de alucinaciones detectadas aplicando la heurística propuesta por modelo y por hablante. A priori, se observa una tendencia clara a generar una mayor cantidad de alucinaciones para los hablantes de la población de estudio. Esto podría correlacionar con la cantidad de silencios prolongados de los mismos

[17] https://github.com/openai/whisper
[18] https://huggingface.co/openai

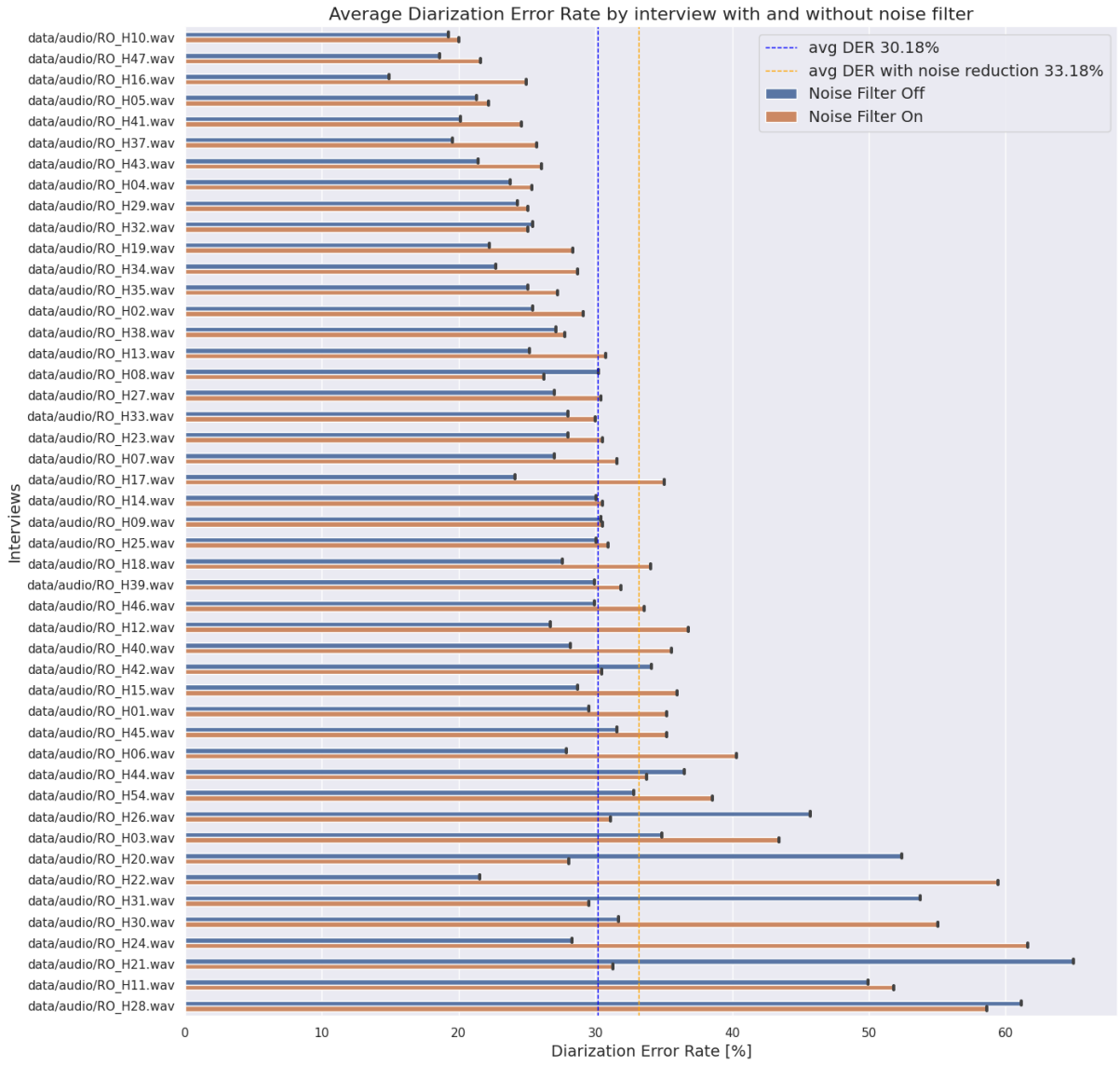

Figura 30. DER promedio por entrevista con y sin filtro de ruido.

Figura 31. Distribución de DER a través de ejecuciones del pipeline.

y el poco entendimiento del modelo a estos nuevos patrones del habla con los que no fue entrenado. Por otro lado, se observa una tendencia esperada respecto a que las variantes del modelo con menor cantidad de parámetros tienen un peor desempeño. Además, los modelos optimizados, aun siendo las variantes con menor cantidad de parámetros, presentan una tendencia a reducir drásticamente la cantidad de alucinaciones presentes teniendo una mejora respecto de los modelos más grandes como *large* y *medium* (ver Tabla 4).

Luego de una inspección manual de las alucinaciones detectadas para cada uno de los modelos no se han encontrado falsos positivos en este aspecto dando, esto último, feedback positivo respecto del funcionamiento de la heurística propuesta. En la Tabla 14 se detallan algunas alucinaciones detectadas por la aplicación.

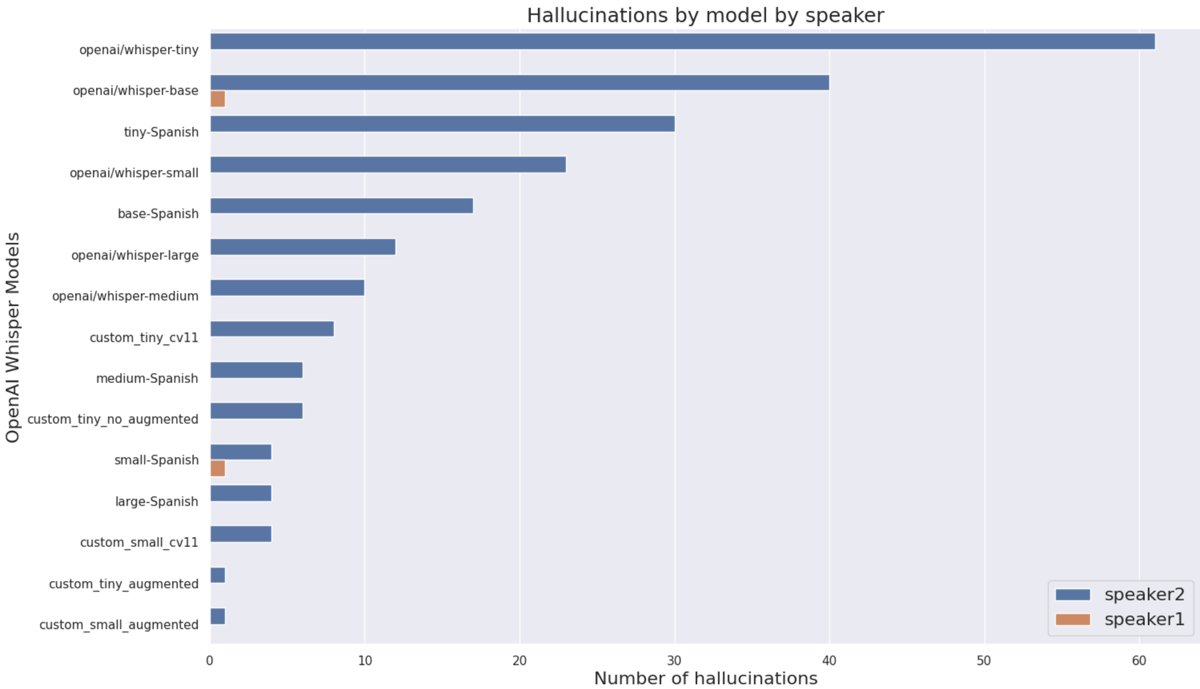

Figura 32. Cantidad de alucinaciones por modelo y hablante. speaker1 hace referencia a la entrevistadora y speaker2 al niño/a entrevistado/a.

Archivo	**Pregunta**	**Hablante**	**Inserciones**	**WER**	**Transcripción del modelo**
RO_H23.wav	q1	speaker2	438	88.6	no te me nada, no ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni ni . . .
RO_H23.wav	q12	speaker2	214	27.6	no, no, no, no, no, no, no, no, no, no, no, no, no, no, no, no, no, no, no, no, no, no, no, no, no, no, no, no, no, no, no, no, no, no, no, no, no ...
RO_H33.wav	t2	speaker2	290	49.2	Pera de tu la, un tiempo, Pera de tu la, un tiempo, Pera de tu la, un tiempo, Pera de tu la, un tiempo, Pera de tu la, un tiempo, Pera de tu la, un tiempo, Pera de tu la, un tiempo, Pera de tu ...
RO_H33.wav	q4	speaker2	324	41.4	y la manta, y la manta y la manta y la manta y la manta y la manta y la manta y la manta y la manta y la manta y la manta y la manta y la manta y la manta y la manta y la manta y la manta y la ...

Tabla 14. Algunas alucinaciones detectadas para el modelo openai/whisper-tiny y su WER

Al realizar este mismo análisis por pregunta y para todos los modelos (ver Figura 33), se observa una concentración en las primeras del test CELF 4. Se entiende que esto se correlaciona con el hecho de que muchas entrevistas no llegan a terminarse, por lo que

las preguntas finales no tienen información en todos los casos y, en consecuencia, hay una menor cantidad de alucinaciones.

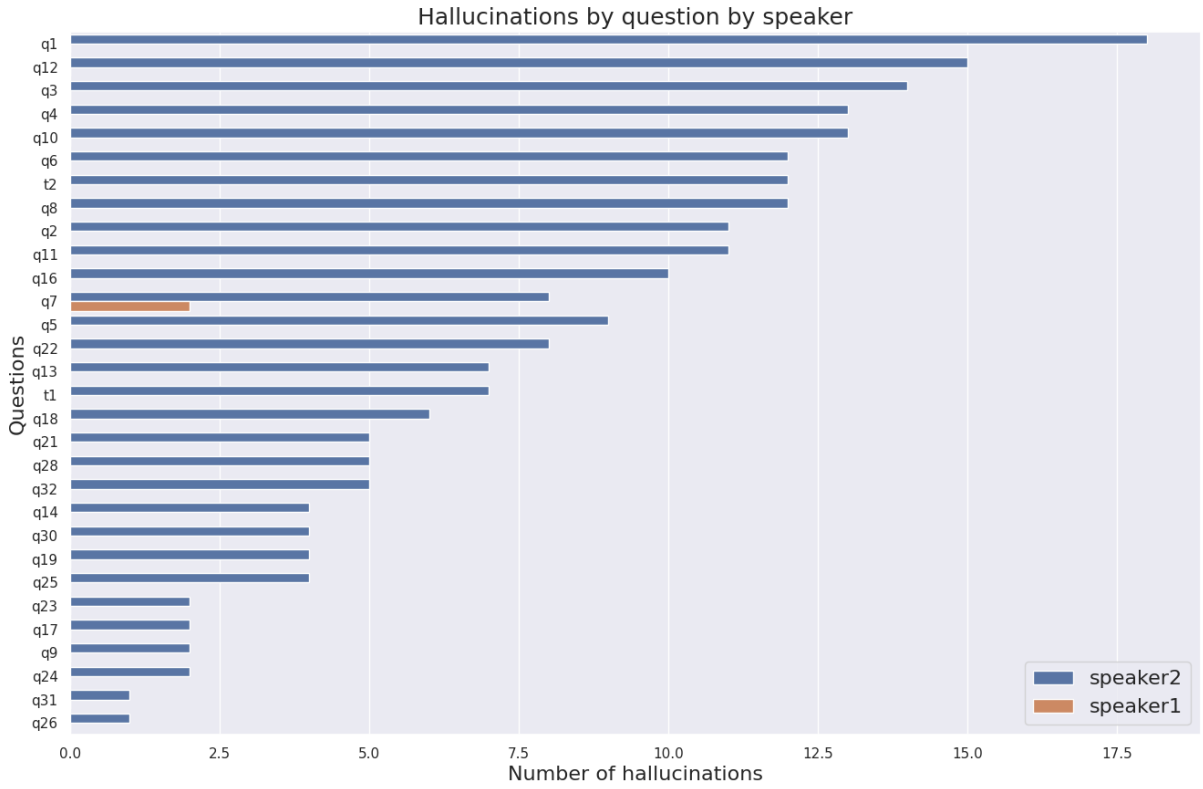

Figura 33. Cantidad de alucinaciones por pregunta y hablante.

Continuando con el análisis, en la Figura 34 se realiza una comparación del WER para las 8 variantes de Whisper optimizadas (ver Tabla 13) y para las 10 variantes originales de Whisper, cinco publicadas por OpenAI y cinco por HuggingFace. En esta comparación, por cada uno de los modelos se observan dos valores; uno que se corresponde al WER utilizando la heurística de alucinaciones y otro donde no se la utiliza.

Para los modelos optimizados, donde prácticamente no hay alucinaciones, la diferencia entre aplicar o no la heurística es nula, mientras que para los modelos originales se puede ver una diferencia de hasta un 32 % (diferencia de promedios Figura 35).

Es claro que si bien esto no soluciona el problema, permite entender en qué proporción se presenta y si hay alguna diferencia con la población de estudio. Además, se evalúan los modelos de una manera más consistente dado que estas alucinaciones, si bien son un problema y deben computarse como un error, al generar *outliers* tan grandes mueven desproporcionadamente la métrica WER.

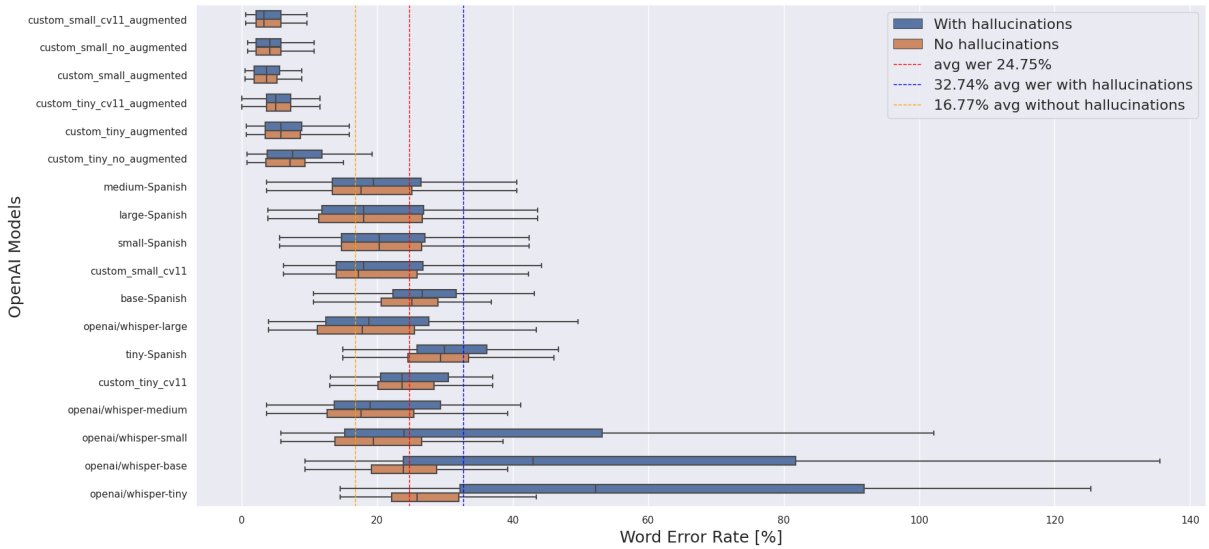

Figura 34. Comparación de WER por modelo con y sin heurística de alucinaciones.

Esto último puede observarse en la Figura 35, donde se muestra por cada modelo el corrimiento (*shift*) que se genera entre utilizar y no utilizar la heurística. Por otro lado, se marcan algunos *outliers*, donde se observan diferencias en el WER de hasta un 2771 % en algunos casos. Estos *outliers* tan grandes tienen sentido entendiendo la naturaleza de la métrica WER, que es sensible a la cantidad de palabras insertadas.

Dados los resultados obtenidos, la heurística será utilizada para remover *outliers* y realizar una evaluación más consistente de los modelos.

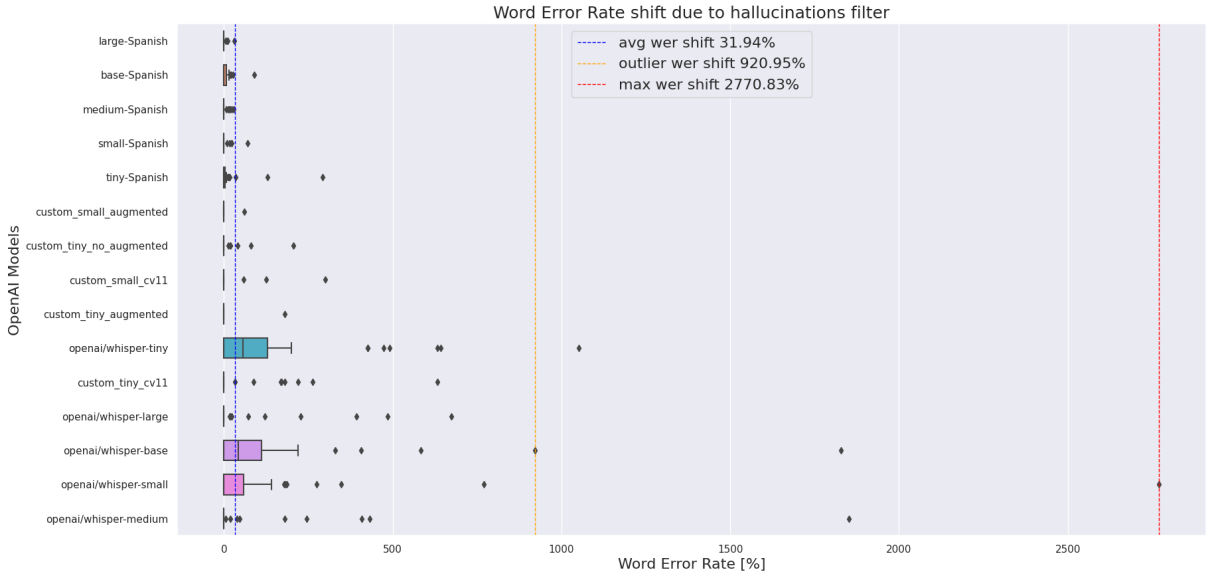

Figura 35. Corrimiento de WER que generan outliers por alucinaciones.

4.2.3. Filtro de ruido

El objetivo de esta sección es entender si existe una mejora en las tareas de ASR al utilizar filtros que atenuen el ruido de las entrevistas. Dados los resultados obtenidos y detallados en la sección anterior, este análisis se realiza sobre el WER de los modelos utilizando la heurística para detectar y remover alucinaciones.

En la Figura 36 , nuevamente se compara el desempeño de las 8 variantes optimizadas y las 10 variantes de Whisper originales. Por cada uno de los modelos listados en el gráfico, habrá 4 valores asociados. El primer par de valores, muestra el WER para el speaker1 (entrevistadora) con y sin aplicación de filtro de ruido mientras que el segundo par de valores, muestra el WER para el *speaker2* (entrevistado/a). Esta disposición permite hacer una comparación por modelo y por hablante ante la aplicación de la tarea propuesta. Además, se agregan líneas verticales que coinciden con los promedios de WER para cada uno de estos casos antes mencionados.

Por último, se observa que el speaker1 (entrevistadora) empeora un 2.92 %, mientras que el *speaker2* (entrevistado/a) lo hace un 8 % en promedio.

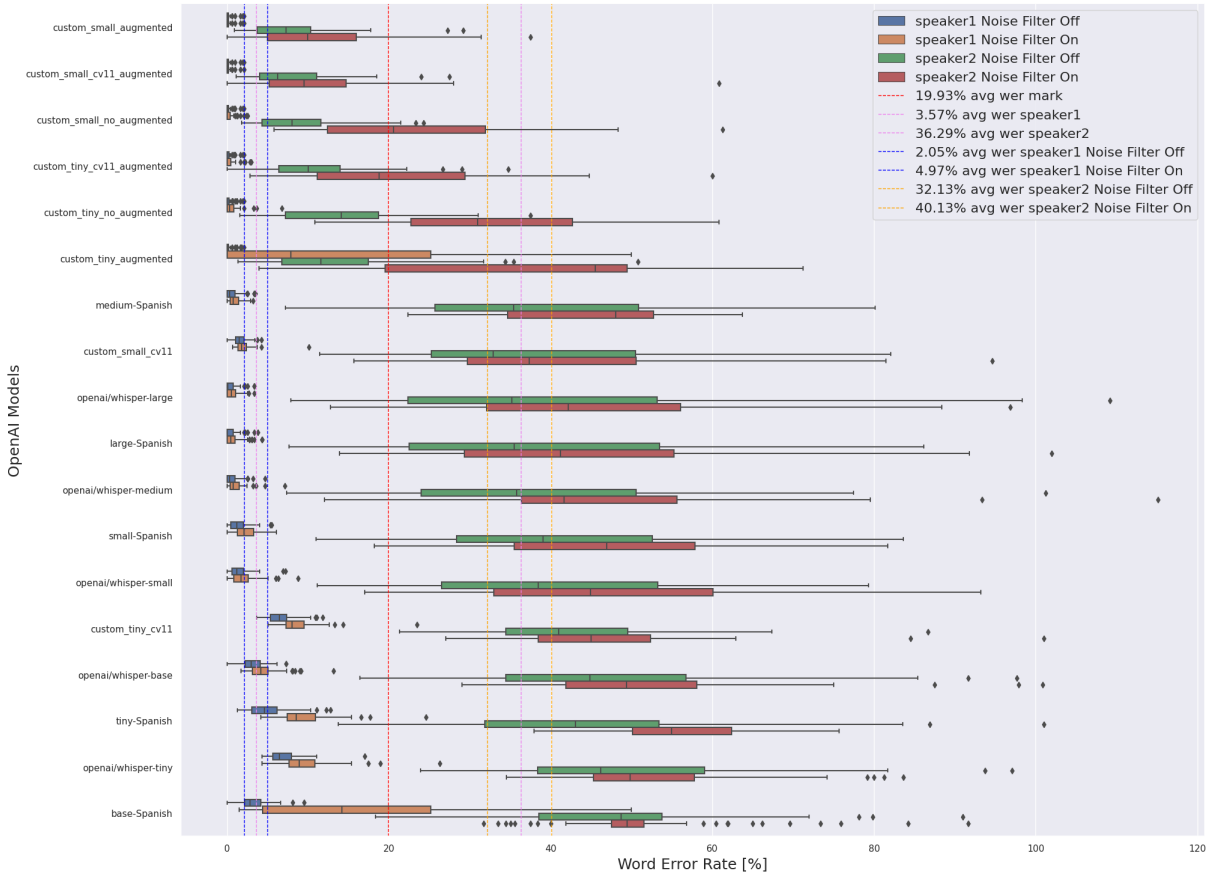

Figura 36. Efecto filtro de ruido en tarea de ASR y comparación entre modelos.

Si bien los primeros resultados obtenidos al aplicar tareas de pre procesamiento no mejoran el desempeño de los modelos, hay un gran espectro de opciones en términos de transformaciones o manipulaciones del audio que sí podrían tener un impacto positivo y que no han sido exploradas en este trabajo dado que el foco se encontraba en la metodología de evaluación.

4.2.4. Google Speech-to-Text

En esta sección se analiza el desempeño del servicio de Google Speech-to-Text al realizar inferencia sobre el conjunto de 47 entrevistas que componen el *dataset* como puede observarse en la Figura 37. Este modelo obtuvo, en su mejor variante, un WER promedio de 3.17 % para el *speaker1* y un 33.81 % para el *speaker2*. En el caso del *speaker1* (entrevistadora), se obtienen resultados de estado del arte para idioma español según lo reportado en Radford et al., 2022.

Con respecto a la aplicación de filtros de ruido, se observa el mismo comportamiento que en el caso de OpenAI Whisper.

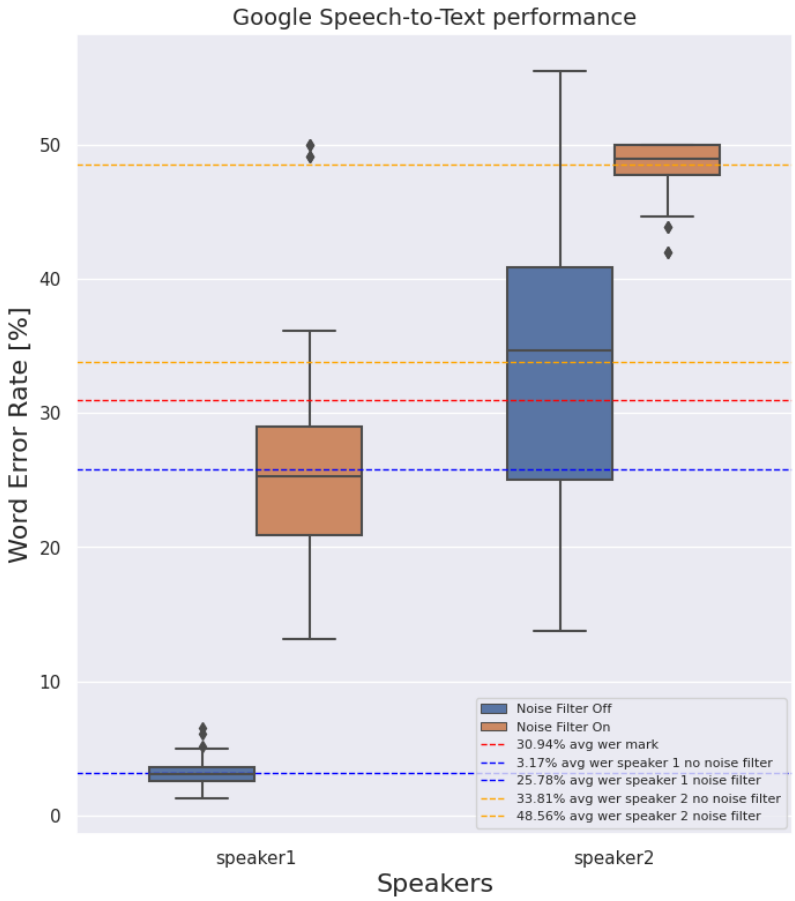

Figura 37. Resultados por hablante para servicio de Google Speech-to-Text.

WER [%] Mejor combinación Google Speech-to-Text		
speaker1	**speaker2**	**Promedio**
3.17	33.81	18.49

Tabla 15. WER para modelo Google Speech-to-Text.

4.2.5. Transferencia de conocimiento

En este apartado se comparan los resultados obtenidos entre los modelos optimizados y sus versiones originales con el objetivo de entender el impacto de la tarea de *fine tuning*.

En la Figura 38 se comparan las distintas versiones de OpenAI Whisper *tiny* que fueron optimizadas utilizando el *dataset* de entrevistas elaborado en el contexto de este trabajo y el *dataset* Common Voice 11. Además, estos resultados son comparados con la performance del modelo Google Speech-to-Text con el objetivo de entender la diferencia entre distintos productos. Por cada modelo, habrá cuatro valores de WER asociados; el primer par muestra la métrica para el *speaker1* (entrevistadora) con y sin filtro de ruido, y el segundo par de valores muestra lo mismo pero para el *speaker2* (niño/a). Además, se dan los valores de WER promedio por *speaker* con y sin filtro de ruido. Este gráfico, permite hacer foco en la comparación de la performance de los modelos *tiny* (menor cantidad de parámetros) y entender que variantes durante el proceso de *fine tuning* generan mejores resultados.

Se puede observar que con las pocas horas de audio utilizadas para realizar transferencia de conocimiento a los modelos hay un cambio drástico en el WER. Todos los modelos que fueron optimizados con el *dataset* de entrevistas obtienen mejores resultados que los modelos homónimos sin optimización, incluso más que la versión *large*. Sin embargo, se insiste en que se espera que estos modelos no generalicen bien dada la cantidad de información utilizada para el *fine tuning*. Además, a diferencia del WER obtenido al evaluar el modelo en el conjunto de test (*held-out dataset*), en este caso se realiza inferencia sobre el corpus completo de entrevistas. Es en este sentido que el proceso tiene *data leakage*, ya que se está realizando inferencia sobre las mismas entrevistas que fueron utilizadas para su entrenamiento. No obstante, la información valiosa que se extrae de estos experimentos está relacionada a validar el hecho de poder optimizar modelos pre entrenados con pocas horas de audio que sean capaces de aprender patrones del habla sobre la población de estudio dada su alta variabilidad en el modelo acústico, su poca representatividad en *datasets* y, además, validar la metodología de análisis y evaluación propuesta.
Además, se ve nuevamente una gran diferencia en el desempeño de los modelos en términos de WER entre la entrevistadora y los niños/as. El detalle del promedio por modelo y por hablante puede verse en el Anexo E.

Se esperan buenos resultados de contar con *datasets* más grandes para realizar el proceso de optimización en base a los experimentos realizados.

Además, en la Figura 39 se muestra el WER promedio para cada uno de los modelos con y sin filtro de ruido considerando a ambos hablantes.

Se realiza exactamente la misma comparación en las Figuras 40 y 41 para las variantes optimizadas del modelo small.

En las Figuras 39 y 41 respectivamente, puede verse que la combinación de transferencia de conocimiento utilizando primero un *dataset* en español como Common Voice 11 y luego otra etapa con el *dataset* de entrevistas no genera mejoras significativas en el WER ni en la varianza de los resultados comparando con la versión del modelo que utiliza *data augmentation* con el *dataset* de entrevistas solamente.

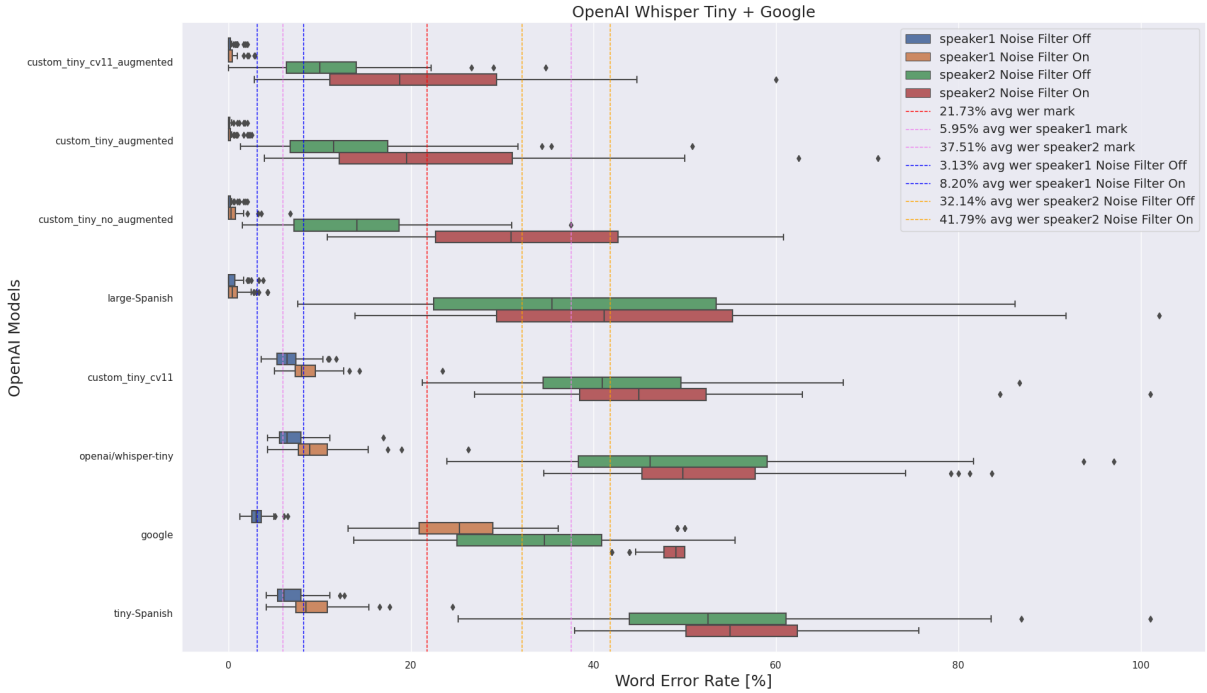

Figura 38. Comparación de modelos tiny optimizados con modelos originales.

Por último, tanto para los modelos *tiny* cómo *small* optimizados, la aplicación de *data augmentation* es la que genera diferencias más considerables. En la Tabla 16 se observa la performance promedio para los modelos con *fine tuning* y la comparación con sus modelos homónimos y la versión más grande de Whisper (la que se espera tenga mejor desempeño).

Modelo	**WER** [%]
custom_small_augmented	**4.244**
custom_small_cv11_augmented	4.263
custom_small_no_augmented	4.538
custom_tiny_cv11_augmented	**5.683**
custom_tiny_augmented	6.756
custom_tiny_no_augmented	7.140
google	**18.490**
custom_small_cv11	19.603
large-Spanish	**19.869**
small-Spanish	**21.21**
tiny-Spanish	**30.04**

Tabla 16. WER promedio considerando ambos hablantes

Las tablas con el detalle del desempeño de los modelos se encuentran en el Anexo E.

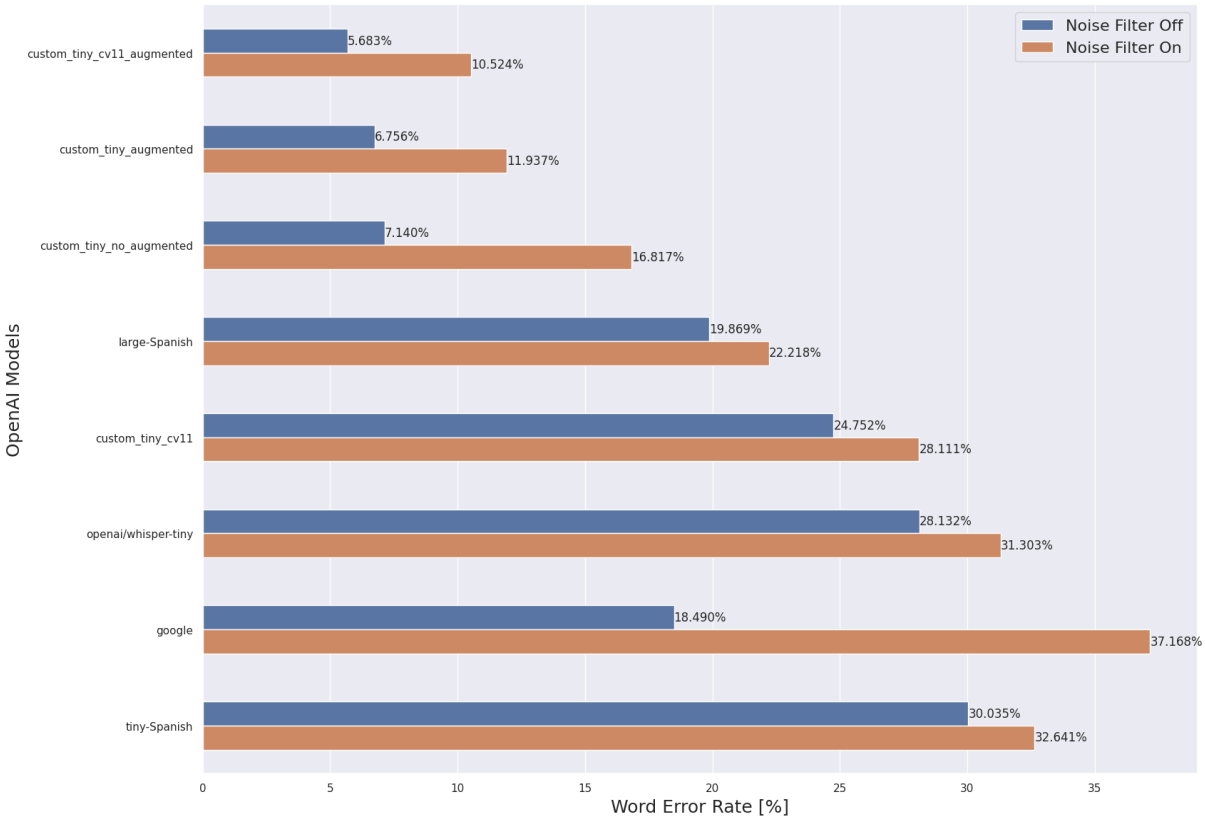

Figura 39. WER promedio de modelos tiny.

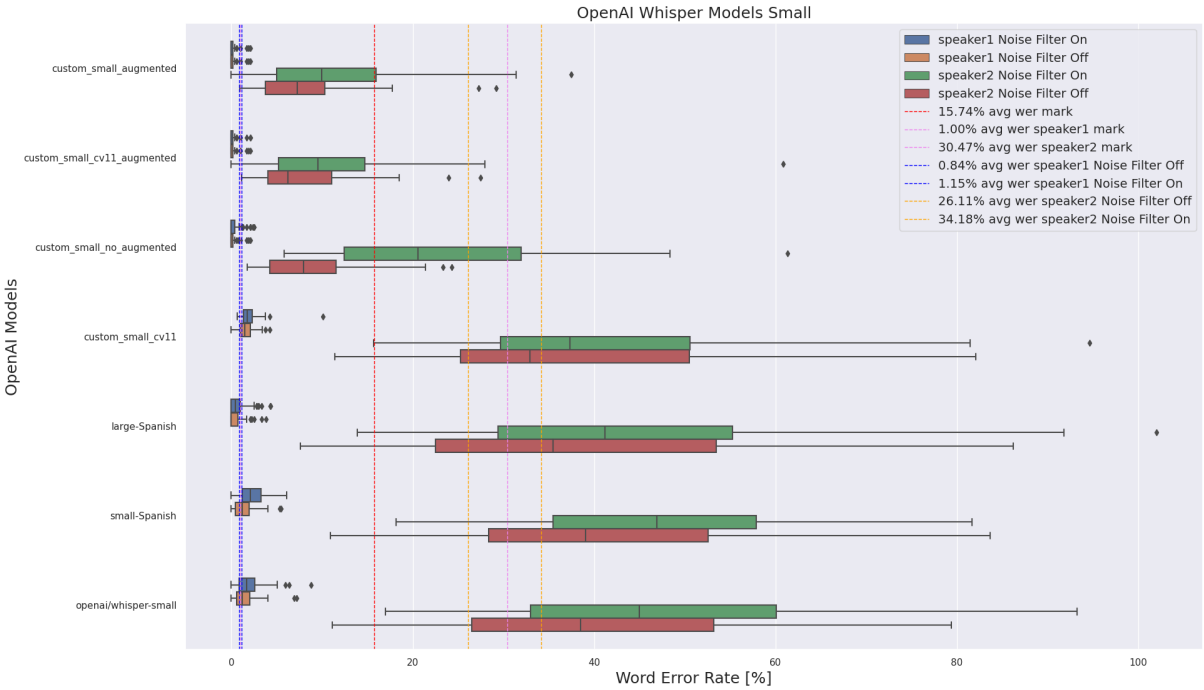

Figura 40. Comparación de modelos small optimizados con modelos originales.

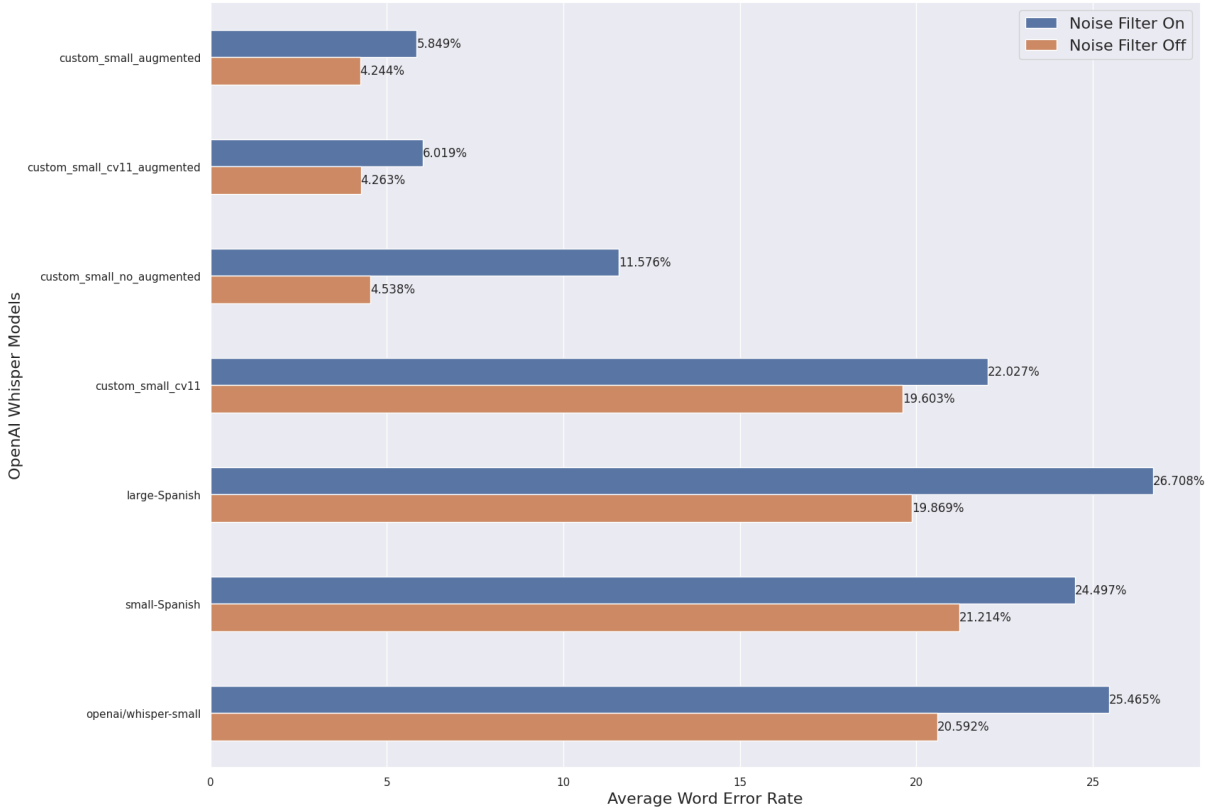

Figura 41. WER promedio de modelos small.

4.3. Corpus para tareas de ASR

Las entrevistas disponibles junto con las transcripciones realizadas en esta tesis conforman un corpus de información perteneciente a un dominio específico como es el de hablantes con patologías en el habla, que puede ser de utilidad para futuras tareas de optimización y transferencia de conocimiento en otros modelos que permitan ofrecer mejoras en distintos campos para esta población minoritaria. Por este motivo, se construye un *dataset* listo para ser distribuido y utilizado junto con un módulo que automatiza su construcción ante la eventual disponibilidad de más entrevistas o audios.

4.4. Aplicación

El código de la aplicación[19] desarrollada en formato de Command-Line Interface (CLI) que implementa la metodología propuesta para generar transcripciones enriquecidas, realizar evaluación y *fine tuning* de modelos de ASR queda disponible para ser mejorada o extendida. En la Figura 42 se muestra la aplicación funcionando con sus respectivos módulos mientras que en la Tabla 17 se describen los módulos principales.

[19]https://github.com/joagonzalez/ditella-mim-asr-tesis

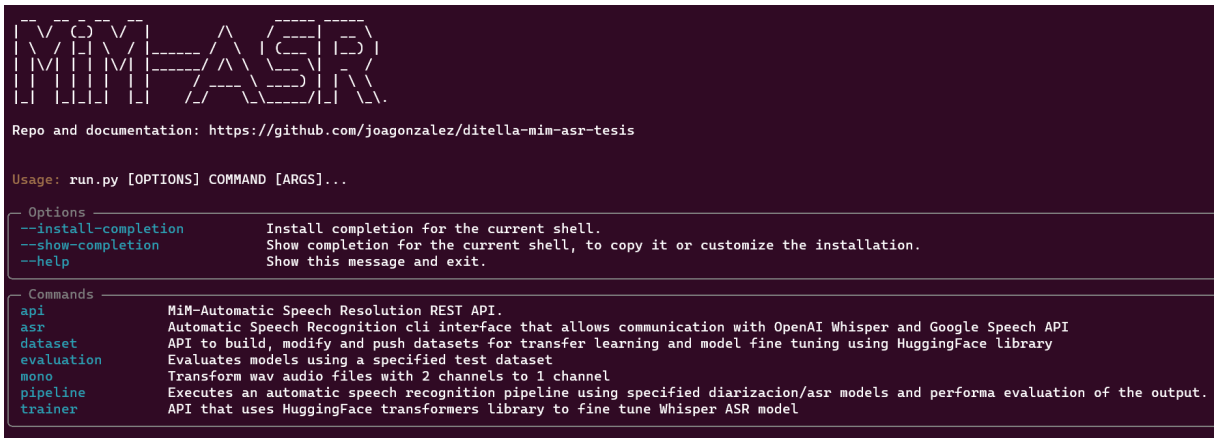

Figura 42. Módulos de aplicación CLI.

Módulo	**Descripción**
pipeline	Ejecuta el pipeline principal de la aplicación. Permite seleccionar modelo para la tarea de ASR y parametrizar el cálculo de métricas como WER o aplicar transformaciónes previas a los audios como filtros de ruido.
Trainer	Pipeline de *fine tuning* de modelo pre entrenado. Se pasa como parámetro el dataset a utilizar, el repositorio donde se guardará el modelo y sus pesos, el modelo base que se optimizará y si se desea utilizar *data augmentation*
Dataset	Pipeline de construcción de *dataset* utilizando archivos de audio en *data/audio* tomando como *ground truth* los archivos de transcripción Praat en *data/diarization*.

Tabla 17. Uso de distintos módulos de aplicación.

En la Figura 43 se muestra cómo ejecutar y parametrizar distintas tareas que permite la aplicación. En una primera instancia, se llama al módulo que implementa una funcionalidad especifica como puede ser entrenamiento o pipeline de transcripción y luego se especifican los parámetros requeridos.

```
# pipeline de transcipción y evaluación sobre el conjunto total de
↪ entrevistas (parámetro batch)
python run.py pipeline --batch --cut-files --asr --ground-truth
↪ --error-rate --comments "batch-gpu-ditella-A100-80" --asr-model custom
↪ --custom-model 'openai/whisper-tiny'

# fine tuning de modelo usando dataset de entrevistas con modelo base tiny
↪ y augmentation
python run.py trainer --dataset 'joagonzalez/asr-interviews-trainer-full'
↪ --hf-repo-name
↪ "joagonzalez/mim-asr-interviews-cv11-full-tiny-augmented-4"
↪ --whisper-model joagonzalez/mim-asr-common_voice11-tiny --custom
↪ --augmented --augmentation-level 4

# fine tuning usando modelo common_voice_11 de modelo small
python run.py trainer --dataset 'mozilla-foundation/common_voice_11_0'
↪ --hf-repo-name "joagonzalez/mim-asr-common_voice11-small"
↪ --whisper-model openai/whisper-small --no-custom --cache-dir
↪ /mnt/pure/datasets --language es

# construcción de dataset utilizando entrevistas y archivos de diarización
↪ disponibles
python run.py dataset --name asr-interviews
```

Figura 43. Ejemplos de uso de distintos módulos de la aplicación detallados en la Tabla 17.

Como se detalla en la sección de metodología, los distintos resultados de las tareas establecidas en el sistema de reconocimiento automático del habla generan métricas que pueden ser consumidas a través de distintos tableros como se observa en las Figuras 44 y 45. Estos tableros pueden extenderse o adaptarse para ser utilizados en trabajos futuros.

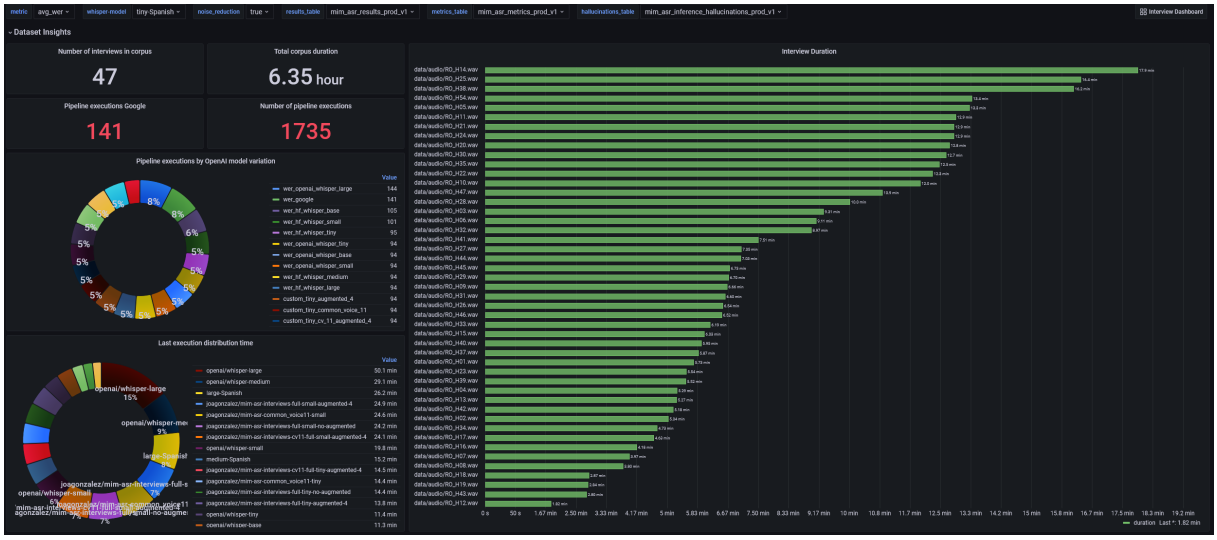

Figura 44. Tablero con información de ejecuciones del pipeline de reconocimiento automático del habla.

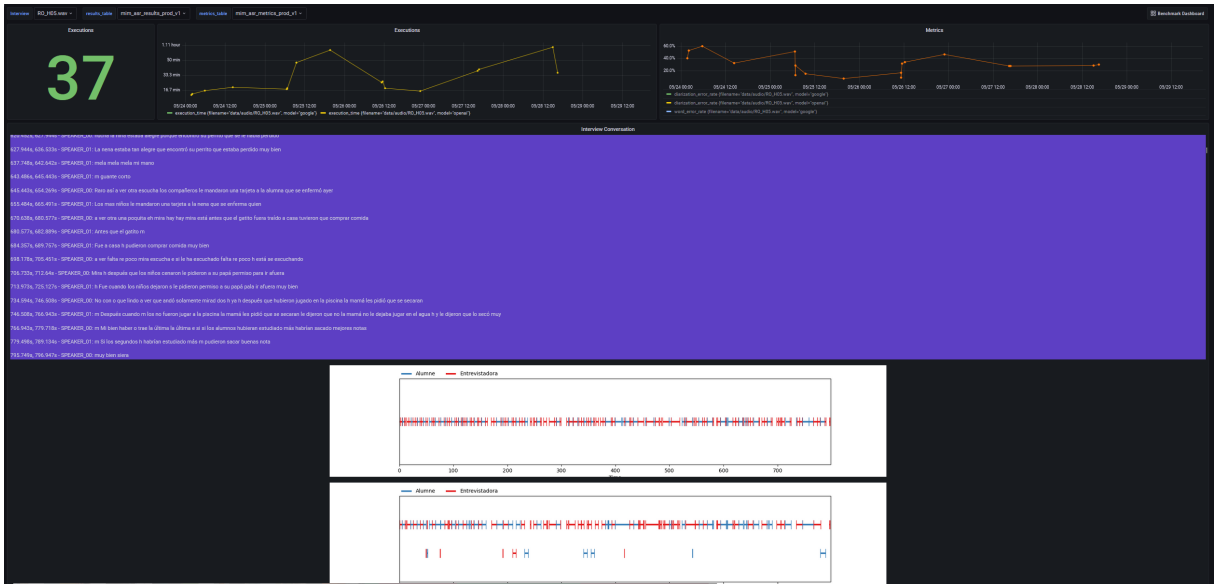

Figura 45. Tablero con métricas por entrevista y transcripción enriquecida.

5. Conclusiones

La metodología propuesta para desarrollar un sistema de evaluación y procesamiento automático del habla fue validado exitosamente a través de su implementación desarrollando una aplicación de software. Utilizando la aplicación desarrollada en el marco de esta tesis se pudieron generar, almacenar y consumir las métricas necesarias para el análisis propuesto. Tanto la metodología como la aplicación podrían ser utilizadas y extendidas en otros trabajos de investigación que requieran este tipo de funcionalidades soportando características como:

- Fácil integración de modelos de ASR al pipeline de evaluación
- Transfer learning de modelos pre entrenados y aplicación de data augmentation
- Preprocesamiento de audios (reducción de ruido, otros)
- Integración con formato de transcripciones del software Praat
- Persistencia de resultados en base de datos relacional
- Módulo para construcción de dataset en base a archivos Praat

Entre las conclusiones principales de esta tesis, puede destacarse el hecho de que los modelos pre entrenados de ASR seleccionados muestran una gran diferencia en su desempeño para poblaciones minoritarias poco representadas generalmente en datos de entrenamiento.

Por su parte, se puede afirmar que el proceso de optimización de los modelos muestra un alto grado de adaptación a los patrones del habla sobre la población de estudio, arrojando mejoras en el Word Error Rate y reduciendo de manera significativa las alucinaciones sin impactar a la métrica sobre hablantes que no presentan patologías en el habla (ver Tabla 13 y Apéndice E).

Los modelos que fueron expuestos a la técnica de *data augmentation* tuvieron mejores resultados que aquellos que no para las dos variantes de OpenAI Whisper utilizadas.

No obstante, estas observaciones deben ser tomadas dentro del contexto de este trabajo, en donde los datos utilizados para entrenamiento y validación no son suficientes para generalizar con datos *out of distribution* y donde se espera alto sesgo y alta varianza en esas situaciones respecto de las métricas obtenidas.

Por otro lado, con respecto a los cálculos resultado de procesar los datos de inferencia para los modelos optimizados, se resalta el hecho de que las métricas serán optimistas dado que la inferencia se realiza sobre información a la que los modelos tuvieron acceso en etapas de entrenamiento. No obstante, la información valiosa que se extrae de la realización de estos experimentos está relacionada a validar el hecho de poder optimizar modelos pre entrenados con pocas horas de audio (aproximadamente 90 minutos) que sean capaces de aprender patrones del habla sobre la población de estudio dada su poca representatividad y, también, validar la metodología de análisis y evaluación propuesta.

Dado que el sistema desarrollado utiliza, en una primera instancia, un modelo de diarización al cual no se le ha aplicado ningún proceso de optimización, se espera que la calidad de los resultados mejore al explorar este camino según los resultados expuestos en Bredin, 2022.

Análisis Prescriptivo

Con poco trabajo adicional, el equipo de la Dra. Carolina Gattei podría comenzar a utilizar la aplicación elaborada en el marco de esta tesis para acelerar los tiempos de transcripción y puntuación de la tarea del test CELF, que actualmente se realizan de forma manual.

Además, podrían utilizarse las métricas elaboradas por el pipeline de transcripción (o integrarse otras que el equipo considere) como un mecanismo para interpretar la inteligibilidad de los/as niños/as y otras variables individuales (edad de equipamiento con el implante coclear, tipo de implante, etiología de la sordera del niño/a, por ejemplo). Retomando lo dicho en la descripción de los objetivos, los modelos entrenados serán integrados al proyecto **DECILE** con el fin de realizar pruebas para automatizar la realización de distintos tests que aporten al diagnóstico y tratamientos en niños y niñas con patologías en el habla y la audición. Esto se realizará desarrollando una interfaz API REST que permita a los componentes actualmente desarrollados en el contexto de ese proyecto hacer uso de los modelos optimizados. Estas tareas se encuentran actualmente en desarrollo y permitirá extender el campo de aplicación de esta tesis.

Por último, habiendo validado la capacidad de este marco de trabajo para aprender modelos acústicos con alta variabilidad y de contar con una mayor cantidad de datos, podrían entrenarse modelos más robustos y con una mejor performance sobre la población de estudio. En este caso, podría evaluarse utilizando una métrica de distancia entre lo que dicen los/as niños/as y lo que dice la entrevistadora para conocer las características generales del habla de los chicos y elaborar hipótesis acerca de cuáles son los aspectos fonéticos que les generan mayor dificultad en la articulación y producción del habla. En este caso, la métrica se utilizaría para diagnosticar a la población de estudio, en vez de al modelo. Esta información podría utilizarse para comprender si existen relaciones entre el nivel de inteligibilidad de los hablantes y otras variables individuales como ya se ha mencionado.

Para concluir, dada la complejidad del problema, el poco trabajo publicado sobre la población de estudio en este contexto y el actual estado del arte de los modelos utilizados, se considera al trabajo realizado en esta tesis como un puntapié inicial cuyos resultados se espera den información relevante a futuras líneas de investigación sobre la temática.

Referencias

Baevski, A., Zhou, H., Mohamed, A., & Auli, M. (2020). wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations. *CoRR*, *abs/2006.11477*. https://arxiv.org/abs/2006.11477

Bredin, H. (2017). pyannote.metrics: A Toolkit for Reproducible Evaluation, Diagnostic, and Error Analysis of Speaker Diarization Systems. *Proc. Interspeech 2017*, 3587-3591. https://doi.org/10.21437/Interspeech.2017-411

Bredin, H. (2022). PYANNOTE.AUDIO 2.1 SPEAKER DIARIZATION PIPELINE: PRINCIPLE, BENCHMARK, AND RECIPE.

Bredin, H., & Laurent, A. (2021). End-to-end speaker segmentation for overlap-aware resegmentation.

Bredin, H., Yin, R., Coria, J. M., Gelly, G., Korshunov, P., Lavechin, M., Fustes, D., Titeux, H., Bouaziz, W., & Gill, M.-P. (2019). pyannote.audio: neural building blocks for speaker diarization.

Chen, G., Chai, S., Wang, G., Du, J., Zhang, W.-Q., Weng, C., Su, D., Povey, D., Trmal, J., Zhang, J., Jin, M., Khudanpur, S., Watanabe, S., Zhao, S., Zou, W., Li, X., Yao, X., Wang, Y., Wang, Y., ... Yan, Z. (2021). GigaSpeech: An Evolving, Multi-domain ASR Corpus with 10,000 Hours of Transcribed Audio.

Dernoncourt, F., Bui, T., & Chang, W. (2018). A Framework for Speech Recognition Benchmarking. *Proc. Interspeech 2018*.

Galvez, D., Diamos, G., Ciro, J., Cerón, J. F., Achorn, K., Gopi, A., Kanter, D., Lam, M., Mazumder, M., & Reddi, V. J. (2021). The People's Speech: A Large-Scale Diverse English Speech Recognition Dataset for Commercial Usage.

Geirhos, R., Jacobsen, J.-H., Michaelis, C., Zemel, R., Brendel, W., Bethge, M., & Wichmann, F. A. (2020). Shortcut learning in deep neural networks. *Nature Machine Intelligence*, *2*(11), 665-673. https://doi.org/10.1038/s42256-020-00257-z

Ghorbani, B., Firat, O., Freitag, M., Bapna, A., Krikun, M., Garcia, X., Chelba, C., & Cherry, C. (2021). Scaling Laws for Neural Machine Translation.

Glasser, A. T., Kushalnagar, K. R., & Kushalnagar, R. S. (2017). Feasibility of Using Automatic Speech Recognition with Voices of Deaf and Hard-of-Hearing Individuals. *Proceedings of the 19th International ACM SIGACCESS Conference on Computers and Accessibility*. https://doi.org/10.1145/3132525.3134819

Gutz, S., Stipancic, K., Yunusova, Y., Berry, J., & Green, J. (2022). Validity of Off-the-Shelf Automatic Speech Recognition for Assessing Speech Intelligibility and Speech Severity in Speakers With Amyotrophic Lateral Sclerosis. *Journal of speech, language, and hearing research : JSLHR*, *65*, 1-16. https://doi.org/10.1044/2022_JSLHR-21-00589

Jeong, J., Mondol, S., Kim, Y., & Lee, S. (2021). An Effective Learning Method for Automatic Speech Recognition in Korean CI Patients' Speech. *Electronics*, *10*(7).

Kafle, S., Glasser, A., Al-khazraji, S., Berke, L., Seita, M., & Huenerfauth, M. (2019). Artificial Intelligence Fairness in the Context of Accessibility Research on Intelligent Systems for People who are Deaf or Hard of Hearing.

Liu, T., & Yu, K. (2022). BER: Balanced Error Rate For Speaker Diarization.

Loshchilov, I., & Hutter, F. (2019). Decoupled Weight Decay Regularization.

Morris, A. C., Maier, V., & Green, P. (2004). From WER and RIL to MER and WIL: improved evaluation measures for connected speech recognition. *Proc. Interspeech 2004*, 2765-2768. https://doi.org/10.21437/Interspeech.2004-668

Radford, A., Wook Kim, J., Xu, T., Brockman, G., McLeavey, I., & Sutskever, C. (2022). Robust Speech Recognition via Large-Scale Weak Supervision. *arXiv:2212.04356*.

Semel, E., & Secord, A. (2006). The Gnats and Gnus Document Preparation System. *Clinical evaluation of language fundamentals, Spanish edition CELF-4*, *41*(7), 73+.

Sennrich, R., Haddow, B., & Birch, A. (2016). Neural Machine Translation of Rare Words with Subword Units.

Shor, J., Emanuel, D., Lang, O., Tuval, O., Brenner, M., Cattiau, J., Vieira, F., McNally, M., Charbonneau, T., Nollstadt, M., Hassidim, A., & Matias, Y. (2019). Personalizing ASR for Dysarthric and Accented Speech with Limited Data. *Interspeech 2019*. https://doi.org/10.21437/interspeech.2019-1427

Snyder, D., Garcia-Romero, D., Sell, G., Povey, D., & Khudanpur, S. (2018). X-Vectors: Robust DNN Embeddings for Speaker Recognition. *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 5329-5333. https://doi.org/10.1109/ICASSP.2018.8461375

Taboh, A., Shalom, D., Bosco, V., Denham, P., & Gattei, C. (2022). Evaluación del lenguaje oral en niños y niñas con hipoacusia: Los tests estandarizados y la edad auditiva [Spoken Language Assessment in Children with Hearing Impairment: Standardized Tests and Hearing Age]. *Revista Signos: Estudios de Lengua y Literatura*, *55*, 928-947. https://doi.org/10.4067/S0718-09342022000300928

Taboh, A., Shalom, D., & Gattei, C. (2021). Spoken language acquisition in children with prelingual hearing loss: A review of their morphosyntactic abilities at the sentence level. https://doi.org/10.31234/osf.io/rj8s9

Tripathi, M., Singh, D., & Susan, S. (2020). Speaker Recognition using SincNet and X-Vector Fusion.

Anexos

A. CELF 4

La versión 4 del **Clinical Evaluation of Language Fundamentals (CELF)** es una herramienta administrada de forma individual para la identificación, diagnóstico y seguimiento de la evaluación de trastornos del lenguaje y la comunicación. Puede ser aplicado en individuos de 5 a 21 años de edad.

Tipo	**ID**	**Pregunta**
Trial	1	Mi hermana está en el sexto grado
Trial	2	¿Enseña lectura el señor López?
Question	1	¿No terminaron los niños la prueba?
Question	2	¿Fue puesta la carta por correo?
Question	3	Esta nota fue enviada por mi maestra
Question	4	La bebita de Carmen jugó con la muñeca
Question	5	Mi amigo no llevó su almuerzo a la escuela.
Question	6	¿Se le olvidó al estudiante hacer su tarea?
Question	7	¿Decidió la familia comprar un auto nuevo?
Question	8	Pedro no encontró al amigo que quería jugar con él
Question	9	El gato que se sube a la mesa es el más travieso
Question	10	María preparó la cena y luego lavó los platos
Question	11	El niño sacó buenas notas y su papá lo llevó de paseo
Question	12	Ella decidió jugar básquetbol aunque le dolía la rodilla
Question	13	El conejito café se comió todas las zanahorias en el jardín
Question	14	Rosa quería comprarse el vestido, aunque no le quedaba bien
Question	15	El desayuno y la cena fueron preparados por Papá
Question	16	Los juguetes nuevos fueron donados por los niños y sus papás
Question	17	El señor que trae el correo a mi casa es mi vecino
Question	18	La ropa no fue doblada ni guardada por los niños
Question	19	La niña que había perdido su anillo estaba muy triste
Question	20	Por qué los niños están cansados, se van a acostar temprano
Question	21	El papá cortó madera, hizo un carrito y se lo regaló a su hijo
Question	22	Los niños no pudieron encontrar los abrigos que su papá empacó anoche
Question	23	El artista no pudo vender los cuadros que pintó el mes pasado
Question	24	El niño se bañó, se vistió, se desayunó y se fue a la escuela
Question	25	Los niños le mandaron flores a su abuelita, quien cumplió años el domingo.
Question	26	Si papá hubiera tenido dinero extra, nos hubiera llevado al circo
Question	27	La niña estaba alegre porque encontró su perrito que se le había perdido
Question	28	Los compañeros le mandaron una tarjeta a la alumna que se enfermo ayer
Question	29	Antes que el gatito fuera traído a casa, fueron a comprar comida
Question	30	Después que los niños cenaron, le pidieron a su papá permiso para ir afuera
Question	31	Después que hubieron jugado en la piscina, la mamá les pidió que se secaran
Question	32	Si los alumnos hubieran estudiado más, habrían sacado mejores notas

Tabla 18. Test CELF 4

B. Criterios de transcripción

A continuación, se detallan los criterios que se establecieron para la construcción del corpus que será utilizado para evaluar y optimizar las tareas de procesamiento automático del habla a lo largo de este trabajo.

- En la transcripción se apunta a escribir la secuencia de palabras que más se aproxime a la producción escuchada. Por palabras se refiere a vocablos válidos en español. Por ejemplo, si se escucha como producción la secuencia .ªvicala", como no es una palabra válida, se transcribe la palabra en español más cercana fonéticamente (.ªvícola", por ejemplo). El sesgo/varianza que se introduce en el ground truth por la persona que realiza la transcripción en términos de interpretación podría mejorarse en un futuro produciendo varias transcripciones independientes por diferentes personas ciegas a los objetivos de la tesis.
- En casos donde la palabra a transcribir no sea clara se marcará con ¿.ªl final. Se reserva ese símbolo para indicar que no queda claro cuál es la palabra. Ejemplo: .ªvícola?". De esta manera, en la etapa de evaluación del modelo se podrán correr pruebas contemplando o no estas situaciones.
- Si más de una palabra pudiera ser válida en la transcripción, se ingresarán ambas separadas por una barra (/). Ejemplo: "llegó/llevó"si hubiese dudas entre esas dos.
- No se tendrán en cuenta ni la sintaxis ni el significado de las palabras. Es decir, una transcripción puede ser una secuencia sin sentido de palabras válidas.
- Se tratará de transcribir el habla de niños/as de manera independiente de las oraciones que dice la entrevistadora. No debería haber influencia entre lo que se espera que el niño/a diga y lo que efectivamente dice.
- No se tendrán en cuenta signos de puntuación, exclamación e interrogación.
- Los segmentos de audio no relevantes para el estudio, se marcarán con el tag especial "ignore".
- En el contexto de las tareas de ASR, sólo se considerarán para su evaluación los segmentos de audio que pertenezcan a preguntas del estudio CELF 4, no serán tenidas en cuenta otras conversaciones que pudiera haber como producto del proceso de entrevista.

C. Esquema de datos

A continuación, se describen los esquemas de las tablas de la base de datos relacional que se utilizaron para guardar información de los resultados del pipeline de reconocimiento automático del habla.

Tablas

- **Metrics:** Tabla principal. Guarda información de la ejecución. Por ejemplo, modelo con el cual se realiza la inferencia, el archivo que se analizó, si se aplica filtro de ruido, métricas de diarización, imágenes generadas de los archivos como espectrograma y forma de onda en base 64, entre otros. El resto de las tablas son resultados específicos que están referenciados a una entrada de Metrics.
- **Results:** Esta tabla modela los resultados de desempeño del modelo para una ejecución en particular, por eso tiene una llave foránea (FK) que establece una relación 1 a 1 con la tabla Metrics. En esta tabla se guardan promedios de WER para la entrevista, desagregando por *speaker*, alucinaciones, entre otros
- **Hallucinations:** Esta tabla modela detalles de las alucinaciones para una ejecución en particular, por eso tiene una llave foránea (FK) que establece una relación 1 a N con la tabla Metrics contemplando que puede haber más de una alucinación durante el proceso de inferencia de una entrevista. Algunos atributos que están presentes en esta tabla son: Pregunta en la cual se hizo presente la alucinación del modelo, el WER que genera, sobre qué *speaker* sucedió, el producto de la alucinación del modelo, entre otros.

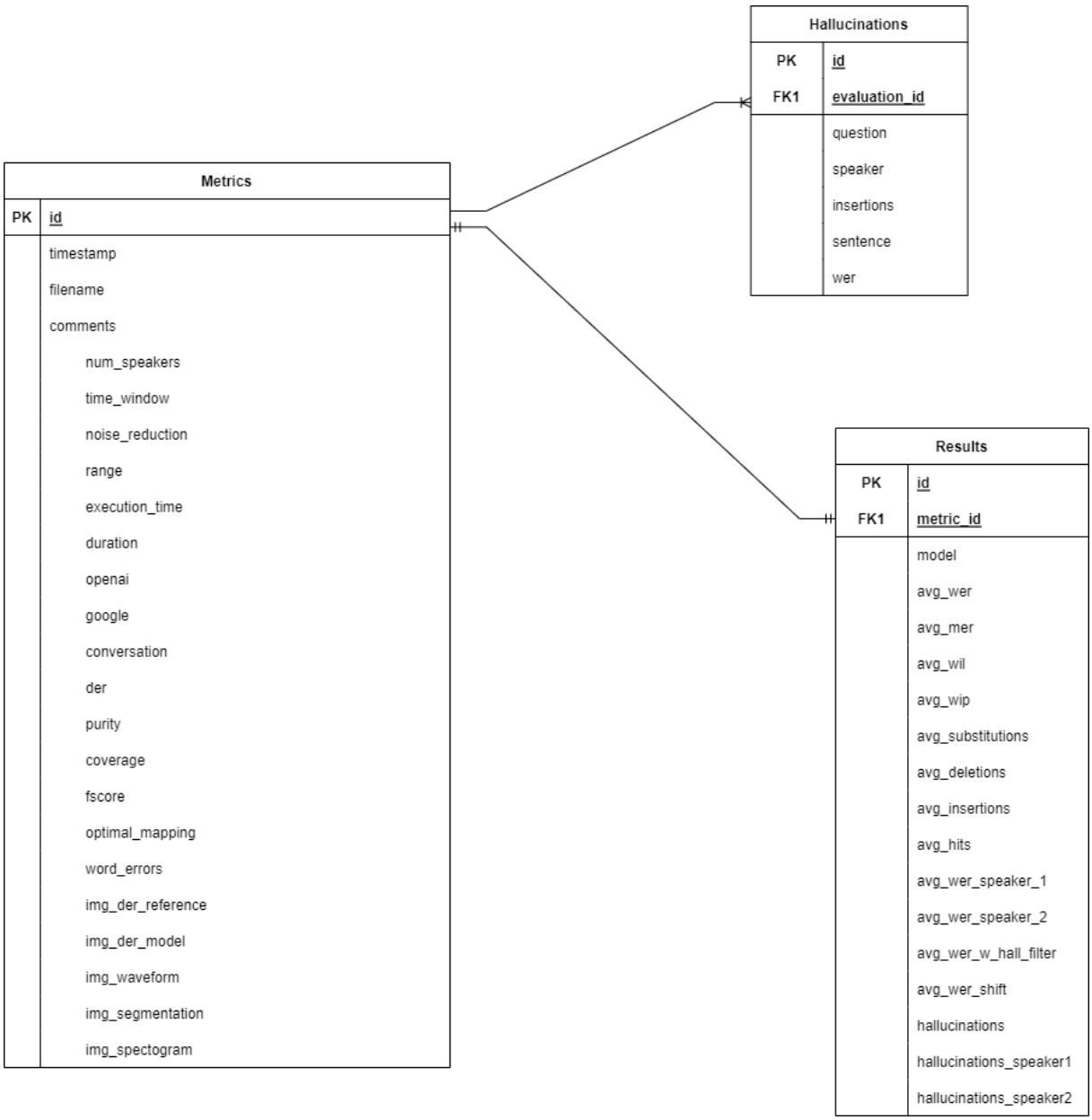

Figura 46. Esquema de tablas que modela el proceso reconocimiento automático del habla.

D. Hiperparámetros y configuración de entrenamiento

Parámetros utilizados para proceso de *fine tuning* de modelos.

Hiperparámetros	**Valor**
Steps	10000
Batch Size	32
Gradient Accumulation Steps	2
fp16 (Mixed Precision)	True
Evaluation Strategy	steps
Evaluation Steps	1000
Warmup Steps	1000
Optimizer	AdamW
Learning Rate Schedule	Linear
Learning Rate	1e-5
Loss function	cross-entropy
Metric for best Model	Word Error Rate
Greater is Better	False
Load best model at end	True

Tabla 19. Hiperparámetros de entrenamiento

E. Tabla de resultados

La tabla muestra el WER promedio por cada uno de los modelos, contemplando el uso, o no, de filtros de ruido y ordenándolos de forma ascendente según performance en niños/as entrevistados/as (speaker2). La última columna de la tabla refleja la diferencia promedio en WER que hay entre hablantes un modelo dado.

		avg_wer_speaker1		avg_wer_speaker2		avg_wer_speakers		shift_speakers
		mean	std	mean	std	mean	std	
openai	noise_reduction							
custom_small_augmented	False	0.0021	0.0072	0.0828	0.1211	0.0424	0.0600	0.0808
custom_small_cv11_augmented	False	0.0021	0.0072	0.0832	0.1192	0.0426	0.0590	0.0811
custom_small_no_augmented	False	0.0021	0.0073	0.0886	0.1199	0.0454	0.0592	0.0865
custom_tiny_cv11_augmented	False	0.0024	0.0076	0.1112	0.1516	0.0568	0.0750	0.1088
custom_small_augmented	True	0.0021	0.0072	0.1149	0.1650	0.0585	0.0819	0.1129
custom_small_cv11_augmented	True	0.0021	0.0074	0.1182	0.1858	0.0602	0.0923	0.1161
custom_tiny_augmented	False	0.0022	0.0074	0.1329	0.1925	0.0676	0.0955	0.1307
custom_tiny_no_augmented	False	0.0025	0.0079	0.1403	0.1834	0.0714	0.0908	0.1377
custom_tiny_cv11_augmented	True	0.0040	0.0115	0.2065	0.2693	0.1052	0.1333	0.2026
custom_small_no_augmented	True	0.0040	0.0106	0.2275	0.2894	0.1158	0.1432	0.2234
custom_tiny_augmented	True	0.0033	0.0103	0.2355	0.3165	0.1194	0.1571	0.2322
custom_tiny_no_augmented	True	0.0067	0.0183	0.3297	0.3801	0.1682	0.1873	0.3230
google	False	0.0317	0.0355	0.3381	0.3738	0.1849	0.1727	0.3064
custom_small_cv11	False	0.0166	0.0211	0.3755	0.4505	0.1960	0.2184	0.3589
medium-Spanish	False	0.0073	0.0163	0.3770	0.4482	0.1922	0.2211	0.3697
openai/whisper-medium	False	0.0072	0.0165	0.3840	0.4699	0.1956	0.2321	0.3768
large-Spanish	False	0.0054	0.0140	0.3919	0.4834	0.1987	0.2395	0.3865
openai/whisper-large	False	0.0054	0.0136	0.3943	0.5098	0.1999	0.2529	0.3888
openai/whisper-small	False	0.0161	0.0269	0.3957	0.4636	0.2059	0.2251	0.3796
small-Spanish	False	0.0143	0.0231	0.4100	0.4824	0.2121	0.2352	0.3957
custom_small_cv11	True	0.0211	0.0293	0.4194	0.4924	0.2203	0.2374	0.3983
custom_tiny_cv11	False	0.0674	0.0727	0.4277	0.4678	0.2475	0.2037	0.3603
large-Spanish	True	0.0083	0.0172	0.4361	0.5071	0.2222	0.2501	0.4278
openai/whisper-large	True	0.0077	0.0149	0.4401	0.5200	0.2239	0.2568	0.4324
medium-Spanish	True	0.0105	0.0171	0.4452	0.4842	0.2279	0.2372	0.4347
openai/whisper-medium	True	0.0115	0.0226	0.4533	0.5401	0.2324	0.2653	0.4418
small-Spanish	True	0.0233	0.0315	0.4666	0.5179	0.2450	0.2486	0.4433
openai/whisper-base	False	0.0314	0.0371	0.4700	0.5363	0.2507	0.2546	0.4386
base-Spanish	False	0.0333	0.0425	0.4731	0.5239	0.2532	0.2472	0.4397
custom_tiny_cv11	True	0.0876	0.0973	0.4746	0.5128	0.2811	0.2170	0.3870
google	True	0.2578	0.2773	0.4856	0.4875	0.3717	0.1253	0.2278
openai/whisper-small	True	0.0212	0.0321	0.4881	0.5622	0.2546	0.2721	0.4668
openai/whisper-tiny	False	0.0695	0.0767	0.4931	0.5472	0.2813	0.2429	0.4236
openai/whisper-base	True	0.0451	0.0543	0.5224	0.5695	0.2838	0.2645	0.4774
base-Spanish	True	0.0475	0.0593	0.5261	0.5577	0.2868	0.2570	0.4786
openai/whisper-tiny	True	0.0971	0.1122	0.5290	0.5608	0.3130	0.2363	0.4319
tiny-Spanish	False	0.0687	0.0753	0.5321	0.5764	0.3004	0.2569	0.4634
	True	0.0937	0.1077	0.5591	0.5759	0.3264	0.2436	0.4654

Tabla 20. Tabla con resultados de performance para distintos modelos y variantes de pre procesamiento de audio.

F. Hardware para entrenamiento e inferencia

Los modelos de diarización y ASR utilizados en este trabajo, dada la cantidad de parámetros, requieren de cantidades considerables de cómputo y, en particular, GPU. Esto es necesario tanto para realizar tareas de inferencia como de entrenamiento.

El departamento de sistemas de la Universidad Torcuato Di Tella permitió el acceso a un servidor con configuración de hardware apta para las tareas necesarias. Con el objetivo de dar información que brinde contexto, referencia y permita reproducibilidad de los resultados desarrollados en esta tesis, se detalla el equipamiento utilizado.

Componente	**Cantidad**	**Descripción**
Procesador	1	Intel® Xeon® Silver 4310 2.1G, 12C/24T, 10.4GT/s, 18M Cache
Memoria RAM	NA	256GB
Placa de Video	2	Nvidia Ampere 100 80GB
Disco Rígido	2	423GB / 1.1TB
Sistema Operativo	NA	Ubuntu 22.04.2 LTS

Tabla 21. Configuración de hardware utilizado para entrenamiento de modelos

G. Librerías

Con el objetivo de facilitar la reproducción de resultados o la metodología de este trabajo se deja el detalle de las librerías utilizadas con sus respectivas versiones.

Nombre	Descripción	Enlace	Versión
noisereduce	Filtro de ruido para señales en dominio de tiempo	https://pypi.org/project/noisereduce	2.0.1
jiwer	Cálculo de métricas de evaluación de sistemas ASR	https://pypi.org/project/jiwer	2.5.1
librosa	Análisis de audio y música	https://librosa.org/doc	0.10.0
pydub	Edición de archivos de audio	https://pypi.org/project/pydub	0.25.1
transformers	Librería que implementa arquitectura transformer de redes neuronales	https://huggingface.co/docs/transformers	4.28.1
datasets	Filtro de ruido para señales en dominio de tiempo	https://pypi.org/project/noisereduce/	2.0.1
audio-metadata	Extraer metainformación de archivos de audio	https://pypi.org/project/audio-metadata	0.11.1
soundfile	Leer y escribir audio en distintos formatos y plataformas	https://pypi.org/project/soundfile	0.12.1
audiomentations	Data augmentation para archivos de audio	https://pypi.org/project/audiomentations	0.29.0
pyannote	Filtro de ruido para señales en dominio de tiempo	https://pypi.org/project/pyannote.audio	5.0.0
whisper	Modelo pre entrenado para tareas de automatic speech recognition	https://pypi.org/project/whisper	20230117
google-cloud-speech	API para consumir el servicio de ASR de Google Cloud Platform	https://pypi.org/project/google-cloud-speech	2.19.0
codecarbon	Filtro de ruido para señales en dominio de tiempo	https://pypi.org/project/codecarbon	2.1.4
typer	Framework para desarrollo de aplicaciones de consola	https://pypi.org/project/typer	0.7.0

Tabla 22. Librerías utilizadas en el contexto de esta tesis